**Звіт с створення простого додатку для редактування фото**

загальні інструкції:

* Цей веб-проект ставить перед вами завдання створити невелику веб-програму, яка дозволить вам виконувати базове редагування фотографій і відео за допомогою веб-камери та деяких попередньо визначених зображень.
* Користувачі програми повинні мати можливість вибирати зображення зі списку зображень, які можна накладати (наприклад, рамку для картини чи інші об’єкти типу «ми не хочемо знати, для чого ви це використовуєте»), робити знімки за допомогою своєї веб-камери та помилуйтеся результатом, який слід змішати обидві картинки.  
  Усі зроблені зображення мають бути загальнодоступними, лайками та коментарями.
* Ваша веб-програма не повинна створювати жодних помилок, попереджень або рядків журналу на будь-якій консолі, на стороні сервера та клієнта. Тим не менш, через відсутність HTTPS допускаються будь-які помилки, пов’язані з getUserMedia().
* Ви повинні використовувати мову ony PHP для створення програми на стороні сервера лише зі стандартною бібліотекою.
* На стороні клієнта ваші сторінки повинні використовувати HTML, CSS і JavaScript.
* Усі фреймворки, мікрофреймворки чи бібліотеки, які ви не створюєте, повністю заборонені, за винятком фреймворків CSS, яким заборонений JavaScript не потрібен.
* Ви повинні використовувати абстракційний драйвер PDO для зв’язку з вашою базою даних, яка має бути доступною для запитів за допомогою SQL. Режим помилки цього драйвера повинен бути встановлений на PDO::ERRMODE\_EXCEPTION
* У вашій програмі не повинно бути жодних витоків безпеки. Ви повинні розглядати принаймні випадки, зазначені в обов'язковій частині. Тим не менш, ми заохочуємо вас глибше заглибитися в безпеку своєї програми, подумати про конфіденційність ваших даних!
* Ви можете використовувати будь-який веб-сервер, як-от Apache, Nginx або навіть вбудований веб-сервер
* Ваша веб-програма має бути принаймні сумісною з Firefox (>= 41) і Chrome (>= 46)

Обов'язкова частина  
  
1 - Загальні риси   
Ви будете розробляти веб-додаток. Навіть якщо це не потрібно, спробуйте структурувати свою програму (наприклад, як програму MVC).  
Ваш веб-сайт повинен мати пристойний макет сторінки (тобто принаймні верхній колонтитул, головний розділ і нижній колонтитул), мати можливість правильно відображатися на мобільних пристроях і мати адаптований макет для невеликої роздільної здатності.  
Усі ваші форми мають мати правильні перевірки, а весь сайт має бути захищеним. Цей пункт є ОБОВ’ЯЗКОВИМ і має бути перевірений під час оцінювання вашої заявки. Щоб мати уявлення, ось деякі речі, які НЕ вважаються БЕЗПЕЧНИМИ:

* Зберігайте звичайні або незашифровані паролі в базі даних.
* Пропонуйте можливість вставляти HTML або «користувацький» JavaScript у погано захищені змінні. • Пропонуйте можливість завантажувати небажаний вміст на сервер.
* Запропонуйте можливість змінити SQL-запит.
* Використовуйте зовнішню форму для маніпулювання так званими особистими даними

2 - Функції користувача

* Програма повинна дозволяти користувачеві зареєструватися, запитавши принаймні дійсну адресу електронної пошти, ім’я користувача та пароль принаймні мінімального рівня складності.
* Після завершення процесу реєстрації користувач повинен підтвердити свій обліковий запис за допомогою унікального посилання, надісланого на адресу електронної пошти, заповнену в реєстраційній формі.
* Тоді користувач повинен мати можливість підключитися до вашої програми, використовуючи своє ім’я користувача та пароль. Він також повинен мати можливість наказати програмі надіслати лист для повторної ініціалізації пароля, якщо він забуде свій пароль.
* Користувач повинен мати можливість від'єднатися одним клацанням миші в будь-який час на будь-якій сторінці.
* Після підключення користувач повинен змінити своє ім'я користувача, адресу електронної пошти або пароль.

3 - Функції галереї

* Ця частина має бути загальнодоступною та має відображати всі зображення, відредаговані всіма користувачами, упорядковані за датою створення. Він також має дозволяти (тільки) підключеному користувачеві лайкати їх та/або коментувати.
* Коли зображення отримує новий коментар, автор зображення повинен бути повідомлений електронною поштою. За замовчуванням це налаштування має бути встановлено як true, але його можна вимкнути в налаштуваннях користувача.
* Список зображень має бути розбитий на сторінки, щонайменше 5 елементів на сторінку.

4 - Можливості редагування   
малюнок V1  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
|                      header                           |  
|.   main                                  side.      |  
|                      footer                             |  
---------------------------------------------------------  
Ця частина має бути доступною лише для користувачів, які автентифіковані/підключені, і обережно відхиляти всіх інших користувачів, які намагаються отримати до неї доступ без успішного входу в систему.  
Ця сторінка має містити 2 розділи:

* Основний розділ, що містить попередній перегляд веб-камери користувача, список накладних зображень і кнопку, що дозволяє зробити знімок.
* Бічна частина, де відображаються мініатюри всіх попередніх зроблених знімків. Макет вашої сторінки зазвичай має виглядати так, як показано на малюнку V.1.
* Зображення, що накладаються, мають бути доступними для вибору, а кнопка, яка дозволяє зробити знімок, має бути неактивною (не можна натискати), доки не вибрано зображення, що накладається.
* Створення остаточного зображення (тобто, серед іншого, накладання двох зображень) має виконуватися на стороні сервера, у PHP.
* Оскільки не всі мають веб-камеру, вам слід дозволити завантаження зображення користувача замість того, щоб знімати його за допомогою веб-камери.
* Користувач повинен мати можливість видаляти свої відредаговані зображення, але лише свої, а не творіння інших користувачів.

5 - Обмеження та обов’язкові речі Підводячи підсумок, ваша кінцева програма повинна враховувати такі технологічні рішення:

* Авторизовані мови:
* [Сервер] PHP
* [Клієнт] HTML - CSS - JavaScript (тільки з вбудованим API браузера) • Авторизовані фреймворки:
* [Сервер] Немає
* [Клієнт] CSS Frameworks допускається, якщо він не додає заборонений JavaScript.

Ваш проект обов'язково повинен містити:

* Файл index.php, який містить точку входу вашого сайту та розташований у корені вашої ієрархії файлів.
* Файл config/setup.php, здатний створювати або відтворювати схему бази даних за допомогою інформації, що міститься у файлі config/database.php.
* Файл config/database.php, що містить вашу конфігурацію бази даних, екземпляр якого буде створено через PDO у такому форматі:
* Файл README з описом розгортання проекту та його запуском.

<?php  
        $DB\_DSN = ....  
        $DB\_USER=....  
        $DB\_PASSWORD=...

* DSN (Ім’я джерела даних) містить необхідну інформацію для підключення до бази даних, наприклад «mysql:dbname=testdb;host=127.0.0.1». Як правило, DSN складається з назви драйвера PDO, після якого йде спеціальний синтаксис для цього драйвера. Для отримання додаткової інформації перегляньте документ PDO кожного драйвера1.

6 - Бонусна частина Якщо необхідна частина виконана повністю і ідеально, ви можете додати будь-який бонус, який забажаєте; Їх оцінять ваші рецензенти. Однак ви все одно повинні дотримуватися вимог у бонусних частинах (тобто обробка зображень має виконуватися на стороні сервера).  
Якщо вам не вистачає натхнення, ось кілька підказок:

* «AJAXify» обмінюється з сервером.
* Запропонуйте попередній перегляд відредагованого результату безпосередньо на веб-камері. ми слід зазначити, що це набагато простіше, ніж здається.
* Виконайте нескінченну розбивку сторінок галереї на сайті.
* Запропонувати користувачеві можливість ділитися своїми зображеннями в соціальних мережах.
* Відтворення анімованого GIF-файлу.

**Виконання**

Лістинги:

/noerrors.fun/

setup.php:

<?php

require\_once 'config/database.php';

try {

// Якщо немає підключення до БД, створюємо її

if (!$pdo) {

$pdo = new PDO("mysql:host=" . DB\_HOST, DB\_USER, DB\_PASS, [

PDO::ATTR\_ERRMODE => PDO::ERRMODE\_EXCEPTION

]);

$pdo->exec("CREATE DATABASE IF NOT EXISTS " . DB\_NAME);

$pdo->exec("USE " . DB\_NAME);

}

// Створення таблиці користувачів

$pdo->exec("CREATE TABLE IF NOT EXISTS users (

id INT AUTO\_INCREMENT PRIMARY KEY,

username VARCHAR(50) NOT NULL UNIQUE,

email VARCHAR(100) NOT NULL UNIQUE,

password VARCHAR(255) NOT NULL,

verified BOOLEAN DEFAULT 0,

activation\_code VARCHAR(255) NULL,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

reset\_token VARCHAR(64) DEFAULT NULL,

reset\_expires DATETIME DEFAULT NULL,

photo\_user\_path VARCHAR(255) DEFAULT '/uploads/images/default\_photo\_user.png',

notify\_new\_comment BOOLEAN DEFAULT 1

)");

// Таблиця для збереження зображень

$pdo->exec("CREATE TABLE IF NOT EXISTS images (

id INT AUTO\_INCREMENT PRIMARY KEY,

user\_id INT NOT NULL,

image\_path VARCHAR(255) NOT NULL,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

FOREIGN KEY (user\_id) REFERENCES users(id) ON DELETE CASCADE

)");

// Таблиця історії редагувань

$pdo->exec("CREATE TABLE IF NOT EXISTS image\_edits (

id INT AUTO\_INCREMENT PRIMARY KEY,

image\_id INT NOT NULL,

edit\_details TEXT NOT NULL,

edited\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

FOREIGN KEY (image\_id) REFERENCES images(id) ON DELETE CASCADE

)");

// Таблиця коментарів

$pdo->exec("CREATE TABLE IF NOT EXISTS comments (

id INT AUTO\_INCREMENT PRIMARY KEY,

image\_id INT NOT NULL,

user\_id INT NOT NULL,

comment TEXT NOT NULL,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

FOREIGN KEY (image\_id) REFERENCES images(id) ON DELETE CASCADE,

FOREIGN KEY (user\_id) REFERENCES users(id) ON DELETE CASCADE

)");

// Таблиця лайків

$pdo->exec("CREATE TABLE IF NOT EXISTS likes (

id INT AUTO\_INCREMENT PRIMARY KEY,

image\_id INT NOT NULL,

user\_id INT NOT NULL,

like\_value INT NOT NULL,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

FOREIGN KEY (image\_id) REFERENCES images(id) ON DELETE CASCADE,

FOREIGN KEY (user\_id) REFERENCES users(id) ON DELETE CASCADE

)");

// Якщо все успішно, перенаправляємо на головну сторінку

header("Location: index.php");

exit;

} catch (PDOException $e) {

die("Помилка встановлення БД: " . $e->getMessage());

}

?>

save\_photo.php:

<?php

session\_start();

require\_once 'http://localhost/photo\_editing\_app/config/database.php';

$data = json\_decode(file\_get\_contents("php://input"));

$image = $data->image;

$image = str\_replace('data:image/png;base64,', '', $image);

$image = str\_replace(' ', '+', $image);

$file = 'uploads/' . uniqid() . '.png';

if (file\_put\_contents($file, base64\_decode($image))) {

$pdo = new PDO($DB\_DSN, $DB\_USER, $DB\_PASSWORD);

$stmt = $pdo->prepare("INSERT INTO photos (user\_id, image\_path) VALUES (?, ?)");

$stmt->execute([$\_SESSION['user\_id'], $file]);

echo json\_encode(["message" => "Фото збережено!", "path" => $file]);

} else {

echo json\_encode(["message" => "Помилка збереження"]);

}

?>

logout.php:

<?php

session\_start();

session\_unset();

session\_destroy();

header("Location: /index.php");

exit();

?>

index.php:

<?php

session\_start();

require\_once 'config/database.php';

require\_once 'includes/header.php';

$page = $\_GET['page'] ?? 'home';

?>

<main id="main-content">

<?php include "pages/$page.php"; ?>

</main>

<?php require\_once 'includes/footer.php'; ?>

actions.php:

<?php

session\_start();

require 'config/database.php'; // Підключення до бази даних

// Ініціалізація PDO

$pdo = new PDO("mysql:host=" . DB\_HOST . ";dbname=" . DB\_NAME . ";charset=utf8", DB\_USER, DB\_PASS, [

PDO::ATTR\_ERRMODE => PDO::ERRMODE\_EXCEPTION

]);

// Головна зміна — підтримка GET-параметру 'action'

$action = $\_POST['action'] ?? $\_GET['action'] ?? '';

switch ($action) {

case 'register':

$username = trim($\_POST['username']);

$email = trim($\_POST['email']);

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

$confirm\_password = $\_POST['confirm\_password'];

if ($password !== $confirm\_password) {

$\_SESSION['message'] = 'Пароль і повторне введення паролю не співпадають';

header("Location: index.php?page=register");

exit;

}

if (strlen($password) < 8) {

$\_SESSION['message'] = 'Мінімальна довжина паролю 8 символів';

header("Location: index.php?page=register");

exit;

}

$stmt = $pdo->prepare("SELECT id FROM users WHERE email = ?");

$stmt->execute([$email]);

if ($stmt->fetch()) {

$\_SESSION['message'] = 'Цей email вже використовується!';

header("Location: index.php?page=register");

exit;

}

$hashed\_password = password\_hash($password, PASSWORD\_DEFAULT);

$activation\_code = md5(uniqid(rand(), true));

$default\_photo\_user\_path = '/uploads/images/default\_photo\_user.png';

$stmt = $pdo->prepare("INSERT INTO users (username, email, password, activation\_code, photo\_user\_path) VALUES (?, ?, ?, ?, ?)");

$stmt->execute([$username, $email, $hashed\_password, $activation\_code, $default\_photo\_user\_path]);

$verification\_link = "https://noerrors.fun/actions.php?action=verify&code=$activation\_code";

mail($email, "Підтвердження реєстрації", "Перейдіть за посиланням для активації: $verification\_link");

$\_SESSION['message'] = 'Реєстрація успішна! Перевірте email для активації.';

header("Location: index.php?page=login");

exit;

case 'login':

$email = trim($\_POST['email']);

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

$stmt = $pdo->prepare("SELECT id, password, verified FROM users WHERE email = ?");

$stmt->execute([$email]);

$user = $stmt->fetch();

if ($user && password\_verify($password, $user['password'])) {

if ($user['verified'] == 0) {

$\_SESSION['message'] = 'Підтвердіть email перед входом.';

header("Location: index.php?page=login");

exit;

}

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

header("Location: index.php?page=dashboard");

exit;

}

$\_SESSION['message'] = 'Невірний email або пароль';

header("Location: index.php?page=login");

exit;

case 'verify':

// Обробка GET['code']

if (!empty($\_GET['code'])) {

$activation\_code = $\_GET['code'];

$stmt = $pdo->prepare("SELECT id FROM users WHERE activation\_code = :code AND verified = 0");

$stmt->execute(['code' => $activation\_code]);

$user = $stmt->fetch(PDO::FETCH\_ASSOC);

if ($user) {

$updateStmt = $pdo->prepare("UPDATE users SET verified = 1, activation\_code = NULL WHERE id = :id");

$updateStmt->execute(['id' => $user['id']]);

$\_SESSION['message'] = '✅ Верифікація успішна! Тепер ви можете увійти.';

} else {

$\_SESSION['message'] = '❌ Невірний або вже активований код.';

}

} else {

$\_SESSION['message'] = '❌ Відсутній код активації.';

}

header("Location: index.php?page=login");

exit;

case 'forgot\_password':

$email = trim($\_POST['email']);

$stmt = $pdo->prepare("SELECT id FROM users WHERE email = ?");

$stmt->execute([$email]);

$user = $stmt->fetch();

if ($user) {

$reset\_code = md5(uniqid(rand(), true));

$stmt = $pdo->prepare("UPDATE users SET reset\_token = ?, reset\_expires = DATE\_ADD(NOW(), INTERVAL 1 HOUR) WHERE email = ?");

$stmt->execute([$reset\_code, $email]);

$reset\_link = "http://noerrors.fun/index.php?page=reset\_password&token=$reset\_code";

mail($email, "Скидання паролю", "Перейдіть за посиланням для скидання паролю: $reset\_link");

}

$\_SESSION['message'] = 'Якщо email існує в системі, на нього буде надіслано інструкції.';

header("Location: index.php?page=login");

exit;

case 'reset\_password':

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

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

$confirm\_password = $\_POST['confirm\_password'];

if ($password !== $confirm\_password) {

$\_SESSION['message'] = 'Паролі не співпадають.';

header("Location: index.php?page=reset\_password&token=$token");

exit;

}

if (strlen($password) < 8) {

$\_SESSION['message'] = 'Мінімальна довжина паролю 8 символів';

header("Location: index.php?page=reset\_password&token=$token");

exit;

}

$stmt = $pdo->prepare("SELECT id FROM users WHERE reset\_token = ? AND reset\_expires > NOW()");

$stmt->execute([$token]);

$user = $stmt->fetch();

if ($user) {

$hashed\_password = password\_hash($password, PASSWORD\_DEFAULT);

$stmt = $pdo->prepare("UPDATE users SET password = ?, reset\_token = NULL, reset\_expires = NULL WHERE id = ?");

$stmt->execute([$hashed\_password, $user['id']]);

$\_SESSION['message'] = 'Пароль успішно змінено!';

header("Location: index.php?page=login");

exit;

} else {

$\_SESSION['message'] = 'Невірний або прострочений токен.';

header("Location: index.php?page=reset\_password&token=$token");

exit;

}

case 'logout':

session\_destroy();

header("Location: index.php?page=login");

exit;

default:

header("Location: index.php");

exit;

}

/noerrors.fun/pages/

dashboard.php:

<?php

session\_start();

require 'config/database.php';

if (!isset($\_SESSION['user\_id'])) {

header("Location: index.php?page=login");

exit;

}

$pdo = new PDO("mysql:host=" . DB\_HOST . ";dbname=" . DB\_NAME . ";charset=utf8", DB\_USER, DB\_PASS, [

PDO::ATTR\_ERRMODE => PDO::ERRMODE\_EXCEPTION

]);

$stmt = $pdo->prepare("SELECT username, email, created\_at, notify\_new\_comment, photo\_user\_path FROM users WHERE id = ?");

$stmt->execute([$\_SESSION['user\_id']]);

$user = $stmt->fetch();

if (!$user) {

session\_destroy();

header("Location: index.php?page=login");

exit;

}

$photoPath = !empty($user['photo\_user\_path']) ? $user['photo\_user\_path'] : '/uploads/images/default\_photo\_user.png';

$username = $user['username'];

function getUserImages($username) {

$dir = \_\_DIR\_\_ . '/../uploads/images/';

$webPath = '/uploads/images/';

$files = [];

if (is\_dir($dir)) {

foreach (scandir($dir) as $file) {

if (in\_array(pathinfo($file, PATHINFO\_EXTENSION), ['jpg', 'jpeg', 'png', 'gif'])) {

if (strpos($file, $username . '\_') === 0 || $file === 'default\_photo\_user.png') {

$files[] = $webPath . $file;

}

}

}

}

return $files;

}

$userImages = getUserImages($username);

?>

<!DOCTYPE html>

<html lang="uk">

<head>

<meta charset="UTF-8">

<title>Кабінет користувача</title>

<link rel="stylesheet" href="/assets/css/dashboard.css">

</head>

<body>

<div class="container-user-dashboard">

<div class="user-dashboard">

<h2>Кабінет користувача <span id="username-dashboard-hello"><?php echo htmlspecialchars($username); ?></span></h2>

<form id="dashboard-form" action="/actions/save\_profile.php" method="post" enctype="multipart/form-data">

<div class="content">

<div class="photo-section">

<div class="photo">

<img src="<?php echo htmlspecialchars($photoPath); ?>" alt="Фото користувача" id="user-photo-preview" data-default="<?php echo htmlspecialchars($photoPath); ?>">

</div>

<div class="photo-buttons">

<label for="photo-user-input" class="photo-upload-btn">Вибрати файл</label>

<input type="file" name="photo\_user" accept="image/\*" id="photo-user-input">

<div id="selected-file-name" class="file-name-display">Файл не вибрано</div>

</div>

</div>

<div class="info-section">

<div class="info-item"><strong><label>Ім’я користувача:</label></strong>

<span><input type="text" name="username" value="<?php echo htmlspecialchars($user['username']); ?>" required></span>

</div>

<div class="info-item"><strong><label>Email:</label></strong>

<span><input type="email" name="email" value="<?php echo htmlspecialchars($user['email']); ?>" required></span>

</div>

<div class="info-item"><strong><label>Дата реєстрації:</label></strong>

<span><input type="text" value="<?php echo date('d.m.Y, H:i:s', strtotime($user['created\_at'])); ?>" readonly></span>

</div>

<div class="info-item">

<strong><label for="notify\_new\_comment">Повідомляти про новий коментар:</label></strong>

<span>

<input type="checkbox" id="notify\_new\_comment" name="notify\_new\_comment" value="1" <?php echo ($user['notify\_new\_comment']) ? 'checked' : ''; ?>>

</span>

</div>

<input type="hidden" name="old\_photo\_path" value="<?php echo htmlspecialchars($user['photo\_user\_path']); ?>">

<input type="hidden" name="selected\_server\_photo" id="selected\_server\_photo" value="">

<input type="hidden" name="photo\_user\_path" id="photo\_user\_path" value="<?php echo htmlspecialchars($photoPath); ?>">

<div class="actions">

<a href="index.php?page=forgot\_password" class="change-password">Змінити пароль</a>

<button type="submit" class="save">Зберегти</button>

<button type="reset" class="cancel">Відмінити</button>

</div>

</div>

</div>

<div class="image-server-selector">

<button type="button" id="toggle-image-panel" class="photo-server-btn">Обрати зображення з сервера</button>

<div id="server-image-panel" class="hidden">

<?php foreach ($userImages as $img): ?>

<img src="<?php echo htmlspecialchars($img); ?>" data-path="<?php echo htmlspecialchars($img); ?>" alt="" class="selectable-server-image">

<?php endforeach; ?>

</div>

</div>

</form>

</div>

</div>

<script src="/assets/js/dashboard.js"></script>

</body>

</html>

edit.php:

<?php

require\_once 'http://localhost/photo\_editing\_app/includes/header.php';

require\_once 'http://localhost/photo\_editing\_app/config/database.php';

require\_once 'http://localhost/photo\_editing\_app/includes/functions.php';

if (!isset($\_SESSION['user'])) {

header("Location: http://localhost/photo\_editing\_app/pages/login.php");

exit();

}

echo "<h2>Редагування фото</h2>";

?>

<div class="editor-container">

<video id="camera" autoplay></video>

<canvas id="canvas"></canvas>

<input type="file" id="upload" accept="image/\*">

<button id="capture">Зробити фото</button>

<button id="save">Зберегти фото</button>

</div>

<script src="http://localhost/photo\_editing\_app/js/script.js"></script>

<?php

require\_once 'http://localhost/photo\_editing\_app/includes/footer.php';

?>

editor.php:

<?php

// Перевірка авторизації

session\_start();

echo "<link rel='stylesheet' href='/assets/css/editor.css'>";

if (!isset($\_SESSION['user\_id'])) {

echo "<p>Будь ласка, увійдіть у систему, щоб редагувати зображення.</p>";

exit;

}

// Підключення до БД

require\_once \_\_DIR\_\_ . '/../config/database.php';

// Отримання ID зображення з URL

$imagePath = null;

$imageId = null;

if (isset($\_GET['id'])) {

$imageId = intval($\_GET['id']);

$stmt = $pdo->prepare("SELECT image\_path FROM images WHERE id = ?");

$stmt->execute([$imageId]);

$image = $stmt->fetch(PDO::FETCH\_ASSOC);

if ($image && file\_exists(\_\_DIR\_\_ . '/../' . ltrim($image['image\_path'], '/'))) {

$imagePath = $image['image\_path'];

}

}

if (!$imagePath) {

echo "<p style='color:red; text-align:center;'>Зображення не знайдено або не вказано.</p>";

$imagePath = '/uploads/images/default\_photo\_user.png';

}

?>

<div class="editor-container"

data-image-path="<?= htmlspecialchars($imagePath) ?>"

data-image-id="<?= $imageId !== null ? (int)$imageId : '' ?>">

<div class="main-editor">

<div class="image-wrapper">

<canvas id="editor-canvas"></canvas>

</div>

<div class="editor-buttons">

<button class="save-button" onclick="saveEditedImage()" disabled>💾 Зберегти накладання</button>

<button class="reset-button" onclick="resetOverlays()">🔄 Відмінити накладання</button>

</div>

</div>

<?php include \_\_DIR\_\_ . '/../includes/aside.php'; ?>

</div>

<!-- Підключення JS -->

<script src="/assets/js/editor.js"></script>

forgot\_password.php:

<h2>Відновлення паролю</h2>

<form action="actions.php" method="post">

<input type="hidden" name="action" value="forgot\_password">

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

<input type="email" name="email" id="email" placeholder="Введіть ваш email" required>

<div><button type="submit">Надіслати інструкції</button></div>

</form>

<div><a href="index.php?page=login">Повернутися до входу</a></div>

<?php

if (isset($\_SESSION['message'])) {

echo "<p style='color:red;'>" . $\_SESSION['message'] . "</p>";

unset($\_SESSION['message']);

}

?>

gallery.php:

<?php

session\_start();

require\_once \_\_DIR\_\_ . '/../config/database.php';

$pdo = new PDO("mysql:host=" . DB\_HOST . ";dbname=" . DB\_NAME . ";charset=utf8", DB\_USER, DB\_PASS, [

PDO::ATTR\_ERRMODE => PDO::ERRMODE\_EXCEPTION

]);

$logged\_in = isset($\_SESSION['user\_id']);

$user\_id = $logged\_in ? $\_SESSION['user\_id'] : null;

// Pagination

$page = isset($\_GET['page\_num']) ? max(1, (int)$\_GET['page\_num']) : 1;

$limit = 12;

$offset = ($page - 1) \* $limit;

// Total count for pagination

$total\_stmt = $pdo->query("SELECT COUNT(\*) FROM images");

$total\_images = $total\_stmt->fetchColumn();

$total\_pages = ceil($total\_images / $limit);

// Get images

$stmt = $pdo->prepare("

SELECT

images.id,

images.image\_path,

images.user\_id,

images.created\_at,

users.username,

(SELECT COUNT(\*) FROM comments WHERE comments.image\_id = images.id) AS comment\_count,

(SELECT COUNT(\*) FROM likes WHERE likes.image\_id = images.id AND likes.like\_value = 1) AS like\_count,

(SELECT COUNT(\*) FROM likes WHERE likes.image\_id = images.id AND likes.like\_value = -1) AS dislike\_count

FROM images

JOIN users ON images.user\_id = users.id

ORDER BY images.created\_at DESC

LIMIT :limit OFFSET :offset

");

$stmt->bindValue(':limit', $limit, PDO::PARAM\_INT);

$stmt->bindValue(':offset', $offset, PDO::PARAM\_INT);

$stmt->execute();

$images = $stmt->fetchAll(PDO::FETCH\_ASSOC);

?>

<!DOCTYPE html>

<html lang="uk">

<head>

<meta charset="UTF-8">

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

<title>Галерея</title>

<link rel="stylesheet" href="/assets/css/gallery.css">

</head>

<body>

<div class="gallery-container">

<?php if ($logged\_in): ?>

<div class="camera-block">

<video id="video" autoplay></video>

<div class="no-camera" id="no-camera" style="display: none;">Веб-камера відсутня</div>

<button id="screenshot-btn" disabled>Зберегти скріншот</button>

</div>

<?php endif; ?>

<div class="gallery-grid">

<?php foreach ($images as $img): ?>

<div class="image-card">

<a href="index.php?page=image\_info&id=<?= $img['id'] ?>">

<img src="<?= htmlspecialchars($img['image\_path']) ?>" alt="Зображення">

</a>

<div class="image-info">

<div>Користувач: <?= htmlspecialchars($img['username']) ?></div>

<div>Дата: <?= htmlspecialchars($img['created\_at']) ?></div>

<div>Коментарі: <?= (int) $img['comment\_count'] ?></div>

<div class="like-section" data-image-id="<?= $img['id'] ?>">

<button class="like-btn" <?= $logged\_in ? '' : 'disabled' ?>>👍 <span class="like-count"><?= (int) $img['like\_count'] ?></span></button>

|

<button class="dislike-btn" <?= $logged\_in ? '' : 'disabled' ?>>👎 <span class="dislike-count"><?= (int) $img['dislike\_count'] ?></span></button>

</div>

<?php if ($logged\_in && $img['user\_id'] == $user\_id): ?>

<a href="index.php?page=editor&id=<?= $img['id'] ?>" class="edit-btn">Редагувати</a>

<?php else: ?>

<button class="edit-btn" disabled>Редагувати</button>

<?php endif; ?>

</div>

</div>

<?php endforeach; ?>

</div>

<?php if ($total\_pages > 1): ?>

<div class="pagination">

<?php for ($i = 1; $i <= $total\_pages; $i++): ?>

<a href="index.php?page=gallery&page\_num=<?= $i ?>" <?= $i === $page ? 'style="font-weight:bold;background:#0056b3"' : '' ?>><?= $i ?></a>

<?php endfor; ?>

</div>

<?php endif; ?>

</div>

<!-- Глобальні змінні -->

<script>

const GALLERY\_LOGGED\_IN = <?= json\_encode($logged\_in) ?>;

const GALLERY\_USER\_ID = <?= json\_encode($user\_id) ?>;

</script>

<script src="/assets/js/gallery.js"></script>

</body>

</html>

home.php:

<!-- pages/home.php -->

<h2>Головна сторінка</h2>

<p>Вітаємо у додатку!</p>

image\_info.php:

<?php

require\_once \_\_DIR\_\_ . '/../config/database.php';

session\_start();

echo "<link rel='stylesheet' href='/assets/css/image\_info.css'>";

$image\_id = $\_GET['id'] ?? null;

if (!$image\_id || !is\_numeric($image\_id)) {

echo "<p>Невірний запит</p>";

return;

}

$pdo = new PDO("mysql:host=" . DB\_HOST . ";dbname=" . DB\_NAME . ";charset=utf8", DB\_USER, DB\_PASS, [

PDO::ATTR\_ERRMODE => PDO::ERRMODE\_EXCEPTION

]);

$stmt = $pdo->prepare("

SELECT images.\*, users.username,

(SELECT COUNT(\*) FROM comments WHERE image\_id = images.id) AS comment\_count,

(SELECT COUNT(\*) FROM likes WHERE image\_id = images.id AND like\_value = 1) AS like\_count,

(SELECT COUNT(\*) FROM likes WHERE image\_id = images.id AND like\_value = -1) AS dislike\_count

FROM images

JOIN users ON images.user\_id = users.id

WHERE images.id = :id

");

$stmt->execute([':id' => $image\_id]);

$image = $stmt->fetch(PDO::FETCH\_ASSOC);

if (!$image) {

echo "<p>Зображення не знайдено</p>";

return;

}

?>

<div class="image-detail-container">

<!-- Блок 1: Зображення -->

<img src="<?= htmlspecialchars($image['image\_path']) ?>" alt="">

<!-- Блок 2: Інформація -->

<div class="image-meta">

<p>Користувач: <?= htmlspecialchars($image['username']) ?></p>

<p>Дата: <?= htmlspecialchars($image['created\_at']) ?></p>

<p id="comment-count">Коментарі: <?= (int) $image['comment\_count'] ?></p>

<p>👍 <?= (int) $image['like\_count'] ?> | 👎 <?= (int) $image['dislike\_count'] ?></p>

</div>

<!-- Блок 3: Коментарі -->

<div class="comments" id="comments-container"></div>

<?php if (isset($\_SESSION['user\_id'])): ?>

<div class="comment-form">

<textarea id="comment-text" placeholder="Напишіть коментар..."></textarea>

<button id="submit-comment">Додати коментар</button>

</div>

<?php endif; ?>

<!-- Блок 4: Повернення до галереї -->

<a href="index.php?page=gallery" class="back-link">← Повернутися до галереї</a>

</div>

<script>

const imageId = <?= (int)$image\_id ?>;

</script>

<script src="/assets/js/image\_info.js"></script>

login.php:

<h2>Увійти</h2>

<?php

if (!empty($\_SESSION['message'])) {

echo '<div class="alert" style="color: green; padding: 10px; text-align: center;">' .

htmlspecialchars($\_SESSION['message']) .

'</div>';

unset($\_SESSION['message']);

}

?>

<form action="actions.php" method="post">

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

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

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

<label for="password">Пароль:</label>

<div class="password-container">

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

<span class="toggle-password" onclick="togglePassword('password')">

<svg width="20" height="20" viewBox="0 0 24 24" fill="none" xmlns="http://www.w3.org/2000/svg">

<path d="M12 5C7 5 3.1 8.3 1.5 12C3.1 15.7 7 19 12 19C17 19 20.9 15.7 22.5 12C20.9 8.3 17 5 12 5Z"

stroke="#007BFF" stroke-width="2" fill="none"/>

<circle cx="12" cy="12" r="3" stroke="#007BFF" stroke-width="2" fill="none"/>

</svg>

</span>

</div>

<button type="submit">Увійти</button>

</form>

<a href="index.php?page=register">Реєстрація</a> |

<a href="index.php?page=forgot\_password">Забули пароль?</a>

<?php if (isset($\_SESSION['message'])): ?>

<p style="color:red;"><?= $\_SESSION['message']; unset($\_SESSION['message']); ?></p>

<?php endif; ?>

<!-- Підключення зовнішнього JS -->

<script src="/assets/js/login.js"></script>

register.php:

<h2>Реєстрація</h2>

<form action="actions.php" method="post">

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

<label for="username">Ім'я:</label>

<input type="text" name="username" id="username" placeholder="Ваше ім'я" required>

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

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

<label for="password">Пароль:</label>

<div class="password-container">

<input type="password" name="password" id="password" placeholder="Введіть пароль" required>

<span class="toggle-password" onclick="togglePassword('password')">

<svg width="20" height="20" viewBox="0 0 24 24" fill="none" xmlns="http://www.w3.org/2000/svg">

<path d="M12 5C7 5 3.1 8.3 1.5 12C3.1 15.7 7 19 12 19C17 19 20.9 15.7 22.5 12C20.9 8.3 17 5 12 5Z"

stroke="#007BFF" stroke-width="2" fill="none"/>

<circle cx="12" cy="12" r="3" stroke="#007BFF" stroke-width="2" fill="none"/>

</svg>

</span>

</div>

<label for="confirm\_password">Повторіть пароль:</label>

<div class="password-container">

<input type="password" name="confirm\_password" id="confirm\_password" placeholder="Повторіть пароль" required>

<span class="toggle-password" onclick="togglePassword('confirm\_password')">

<svg width="20" height="20" viewBox="0 0 24 24" fill="none" xmlns="http://www.w3.org/2000/svg">

<path d="M12 5C7 5 3.1 8.3 1.5 12C3.1 15.7 7 19 12 19C17 19 20.9 15.7 22.5 12C20.9 8.3 17 5 12 5Z"

stroke="#007BFF" stroke-width="2" fill="none"/>

<circle cx="12" cy="12" r="3" stroke="#007BFF" stroke-width="2" fill="none"/>

</svg>

</span>

</div>

<div><button type="submit">Зареєструватися</button></div>

</form>

<div><a href="index.php?page=login">Вже є акаунт? Увійти</a></div>

<?php

if (isset($\_SESSION['message'])) {

echo "<p style='color:red;'>" . $\_SESSION['message'] . "</p>";

unset($\_SESSION['message']);

}

?>

<script src="/assets/js/login.js"></script>

reset\_password.php:

<h2>Скидання паролю</h2>

<form action="actions.php" method="post">

<input type="hidden" name="action" value="reset\_password">

<input type="hidden" name="token" value="<?= htmlspecialchars($\_GET['token'] ?? '') ?>">

<label for="password">Новий пароль:</label>

<div class="password-container">

<input type="password" name="password" id="password" placeholder="Введіть новий пароль" required>

<span class="toggle-password" onclick="togglePassword('password')">

<svg width="20" height="20" viewBox="0 0 24 24" fill="none" xmlns="http://www.w3.org/2000/svg">

<path d="M12 5C7 5 3.1 8.3 1.5 12C3.1 15.7 7 19 12 19C17 19 20.9 15.7 22.5 12C20.9 8.3 17 5 12 5Z"

stroke="#007BFF" stroke-width="2" fill="none"/>

<circle cx="12" cy="12" r="3" stroke="#007BFF" stroke-width="2" fill="none"/>

</svg>

</span>

</div>

<label for="confirm\_password">Повторіть пароль:</label>

<div class="password-container">

<input type="password" name="confirm\_password" id="confirm\_password" placeholder="Повторіть пароль" required>

<span class="toggle-password" onclick="togglePassword('confirm\_password')">

<svg width="20" height="20" viewBox="0 0 24 24" fill="none" xmlns="http://www.w3.org/2000/svg">

<path d="M12 5C7 5 3.1 8.3 1.5 12C3.1 15.7 7 19 12 19C17 19 20.9 15.7 22.5 12C20.9 8.3 17 5 12 5Z"

stroke="#007BFF" stroke-width="2" fill="none"/>

<circle cx="12" cy="12" r="3" stroke="#007BFF" stroke-width="2" fill="none"/>

</svg>

</span>

</div>

<div><button type="submit">Зберегти новий пароль</button></div>

</form>

<div><a href="index.php?page=login">Повернутися до входу</a></div>

<?php

if (isset($\_SESSION['message'])) {

echo "<p style='color:red;'>" . $\_SESSION['message'] . "</p>";

unset($\_SESSION['message']);

}

?>

<script src="/assets/js/login.js"></script>

noerrors.fun/js/

script.js:

const video = document.querySelector('video');

const canvas = document.querySelector('canvas');

const snapButton = document.querySelector('#snap');

// Доступ до камери

navigator.mediaDevices.getUserMedia({ video: true })

.then(stream => video.srcObject = stream)

.catch(err => console.error("Камера недоступна!", err));

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

const context = canvas.getContext('2d');

context.drawImage(video, 0, 0, canvas.width, canvas.height);

const photo = canvas.toDataURL('image/png');

fetch('save\_photo.php', {

method: 'POST',

body: JSON.stringify({ image: photo }),

headers: { 'Content-Type': 'application/json' }

})

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

.then(data => console.log(data))

.catch(err => console.error("Помилка збереження!", err));

});

noerrors.fun/includes/

aside.php:

<aside class="overlay-list">

<?php

$overlayDir = \_\_DIR\_\_ . '/../uploads/overlay\_images';

$overlayUrl = '/uploads/overlay\_images';

if (is\_dir($overlayDir)) {

$files = scandir($overlayDir);

foreach ($files as $file) {

$ext = strtolower(pathinfo($file, PATHINFO\_EXTENSION));

if (in\_array($ext, ['jpg', 'jpeg', 'png', 'gif'])) {

echo '<img src="' . $overlayUrl . '/' . htmlspecialchars($file) . '" alt="Overlay" onclick="addOverlay(this.src)">';

}

}

} else {

echo '<p>Немає зображень для накладання.</p>';

}

?>

</aside>

footer.php:

<footer>

<p>© 2025 Фоторедактор. Усі права захищено.</p>

</footer>

</body>

</html>

functions.php:

<?php

// Підключення до бази даних

function connectDB() {

require\_once \_\_DIR\_\_ . 'http://localhost/photo\_editing\_app/config/database.php';

try {

return new PDO("mysql:host=$db\_host;dbname=$db\_name", $db\_user, $db\_pass);

} catch (PDOException $e) {

die("Помилка підключення: " . $e->getMessage());

}

}

// Функція для перевірки, чи користувач авторизований

function isLoggedIn() {

return isset($\_SESSION['user\_id']);

}

// Функція для редиректу

function redirect($url) {

header("Location: $url");

exit();

}

// Функція для очищення даних від шкідливого коду

function sanitizeInput($data) {

return htmlspecialchars(trim($data));

}

?>

header.php:

<!DOCTYPE html>

<html lang="uk">

<head>

<meta charset="UTF-8">

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

<title>Фоторедактор</title>

<link rel="icon" href="favicon.ico">

<link rel="stylesheet" href="/assets/css/style.css">

</head>

<body>

<?php

require\_once 'config/check\_db.php';

$editorLink = "index.php?page=editor";

if (isset($\_SESSION['user\_id'])) {

$userId = (int)$\_SESSION['user\_id'];

$stmt = $pdo->prepare("SELECT photo\_user\_path FROM users WHERE id = ?");

$stmt->execute([$userId]);

$user = $stmt->fetch(PDO::FETCH\_ASSOC);

if ($user) {

$photoPath = $user['photo\_user\_path'];

$stmt2 = $pdo->prepare("SELECT id FROM images WHERE image\_path = ?");

$stmt2->execute([$photoPath]);

$image = $stmt2->fetch(PDO::FETCH\_ASSOC);

if ($image) {

$editorLink = "index.php?page=editor&id=" . (int)$image['id'];

} else {

$defaultPhotoPath = '/uploads/images/default\_photo\_user.png';

$stmt3 = $pdo->prepare("SELECT id FROM images WHERE image\_path = ?");

$stmt3->execute([$defaultPhotoPath]);

$defaultImage = $stmt3->fetch(PDO::FETCH\_ASSOC);

if ($defaultImage) {

$editorLink = "index.php?page=editor&id=" . (int)$defaultImage['id'];

}

}

}

}

?>

<header>

<nav>

<div class="burger" onclick="toggleMenu()">

<div></div>

<div></div>

<div></div>

</div>

<ul id="menu">

<?php if ($db\_exists): ?>

<li><a href="index.php">Головна</a></li>

<li><a href="index.php?page=gallery">Галерея</a></li>

<?php if (isset($\_SESSION['user\_id'])): ?>

<li><a href="<?= htmlspecialchars($editorLink) ?>">Редактор</a></li>

<li><a href="index.php?page=dashboard">Кабінет</a></li>

<li><a href="/actions/logout.php">Вийти</a></li>

<?php else: ?>

<li><a href="index.php?page=login">Увійти</a></li>

<?php endif; ?>

<?php else: ?>

<li><a href="setup.php">Встановити БД</a></li>

<?php endif; ?>

</ul>

</nav>

</header>

<!-- Підключення js скрипта -->

<script src="/assets/js/header.js"></script>

main.php:

<main>

<h1>Ласкаво просимо до нашого веб-додатку!</h1>

<p>Це стартова сторінка, де ви можете редагувати зображення та переглядати галерею.</p>

</main>

noerrors.fun/css/

styles.css:

/\* Основні стилі \*/

body {

font-family: 'Arial', sans-serif;

background-color: #f4f4f4;

color: #333;

margin: 0;

padding: 0;

box-sizing: border-box;

}

/\* Заголовок \*/

header {

background-color: #4CAF50;

color: white;

padding: 10px 0;

text-align: center;

font-size: 24px;

}

/\* Футер \*/

footer {

background-color: #ddd;

color: #555;

text-align: center;

padding: 10px 0;

position: fixed;

width: 100%;

bottom: 0;

}

/\* Контейнер логіну та реєстрації \*/

.login-container, .register-container {

width: 300px;

margin: 100px auto;

padding: 20px;

background-color: #fff;

box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

border-radius: 10px;

text-align: center;

}

/\* Поля вводу \*/

input[type="text"], input[type="password"] {

width: calc(100% - 20px);

padding: 10px;

margin: 10px 0;

border: 1px solid #ddd;

border-radius: 5px;

}

/\* Кнопки \*/

button {

width: 100%;

padding: 10px;

background-color: #4CAF50;

color: white;

border: none;

border-radius: 5px;

cursor: pointer;

transition: 0.3s;

}

button:hover {

background-color: #45a049;

}

/\* Повідомлення про помилки \*/

.error {

color: red;

font-size: 14px;

margin-bottom: 10px;

}

/\* Галерея \*/

.gallery-container {

display: grid;

grid-template-columns: repeat(auto-fill, minmax(200px, 1fr));

gap: 10px;

padding: 20px;

}

.gallery-item img {

width: 100%;

border-radius: 10px;

transition: 0.3s;

}

.gallery-item img:hover {

transform: scale(1.05);

}

/\* Редагування фото \*/

.edit-container {

width: 80%;

margin: 20px auto;

text-align: center;

}

#video, #canvas {

border: 1px solid #ddd;

border-radius: 10px;

margin-top: 10px;

}

noerrors.fun/config/

check\_db.php:

<?php

require\_once 'database.php';

// Перевірка, чи підключення до БД встановлено

if (!$pdo) {

$db\_exists = false;

} else {

// Перевіряємо, чи є необхідні таблиці

try {

$result = $pdo->query("SHOW TABLES LIKE 'users'");

$db\_exists = $result->rowCount() > 0;

} catch (PDOException $e) {

$db\_exists = false;

}

}

?>

database.php:

<?php

define('DB\_HOST', 'localhost');

define('DB\_NAME', 'photo\_edit\_app');

define('DB\_USER', 'root');

define('DB\_PASS', '');

try {

$pdo = new PDO("mysql:host=" . DB\_HOST . ";dbname=" . DB\_NAME, DB\_USER, DB\_PASS, [

PDO::ATTR\_ERRMODE => PDO::ERRMODE\_EXCEPTION,

PDO::ATTR\_DEFAULT\_FETCH\_MODE => PDO::FETCH\_ASSOC,

PDO::ATTR\_EMULATE\_PREPARES => false

]);

} catch (PDOException $e) {

$pdo = null;

}

?>

setup.php:

<?php

require\_once 'database.php';

try {

$pdo = new PDO($DB\_DSN, $DB\_USER, $DB\_PASSWORD);

$pdo->setAttribute(PDO::ATTR\_ERRMODE, PDO::ERRMODE\_EXCEPTION);

// Таблиця користувачів

$pdo->exec("CREATE TABLE IF NOT EXISTS users (

id INT PRIMARY KEY AUTO\_INCREMENT,

username VARCHAR(50) UNIQUE NOT NULL,

email VARCHAR(100) UNIQUE NOT NULL,

password VARCHAR(255) NOT NULL,

confirmed BOOLEAN DEFAULT 0,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP)");

// Таблиця для фото

$pdo->exec("CREATE TABLE IF NOT EXISTS photos (

id INT PRIMARY KEY AUTO\_INCREMENT,

user\_id INT NOT NULL,

image\_path VARCHAR(255) NOT NULL,

likes INT DEFAULT 0,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

FOREIGN KEY (user\_id) REFERENCES users(id) ON DELETE CASCADE)");

echo "База даних успішно створена!";

} catch (PDOException $e) {

die("Помилка створення БД: " . $e->getMessage());

}

?>

noerrors.fun/assets/css/

dashboard.css:

/\* Стилі для dashboard.php - Кабінет користувача \*/

.container-user-dashboard {

font-family: Arial, sans-serif;

display: flex;

justify-content: center;

align-items: baseline;

/\*height: 100vh;\*/

background-color: #f8f9fa;

margin: 0;

}

.user-dashboard {

max-width: 700px;

background: white;

padding: 20px;

border-radius: 10px;

box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

display: flex;

flex-direction: column;

align-items: center;

gap: 20px;

}

form#dashboard-form {

max-width: 700px;

}

.user-dashboard h2 {

text-align: center;

margin: 0;

}

#username-dashboard-hello {

font-weight: 600;

color: #0056b3;

font-style: italic;

}

.content {

display: flex;

gap: 20px;

width: 100%;

}

.photo-section {

width: 30%;

text-align: center;

}

.photo {

width: 100%;

height: 200px;

display: flex;

align-items: center;

justify-content: center;

border-radius: 5px;

background: #ccc;

}

.photo img {

max-width: 100%;

max-height: 100%;

border-radius: 5px;

}

.photo-buttons {

margin-top: 33px;

display: flex;

gap: 10px;

}

.photo-buttons button {

flex: 1;

padding: 8px;

border: none;

border-radius: 5px;

cursor: pointer;

background: #007bff;

color: white;

font-size: 14px;

transition: background 0.3s ease;

}

.photo-buttons button:hover {

background: #0056b3;

}

/\* Зміна фото-кнопок зверху \*/

.photo-buttons {

display: flex;

flex-direction: column;

align-items: stretch;

gap: 6px;

max-width: 100%;

}

/\* Сховати стандартний file input \*/

#photo-user-input {

display: none;

}

/\* Стиль кнопки як у .photo-buttons button \*/

.photo-upload-btn {

display: inline-block;

padding: 8px;

border: none;

border-radius: 5px;

cursor: pointer;

background: #007bff;

color: white;

font-size: 14px;

text-align: center;

transition: background 0.3s ease;

}

.photo-upload-btn:hover {

background: #0056b3;

}

/\* Виведення назви файлу \*/

.file-name-display {

font-size: 13px;

color: #333;

white-space: nowrap;

overflow: hidden;

text-overflow: ellipsis;

max-width: 100%;

}

/\* END зміна фото-кнопок зверху \*/

.info-section {

width: auto;

}

.info-item {

display: flex;

justify-content: space-between;

padding: 10px 0;

border-bottom: 1px solid #ddd;

font-size: 14px;

}

.info-item input {

width: 100%;

padding: 5px;

border: 1px solid #ccc;

border-radius: 4px;

background-color: #e9ecef;

font-size: 14px;

}

.info-item label {

font-size: 16px;

}

.actions {

margin-top: 20px;

text-align: center;

}

.actions a, .actions button {

display: inline-block;

padding: 10px 20px;

margin: 0 5px;

border: none;

border-radius: 5px;

cursor: pointer;

text-decoration: none;

color: white;

font-size: 14px;

transition: background 0.3s ease, filter 0.3s ease;

}

.actions a:hover, .actions button:hover {

filter: brightness(90%);

}

.change-password {

background: #ffc107;

}

.save {

background: #28a745;

}

.cancel {

background: #dc3545;

}

.save {

background-color: #28a745;

color: white;

}

.save:hover {

background-color: #1e7e34;

}

.cancel {

background-color: #6c757d;

color: white;

margin-left: 10px;

}

.cancel:hover {

background-color: #545b62;

}

input[type="checkbox"] {

width: 18px;

height: 18px;

accent-color: #007BFF; /\* Синій колір, як у Bootstrap або подібний \*/

cursor: pointer;

transition: all 0.2s ease-in-out;

}

/\* Панель вибору зображень з серверу \*/

/\* Забезпечує, щоб блок був у новому рядку \*/

.full-width {

width: 100%;

display: block;

margin-top: 20px;

}

/\* Щоб панель була повністю нижче форми \*/

.image-server-selector {

margin-top: 10px;

width: 100%;

}

#server-image-panel {

margin-top: 10px;

display: none;

flex-wrap: wrap;

gap: 10px;

border: 1px solid #ccc;

padding: 10px;

}

#server-image-panel.open {

display: flex; /\* Показати панель \*/

/\*max-height: 500px; /\* Максимальна висота \*/

padding: 10px;

}

#server-image-panel img {

width: 100px;

height: 100px;

object-fit: cover;

cursor: pointer;

border: 2px solid transparent;

}

#server-image-panel img.selected {

border: 2px solid #007bff;

}

#server-image-panel img:hover {

border-color: #007bff;

}

/\* END Панель вибору зображень з серверу \*/

/\* Адаптація Кабінету користувача для мобільних пристроїв \*/

/\* Кабінет користувача \*/

@media (max-width: 768px) {

.content {

flex-direction: column;

align-items: center;

gap: 10px;

}

.photo-section, .info-section {

width: 100%;

}

.photo {

height: 150px;

}

.photo-buttons {

flex-direction: column;

align-items: center;

width: 100%;

}

.photo-buttons button {

width: 80%;

}

.info-item {

flex-direction: column;

text-align: center;

}

form input[type="text"],

form input[type="email"],

form input[type="password"] {

width: 90%;

}

.actions {

display: flex;

flex-direction: column;

gap: 10px;

width: 100%;

}

.actions a, .actions button {

width: auto;

}

}

editor.css:

.editor-container {

display: flex;

gap: 20px;

justify-content: center;

align-items: flex-start;

padding: 20px;

}

.main-editor {

width: 700px;

text-align: center;

}

.image-wrapper {

border: 2px solid #ccc;

display: inline-block;

}

.editor-buttons {

margin-top: 10px;

display: flex;

gap: 10px;

justify-content: center;

}

.editor-buttons button {

padding: 10px 20px;

font-size: 16px;

background-color: #4CAF50;

color: white;

border: none;

border-radius: 5px;

cursor: pointer;

transition: background-color 0.3s;

}

.editor-buttons button:hover {

background-color: #45a049;

}

/\* Кнопка "Зберегти" \*/

.editor-buttons button.save-button {

background-color: #007bff; /\* синій \*/

color: white;

}

/\* "Зберегти" при наведенні \*/

.editor-buttons button.save-button:hover:not(:disabled) {

background-color: #0056b3; /\* темніший синій \*/

}

/\* Кнопка "Відмінити" \*/

.editor-buttons button.reset-button {

background-color: #6c757d; /\* темно-сірий \*/

color: white;

}

/\* "Відмінити" при наведенні \*/

.editor-buttons button.reset-button:hover {

background-color: #5a6268; /\* темніший сірий \*/

}

/\* Disabled кнопка \*/

.editor-buttons button:disabled {

background-color: #b0c4de; /\* світло-синій \*/

cursor: not-allowed;

opacity: 0.6;

}

.overlay-list {

width: 150px;

display: flex;

flex-direction: column;

gap: 10px;

}

.overlay-list img {

width: 100%;

cursor: pointer;

border: 1px solid #aaa;

transition: 0.2s;

}

.overlay-list img:hover {

border-color: #333;

}

/\* Мобільна адаптація \*/

@media (max-width: 768px) {

.editor-container {

flex-direction: column;

align-items: center;

padding: 10px;

}

.main-editor {

width: 100%;

}

.editor-buttons {

flex-direction: column;

align-items: stretch;

}

.editor-buttons button {

width: 100%;

}

.overlay-list {

width: 100%;

flex-direction: row;

flex-wrap: wrap;

justify-content: center;

}

.overlay-list img {

width: 30%;

min-width: 80px;

}

canvas {

width: 320px;

}

}

gallery.css:

/\* Стилі для gallery.php \*/

.gallery-container { max-width: 700px; margin: auto; display: flex; flex-direction: column; gap: 20px; align-items: center; }

.camera-block { background: #fff; border-radius: 10px; padding: 15px 0; width: 100%; position: relative; }

.camera-block video { width: 100%; border-radius: 10px; }

.no-camera { background: rgba(128, 128, 128, 0.3); padding: 20px; text-align: center; font-size: 18px; border-radius: 10px; }

#screenshot-btn { margin-top: 10px; padding: 8px 16px; font-size: 14px; border: none; border-radius: 8px; background: #007bff; color: white; cursor: pointer; }

#screenshot-btn:disabled { background: #aaa; cursor: not-allowed; }

.gallery-grid { display: flex; flex-wrap: wrap; gap: 10px; justify-content: space-between; width: 100%; }

.image-card { background: white; border: 1px solid transparent; border-radius: 8px; overflow: hidden; width: calc(33.33% - 10px); position: relative; box-shadow: 0 0 5px rgba(0,0,0,0.1); transition: border 0.1s; }

.image-card:hover { border: 1px solid #007bff; }

.image-card img { width: 100%; display: block; }

.image-info { padding: 5px 10px; font-size: 12px; color: #555; }

.edit-btn { margin-top: 5px; display: inline-block; background: #28a745; color: white; padding: 4px 8px; font-size: 12px; border-radius: 4px; text-decoration: none; }

.edit-btn[disabled] { background: #ccc; color: #666; cursor: not-allowed; text-decoration: none; }

.pagination { text-align: center; margin-top: 20px; }

.pagination a { margin: 0 5px; text-decoration: none; padding: 6px 10px; background: #007bff; color: white; border-radius: 4px; }

.like-btn.voted, .dislike-btn.voted { background-color: #007bff; color: white; }

@media (max-width: 768px) {

.image-card { width: calc(50% - 10px); }

.camera-block { max-width: 100%; }

}

image\_info.css:

/\* Стилі для image\_info.php \*/

.image-detail-container {

max-width: 700px;

margin: 20px auto;

padding: 20px;

background: #f4f4f4;

border-radius: 10px;

box-shadow: 0 0 8px rgba(0,0,0,0.1);

}

.image-detail-container img {

display: block;

max-width: 100%;

height: auto;

margin: 0 auto;

border-radius: 5px;

}

.image-meta {

margin-top: 15px;

padding: 10px;

background: #fff;

border-radius: 5px;

}

.comments {

margin-top: 20px;

text-align: left;

}

.comment {

padding: 8px;

background: #fff;

margin-bottom: 10px;

border-left: 4px solid #007BFF;

border-radius: 5px;

}

.comment-form {

margin-top: 15px;

background: #fff;

padding: 10px;

border-radius: 5px;

box-shadow: 0 0 4px rgba(0,0,0,0.1);

}

.comment-form textarea {

width: 100%;

max-width: 100%;

box-sizing: border-box;

height: 80px;

resize: vertical;

padding: 8px;

border-radius: 4px;

border: 1px solid #ccc;

font-size: 14px;

font-family: inherit;

}

.comment-form button,

.back-link {

display: inline-block;

padding: 8px 16px;

color: #fff;

border: none;

border-radius: 5px;

font-size: 14px;

text-decoration: none;

cursor: pointer;

transition: background-color 0.3s ease, box-shadow 0.2s ease;

margin-top: 10px;

}

.comment-form button {

background-color: #28a745; /\* зелена кнопка для "Додати коментар" \*/

}

.comment-form button:hover {

background-color: #218838;

}

.back-link {

background-color: #6c757d;

text-align: center;

}

.back-link:hover {

background-color: #5a6268;

box-shadow: 0 2px 6px rgba(0,0,0,0.2);

}

style.ss:

body {

font-family: Arial, sans-serif;

margin: 0;

padding: 0;

display: flex;

flex-direction: column;

min-height: 100vh;

/\*background-color: yellowgreen;\*/

}

/\* Верхній колонтитул \*/

header {

background-color: #007bff;

color: white;

padding: 15px;

text-align: center;

}

nav ul {

list-style: none;

padding: 0;

flex-direction: row;

align-content: center;

justify-content: center;

align-items: center;

flex-wrap: wrap;

}

nav ul li {

display: inline;

/\*margin-right: 15px;\*/

padding: 0 30px;

}

nav ul li a {

color: white;

text-decoration: none;

text-transform: uppercase;

}

nav ul li a:hover {

color: #ffcc00;

}

/\* Бургер-меню \*/

.burger {

display: none;

flex-direction: column;

justify-content: space-between;

width: 35px;

height: 30px;

position: absolute;

right: 20px;

top: 20px;

cursor: pointer;

}

.burger div {

width: 100%;

height: 4px;

background-color: white;

border-radius: 2px;

}

/\* Основний контент \*/

main {

flex-grow: 1;

padding: 20px;

background-color: #f8f9fa;

text-align: center;

}

/\* Нижній колонтитул \*/

footer {

background-color: #343a40;

color: white;

text-align: center;

padding: 10px;

}

/\* Форми \*/

/\* Загальні стилі для форми \*/

form {

max-width: 400px;

margin: 20px auto;

padding: 20px;

background: #f9f9f9;

border-radius: 10px;

box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

display: flex;

flex-direction: column;

}

/\* Стилі для міток \*/

form label {

font-weight: bold;

margin-top: 10px;

}

/\* Поля вводу \*/

form input[type="text"],

form input[type="email"],

form input[type="password"] {

width: auto;

padding: 10px;

border: 1px solid #ccc;

border-radius: 5px;

}

#password, #confirm\_password {

min-width: 100%;

}

/\*Вставка \*/

.password-container {

position: relative;

display: inline-block;

}

.password-container input {

padding-right: 30px; /\* Робимо місце для іконки всередині поля \*/

width: 100%;

box-sizing: border-box;

height: 40px;

}

.password-container .toggle-password {

position: absolute;

right: 10px;

top: 50%;

transform: translateY(-50%);

cursor: pointer;

user-select: none;

display: flex;

align-items: center;

justify-content: center;

width: 24px;

height: 24px;

}

/\* Кнопка \*/

form button {

margin-top: 15px;

padding: 10px;

background: #007bff;

color: white;

border: none;

border-radius: 5px;

cursor: pointer;

font-size: 16px;

}

form button:hover {

background: #0056b3;

}

/\* Повідомлення про помилки \*/

.error-message {

color: red;

font-weight: bold;

text-align: center;

margin-top: 10px;

}

/\* Повідомлення про успішні дії \*/

.success-message {

color: green;

font-weight: bold;

text-align: center;

margin-top: 10px;

}

/\* Посилання \*/

form a {

text-align: center;

display: block;

margin-top: 10px;

}

/\* Бургер меню \*/

@media (max-width: 768px) {

header {

padding: 36px;

}

nav ul {

display: none;

flex-direction: column;

position: fixed;

top: 64px;

left: 5px;

right: 5px;

bottom: 64px;

background: rgba(0, 86, 179, 0.84);

box-shadow: 0 0 10px rgba(0, 0, 0, 0.2);

border-radius: 5px;

overflow: hidden;

padding: 0;

align-items: center;

justify-content: center;

margin-left: 5px;

margin-right: 5px;

z-index: 100;

}

nav ul.active {

display: flex;

}

nav ul li {

padding: 25px;

text-align: center;

width: 100%;

}

nav ul li a {

display: block;

padding: 10px;

font-size: 30px;

color: white;

transition: 0.3s;

text-transform: uppercase;

}

nav ul li a:hover, nav ul li a:focus {

background: #004494;

border-radius: 5px;

}

.burger {

display: flex;

}

}

/\*\*/

noerrors.fun/assets/js/

dashboard.js:

// Ініціалізація відображення назви вибраного файлу

function initFileNameDisplay(fileInputId, fileNameDivId, defaultText = 'Файл не вибрано') {

const fileInput = document.getElementById(fileInputId);

const fileNameDiv = document.getElementById(fileNameDivId);

if (!fileInput || !fileNameDiv) return;

fileInput.addEventListener('change', function () {

if (fileInput.files.length > 0) {

const name = fileInput.files[0].name;

fileNameDiv.textContent = name;

fileNameDiv.title = name;

} else {

fileNameDiv.textContent = defaultText;

fileNameDiv.title = '';

}

});

}

// Ініціалізація прев'ю фото при виборі з комп'ютера

function initUserPhotoPreview(fileInputId, previewImgId, fileNameDivId, defaultPhotoPath) {

const fileInput = document.getElementById(fileInputId);

const previewImg = document.getElementById(previewImgId);

const fileNameDiv = document.getElementById(fileNameDivId);

if (!fileInput || !previewImg || !fileNameDiv) return;

fileInput.addEventListener('change', function () {

const file = this.files[0];

if (file) {

const reader = new FileReader();

reader.onload = function (e) {

previewImg.src = e.target.result;

};

reader.readAsDataURL(file);

fileNameDiv.textContent = file.name;

fileNameDiv.title = file.name;

} else {

previewImg.src = defaultPhotoPath;

fileNameDiv.textContent = 'Файл не вибрано';

fileNameDiv.title = '';

}

});

}

// Ініціалізація обнулення фото при натисканні кнопки "Скинути"

function initResetUserPhoto(formId, fileInputId, previewImgId, fileNameDivId, defaultPhotoPath = '/uploads/images/default\_photo\_user.png') {

const form = document.getElementById(formId);

if (!form) return;

form.addEventListener('reset', function () {

const fileInput = document.getElementById(fileInputId);

const previewImg = document.getElementById(previewImgId);

const fileNameDiv = document.getElementById(fileNameDivId);

if (fileInput) fileInput.value = "";

if (previewImg) {

const defaultSrc = previewImg.getAttribute('data-default');

previewImg.src = defaultSrc || defaultPhotoPath;

}

if (fileNameDiv) {

fileNameDiv.textContent = 'Файл не вибрано';

fileNameDiv.title = '';

}

});

}

// Ініціалізація взаємодії з панеллю серверних зображень

function initServerImagePanel() {

const toggleButton = document.getElementById('toggle-image-panel');

const imagePanel = document.getElementById('server-image-panel');

if (toggleButton && imagePanel) {

toggleButton.addEventListener('click', function () {

imagePanel.classList.toggle('open');

});

}

const serverImages = document.querySelectorAll('.selectable-server-image');

const previewImg = document.getElementById('user-photo-preview');

const photoPathInput = document.getElementById('photo\_user\_path');

const selectedPhotoInput = document.getElementById('selected\_server\_photo');

const fileInput = document.getElementById('photo-user-input');

const fileNameDiv = document.getElementById('selected-file-name');

serverImages.forEach(img => {

img.addEventListener('click', function () {

serverImages.forEach(i => i.classList.remove('selected'));

this.classList.add('selected');

const path = this.getAttribute('data-path');

if (previewImg) previewImg.src = path;

if (photoPathInput) photoPathInput.value = path;

if (selectedPhotoInput) selectedPhotoInput.value = path;

if (fileInput) fileInput.value = "";

if (fileNameDiv) {

fileNameDiv.textContent = path;

fileNameDiv.title = path;

}

});

});

}

// Виклик ініціалізаторів після завантаження DOM

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

initFileNameDisplay('photo-user-input', 'selected-file-name');

initUserPhotoPreview(

'photo-user-input',

'user-photo-preview',

'selected-file-name',

'/uploads/images/default\_photo\_user.png'

);

initResetUserPhoto('dashboard-form', 'photo-user-input', 'user-photo-preview', 'selected-file-name');

initServerImagePanel();

});

editor.js:

let baseImage;

let hasChanges = false;

function initEditor() {

const canvas = document.getElementById('editor-canvas');

const ctx = canvas.getContext('2d');

const container = document.querySelector('.editor-container');

const imagePath = container.dataset.imagePath;

const imageId = container.dataset.imageId || null;

// Завантаження базового зображення

baseImage = new Image();

baseImage.src = imagePath;

baseImage.onload = function () {

resizeCanvasToImage(baseImage);

drawBaseImage();

};

// Зберігаємо ID і шлях у глобальні змінні для інших функцій

window.editorImagePath = imagePath;

window.editorImageId = imageId;

updateSaveButtonState();

}

function resizeCanvasToImage(image) {

const canvas = document.getElementById('editor-canvas');

const maxWidth = 640;

const maxHeight = 480;

const aspectRatio = image.width / image.height;

let newWidth = image.width;

let newHeight = image.height;

if (newWidth > maxWidth) {

newWidth = maxWidth;

newHeight = newWidth / aspectRatio;

}

if (newHeight > maxHeight) {

newHeight = maxHeight;

newWidth = newHeight \* aspectRatio;

}

canvas.width = Math.round(newWidth);

canvas.height = Math.round(newHeight);

}

function drawBaseImage() {

const canvas = document.getElementById('editor-canvas');

const ctx = canvas.getContext('2d');

ctx.clearRect(0, 0, canvas.width, canvas.height);

ctx.drawImage(baseImage, 0, 0, canvas.width, canvas.height);

}

function addOverlay(src) {

const canvas = document.getElementById('editor-canvas');

const ctx = canvas.getContext('2d');

const overlay = new Image();

overlay.src = src;

overlay.onload = function () {

const width = canvas.width \* 0.2;

const height = canvas.height \* 0.2;

ctx.drawImage(overlay, canvas.width - width - 10, canvas.height - height - 10, width, height);

hasChanges = true;

updateSaveButtonState();

};

}

function saveEditedImage() {

if (!hasChanges) {

alert('Немає змін для збереження.');

return;

}

const canvas = document.getElementById('editor-canvas');

const dataURL = canvas.toDataURL('image/jpeg');

fetch('/actions/save\_edited\_image.php', {

method: 'POST',

headers: { 'Content-Type': 'application/json' },

body: JSON.stringify({

image: dataURL,

original\_path: window.editorImagePath,

image\_id: window.editorImageId

})

})

.then(res => res.json())

.then(data => {

if (data.success) {

alert('Зображення збережено успішно!');

hasChanges = false;

updateSaveButtonState();

} else {

alert('Помилка збереження: ' + data.error);

}

})

.catch(err => {

console.error(err);

alert('Щось пішло не так...');

});

}

function resetOverlays() {

drawBaseImage();

hasChanges = false;

updateSaveButtonState();

}

function updateSaveButtonState() {

const saveButton = document.querySelector('.editor-buttons button[onclick="saveEditedImage()"]');

if (saveButton) {

saveButton.disabled = !hasChanges;

}

}

document.addEventListener('DOMContentLoaded', initEditor);

gallery.js:

document.addEventListener('DOMContentLoaded', () => {

// Обробка лайків/дизлайків

document.querySelectorAll('.like-section').forEach(section => {

const imageId = section.dataset.imageId;

const likeBtn = section.querySelector('.like-btn');

const dislikeBtn = section.querySelector('.dislike-btn');

const likeCountSpan = section.querySelector('.like-count');

const dislikeCountSpan = section.querySelector('.dislike-count');

let userVote = null;

function updateButtonStyles() {

likeBtn.classList.toggle('voted', userVote === 1);

dislikeBtn.classList.toggle('voted', userVote === -1);

}

function vote(value) {

fetch('/actions/like\_handler.php', {

method: 'POST',

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

body: new URLSearchParams({

image\_id: imageId,

like\_value: value

})

})

.then(res => res.json())

.then(data => {

if (data.error) {

alert(data.error);

} else {

likeCountSpan.textContent = data.like\_count;

dislikeCountSpan.textContent = data.dislike\_count;

userVote = value;

updateButtonStyles();

}

})

.catch(err => console.error('Fetch error:', err));

}

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

if (userVote === 1) {

userVote = null;

likeCountSpan.textContent = parseInt(likeCountSpan.textContent) - 1;

} else {

vote(1);

}

updateButtonStyles();

});

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

if (userVote === -1) {

userVote = null;

dislikeCountSpan.textContent = parseInt(dislikeCountSpan.textContent) - 1;

} else {

vote(-1);

}

updateButtonStyles();

});

});

// Камера і скріншоти

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

const noCamera = document.getElementById('no-camera');

const screenshotBtn = document.getElementById('screenshot-btn');

if (video && screenshotBtn) {

navigator.mediaDevices.getUserMedia({ video: true })

.then(stream => {

video.srcObject = stream;

screenshotBtn.disabled = false;

})

.catch(() => {

if (noCamera) noCamera.style.display = 'block';

});

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

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

canvas.width = video.videoWidth;

canvas.height = video.videoHeight;

const ctx = canvas.getContext('2d');

ctx.drawImage(video, 0, 0, canvas.width, canvas.height);

const dataURL = canvas.toDataURL('image/jpeg');

fetch('/actions/save\_screenshot.php', {

method: 'POST',

headers: { 'Content-Type': 'application/json' },

body: JSON.stringify({ image: dataURL }),

credentials: 'same-origin'

})

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

.then(data => {

if (data.success) {

alert('Скріншот збережено!');

location.reload();

} else {

alert('Помилка: ' + data.error);

}

})

.catch(error => console.error('Fetch error:', error));

});

}

});

header.js:

// Перемикає клас .active для меню при кліку на бургер

function toggleMenu() {

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

menu.classList.toggle('active');

}

// Закриває меню при кліку поза ним

document.addEventListener('click', function(event) {

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

const burger = document.querySelector('.burger');

if (!menu.contains(event.target) && !burger.contains(event.target)) {

menu.classList.remove('active');

}

});

// Закриває меню при натисканні клавіші Escape

document.addEventListener('keydown', function(event) {

if (event.key === 'Escape') {

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

menu.classList.remove('active');

}

});

image\_info.js:

document.addEventListener("DOMContentLoaded", () => {

// Завантажуємо всі коментарі

function loadComments() {

fetch(`/actions/get\_comments.php?image\_id=${imageId}`)

.then(res => res.json())

.then(data => {

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

container.innerHTML = '';

data.forEach(comment => {

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

div.className = 'comment';

// Форматуємо дату коментаря

const createdAt = new Date(comment.created\_at);

const formattedDate = createdAt.toLocaleString('uk-UA', {

year: 'numeric',

month: '2-digit',

day: '2-digit',

hour: '2-digit',

minute: '2-digit',

second: '2-digit'

});

div.innerHTML = `<strong>${comment.username} (${formattedDate}):</strong> ${comment.comment}`;

container.appendChild(div);

});

});

}

// Оновлюємо лічильник коментарів

function updateCommentCount(imageId) {

fetch(`/actions/get\_comment\_count.php?image\_id=${imageId}`)

.then(res => res.json())

.then(data => {

if (data.count !== undefined) {

const commentCountText = document.getElementById('comment-count');

commentCountText.textContent = `Коментарі: ${data.count}`;

}

})

.catch(err => console.error('Помилка при оновленні лічильника коментарів:', err));

}

// Ініціалізація

loadComments();

// Обробка надсилання коментаря

const submitBtn = document.getElementById('submit-comment');

if (submitBtn) {

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

const textEl = document.getElementById('comment-text');

const text = textEl.value.trim();

if (!text) return;

fetch('/actions/add\_comment.php', {

method: 'POST',

headers: {'Content-Type': 'application/json'},

body: JSON.stringify({image\_id: imageId, comment: text})

})

.then(res => res.json())

.then(data => {

if (data.success) {

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

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

div.className = 'comment';

const datetime = new Date(data.created\_at);

const formatted = `${datetime.toLocaleDateString()} ${datetime.toLocaleTimeString([], {hour: '2-digit', minute: '2-digit'})}`;

div.innerHTML = `<strong>${data.username}</strong> (${formatted}): ${text}`;

container.appendChild(div);

textEl.value = '';

updateCommentCount(imageId);

} else {

alert('Помилка: ' + data.error);

}

});

});

}

});

login.js:

// Функція для перемикання видимості пароля

function togglePassword(fieldId) {

const passwordField = document.getElementById(fieldId);

if (!passwordField) return;

// Перемикає тип поля між "password" та "text"

passwordField.type = passwordField.type === "password" ? "text" : "password";

}

/noerrors.fun/actions/

add\_comment.php:

<?php

session\_start();

require\_once \_\_DIR\_\_ . '/../config/database.php';

header('Content-Type: application/json; charset=utf-8');

// Перевірка авторизації

if (!isset($\_SESSION['user\_id'])) {

echo json\_encode(['success' => false, 'error' => 'Не авторизовано']);

exit;

}

$data = json\_decode(file\_get\_contents('php://input'), true);

$image\_id = $data['image\_id'] ?? null;

$comment = trim($data['comment'] ?? '');

if (!$image\_id || !$comment) {

echo json\_encode(['success' => false, 'error' => 'Невірні дані']);

exit;

}

$pdo = new PDO("mysql:host=" . DB\_HOST . ";dbname=" . DB\_NAME . ";charset=utf8", DB\_USER, DB\_PASS);

// Додавання коментаря

$stmt = $pdo->prepare("INSERT INTO comments (image\_id, user\_id, comment, created\_at) VALUES (:image\_id, :user\_id, :comment, NOW())");

$stmt->execute([

':image\_id' => $image\_id,

':user\_id' => $\_SESSION['user\_id'],

':comment' => $comment

]);

// Отримуємо username коментатора

$stmt = $pdo->prepare("SELECT username FROM users WHERE id = :id LIMIT 1");

$stmt->execute([':id' => $\_SESSION['user\_id']]);

$commenter\_username = $stmt->fetchColumn();

// Отримуємо дані про власника зображення

$stmt = $pdo->prepare("

SELECT u.id AS owner\_id, u.email, u.username AS owner\_username, u.notify\_new\_comment

FROM images i

JOIN users u ON i.user\_id = u.id

WHERE i.id = :image\_id

LIMIT 1

");

$stmt->execute([':image\_id' => $image\_id]);

$image\_owner = $stmt->fetch(PDO::FETCH\_ASSOC);

if ($image\_owner) {

$owner\_id = (int)$image\_owner['owner\_id'];

$owner\_email = $image\_owner['email'];

$owner\_username = $image\_owner['owner\_username'];

$notify = (int)$image\_owner['notify\_new\_comment'];

$commenter\_id = (int)$\_SESSION['user\_id'];

// Не надсилати, якщо власник = коментатор або notify = 0

if ($owner\_id !== $commenter\_id && $notify === 1) {

$comment\_time = date('Y-m-d H:i:s');

$image\_link = $\_SERVER['REQUEST\_SCHEME'] . '://' . $\_SERVER['HTTP\_HOST'] . "/index.php?page=image\_info&id=" . $image\_id;

$subject = "Новий коментар до вашого зображення";

$message = "Привіт, $owner\_username!\n\n"

. "Користувач \"$commenter\_username\" залишив новий коментар до вашого зображення.\n\n"

. "Дата та час: $comment\_time\n"

. "Коментар:\n\"$comment\"\n\n"

. "Переглянути зображення: $image\_link\n\n"

. "З повагою,\nКоманда сайту Фоторедактор";

$headers = "From: no-reply@" . $\_SERVER['HTTP\_HOST'] . "\r\n" .

"Content-Type: text/plain; charset=UTF-8\r\n";

// Надсилання листа

$mail\_result = mail(

$owner\_email,

"=?UTF-8?B?" . base64\_encode($subject) . "?=",

$message,

$headers

);

}

}

// Відповідь

echo json\_encode([

'success' => true,

'username' => $commenter\_username,

'comment' => $comment,

'created\_at' => date('Y-m-d H:i:s')

], JSON\_UNESCAPED\_UNICODE);

get\_comment\_count.php:

<?php

require\_once \_\_DIR\_\_ . '/../config/database.php';

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

$image\_id = $\_GET['image\_id'] ?? null;

if (!$image\_id || !is\_numeric($image\_id)) {

echo json\_encode(['error' => 'Invalid image\_id']);

exit;

}

try {

$pdo = new PDO("mysql:host=" . DB\_HOST . ";dbname=" . DB\_NAME . ";charset=utf8", DB\_USER, DB\_PASS, [

PDO::ATTR\_ERRMODE => PDO::ERRMODE\_EXCEPTION

]);

$stmt = $pdo->prepare("SELECT COUNT(\*) AS count FROM comments WHERE image\_id = ?");

$stmt->execute([$image\_id]);

$row = $stmt->fetch(PDO::FETCH\_ASSOC);

echo json\_encode(['count' => (int)$row['count']]);

} catch (Exception $e) {

echo json\_encode(['error' => 'Database error']);

}

get\_comments.php:

<?php

require\_once \_\_DIR\_\_ . '/../config/database.php';

$image\_id = $\_GET['image\_id'] ?? null;

if (!$image\_id || !is\_numeric($image\_id)) {

echo json\_encode([]);

exit;

}

$pdo = new PDO(

"mysql:host=" . DB\_HOST . ";dbname=" . DB\_NAME . ";charset=utf8",

DB\_USER,

DB\_PASS,

[PDO::ATTR\_ERRMODE => PDO::ERRMODE\_EXCEPTION]

);

// Додаємо поле created\_at до вибірки

$stmt = $pdo->prepare("

SELECT c.comment, c.created\_at, u.username

FROM comments c

JOIN users u ON c.user\_id = u.id

WHERE c.image\_id = :id

ORDER BY c.created\_at ASC

");

$stmt->execute([':id' => $image\_id]);

// Повертаємо масив у JSON-форматі

echo json\_encode($stmt->fetchAll(PDO::FETCH\_ASSOC), JSON\_UNESCAPED\_UNICODE);

like\_handler.php:

<?php

session\_start();

require\_once \_\_DIR\_\_ . '/../config/database.php';

if (!isset($\_SESSION['user\_id'], $\_POST['image\_id'], $\_POST['like\_value'])) {

http\_response\_code(400);

echo json\_encode(['error' => 'Недостатньо даних']);

exit;

}

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

$image\_id = (int)$\_POST['image\_id'];

$like\_value = (int)$\_POST['like\_value'];

if (!in\_array($like\_value, [1, -1])) {

http\_response\_code(400);

echo json\_encode(['error' => 'Некоректне значення']);

exit;

}

$pdo = new PDO("mysql:host=" . DB\_HOST . ";dbname=" . DB\_NAME . ";charset=utf8", DB\_USER, DB\_PASS, [

PDO::ATTR\_ERRMODE => PDO::ERRMODE\_EXCEPTION

]);

// Перевірка, чи вже є лайк/дизлайк від користувача

$stmt = $pdo->prepare("SELECT \* FROM likes WHERE image\_id = ? AND user\_id = ?");

$stmt->execute([$image\_id, $user\_id]);

$existing = $stmt->fetch();

if ($existing) {

http\_response\_code(409); // Код помилки HTTP 409 Conflict

echo json\_encode(['error' => 'Вже голосував']);

exit;

}

// Додаємо запис

$stmt = $pdo->prepare("INSERT INTO likes (image\_id, user\_id, like\_value) VALUES (?, ?, ?)");

$stmt->execute([$image\_id, $user\_id, $like\_value]);

// Повертаємо нові значення

$stmt = $pdo->prepare("SELECT

SUM(CASE WHEN like\_value = 1 THEN 1 ELSE 0 END) AS like\_count,

SUM(CASE WHEN like\_value = -1 THEN 1 ELSE 0 END) AS dislike\_count

FROM likes WHERE image\_id = ?");

$stmt->execute([$image\_id]);

$counts = $stmt->fetch(PDO::FETCH\_ASSOC);

echo json\_encode([

'like\_count' => (int) $counts['like\_count'],

'dislike\_count' => (int) $counts['dislike\_count']

]);

logout.php:

<?php

session\_start();

session\_unset(); // Видаляє всі змінні сесії

session\_destroy(); // Завершує сесію

// Перенаправлення на головну сторінку після виходу

header("Location: /index.php");

exit;

?>

save\_edited\_image.php:

<?php

/\*

// Увімкнути помилки для дебагу

ini\_set('display\_errors', 1);

ini\_set('display\_startup\_errors', 1);

error\_reporting(E\_ALL);

\*/

// Перевірка сесії

session\_start();

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

if (!isset($\_SESSION['user\_id'])) {

echo json\_encode(['success' => false, 'error' => 'Користувач не авторизований']);

exit;

}

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

// Отримання JSON-запиту

$rawData = file\_get\_contents("php://input");

$data = json\_decode($rawData, true);

// Перевірка вхідних даних

if (

!$data ||

!isset($data['image']) ||

!preg\_match('/^data:image\/jpeg;base64,/', $data['image']) ||

!isset($data['image\_id'])

) {

echo json\_encode(['success' => false, 'error' => 'Невірні вхідні дані']);

exit;

}

$imageId = (int)$data['image\_id'];

$base64Image = str\_replace('data:image/jpeg;base64,', '', $data['image']);

$imageData = base64\_decode($base64Image);

if ($imageData === false) {

echo json\_encode(['success' => false, 'error' => 'Помилка декодування зображення']);

exit;

}

// Підключення до БД

require\_once \_\_DIR\_\_ . '/../config/database.php';

// Отримання імені користувача

$stmt = $pdo->prepare("SELECT username FROM users WHERE id = ?");

$stmt->execute([$userId]);

$user = $stmt->fetch();

if (!$user) {

echo json\_encode(['success' => false, 'error' => 'Користувача не знайдено']);

exit;

}

$username = preg\_replace('/[^a-zA-Z0-9\_-]/', '', $user['username']); // для безпеки

// Перевірка, що зображення належить користувачу

$stmt = $pdo->prepare("SELECT image\_path FROM images WHERE id = ? AND user\_id = ?");

$stmt->execute([$imageId, $userId]);

$original = $stmt->fetch();

if (!$original) {

echo json\_encode(['success' => false, 'error' => 'Оригінальне зображення не знайдено']);

exit;

}

// Параметри нового файлу

$ext = 'jpg';

$timestamp = round(microtime(true) \* 1000); // мілісекунди

$newFileName = $username . '\_' . $timestamp . '\_edited.' . $ext;

$uploadsDir = \_\_DIR\_\_ . '/../uploads/images/';

$webPath = '/uploads/images/';

$fullPath = $uploadsDir . $newFileName;

$relativePath = $webPath . $newFileName;

// Спроба зберегти файл

if (file\_put\_contents($fullPath, $imageData) === false) {

echo json\_encode(['success' => false, 'error' => 'Не вдалося зберегти зображення']);

exit;

}

// Додавання запису в базу

$stmt = $pdo->prepare("INSERT INTO images (user\_id, image\_path, created\_at) VALUES (?, ?, NOW())");

$success = $stmt->execute([$userId, $relativePath]);

if ($success) {

echo json\_encode(['success' => true, 'path' => $relativePath]);

} else {

echo json\_encode(['success' => false, 'error' => 'Не вдалося записати в базу']);

}

save\_profile.php:

<?php

session\_start();

require\_once \_\_DIR\_\_ . '/../config/database.php';

if (!isset($\_SESSION['user\_id'])) {

header('Location: ../index.php?page=login');

exit;

}

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

$username = trim($\_POST['username'] ?? '');

$email = trim($\_POST['email'] ?? '');

$notify = isset($\_POST['notify\_new\_comment']) ? 1 : 0;

$old\_photo = $\_POST['old\_photo\_path'] ?? '';

$server\_selected = $\_POST['selected\_server\_photo'] ?? '';

$new\_photo\_path = $old\_photo;

// Якщо завантажено новий файл

if (isset($\_FILES['photo\_user']) && $\_FILES['photo\_user']['error'] === UPLOAD\_ERR\_OK) {

$tmp = $\_FILES['photo\_user']['tmp\_name'];

$ext = strtolower(pathinfo($\_FILES['photo\_user']['name'], PATHINFO\_EXTENSION));

$allowed = ['jpg', 'jpeg', 'png', 'gif'];

if (in\_array($ext, $allowed)) {

$pdo = new PDO("mysql:host=" . DB\_HOST . ";dbname=" . DB\_NAME . ";charset=utf8", DB\_USER, DB\_PASS, [

PDO::ATTR\_ERRMODE => PDO::ERRMODE\_EXCEPTION

]);

$stmt = $pdo->prepare("SELECT username FROM users WHERE id = ?");

$stmt->execute([$user\_id]);

$row = $stmt->fetch();

$username\_db = $row ? $row['username'] : 'user';

$filename = $username\_db . '\_' . round(microtime(true) \* 1000) . '.' . $ext;

$upload\_dir = \_\_DIR\_\_ . '/../uploads/images/';

$destination = $upload\_dir . $filename;

if (!is\_dir($upload\_dir)) {

mkdir($upload\_dir, 0775, true);

}

if (move\_uploaded\_file($tmp, $destination)) {

$new\_photo\_path = '/uploads/images/' . $filename;

// Перевірка дубля

$checkStmt = $pdo->prepare("SELECT COUNT(\*) FROM images WHERE user\_id = ? AND image\_path = ?");

$checkStmt->execute([$user\_id, $new\_photo\_path]);

$exists = $checkStmt->fetchColumn();

if (!$exists) {

$insertStmt = $pdo->prepare("INSERT INTO images (user\_id, image\_path) VALUES (?, ?)");

$insertStmt->execute([$user\_id, $new\_photo\_path]);

}

}

}

} elseif (!empty($server\_selected)) {

$new\_photo\_path = $server\_selected;

}

// Оновлення в БД

$pdo = new PDO("mysql:host=" . DB\_HOST . ";dbname=" . DB\_NAME . ";charset=utf8", DB\_USER, DB\_PASS, [

PDO::ATTR\_ERRMODE => PDO::ERRMODE\_EXCEPTION

]);

$stmt = $pdo->prepare("UPDATE users SET username = ?, email = ?, notify\_new\_comment = ?, photo\_user\_path = ? WHERE id = ?");

$stmt->execute([$username, $email, $notify, $new\_photo\_path, $user\_id]);

header('Location: ../index.php?page=dashboard');

exit;

save\_screenshot.php:

<?php

session\_start();

//file\_put\_contents(\_\_DIR\_\_ . '/debug\_session.txt', print\_r($\_SESSION, true));

require\_once \_\_DIR\_\_ . '/../config/database.php';

/\*if (!isset($\_SESSION['user\_id']) || !isset($\_SESSION['username'])) {

http\_response\_code(403);

echo json\_encode(['success' => false, 'error' => 'Користувач не авторизований']);

exit;

}\*/

$data = json\_decode(file\_get\_contents('php://input'), true);

if (!isset($data['image'])) {

http\_response\_code(400);

echo json\_encode(['success' => false, 'error' => 'Немає зображення']);

exit;

}

$imageData = $data['image'];

$imageData = str\_replace('data:image/jpeg;base64,', '', $imageData);

$imageData = str\_replace(' ', '+', $imageData);

$decodedImage = base64\_decode($imageData);

//file\_put\_contents(\_\_DIR\_\_ . '/debug\_image\_data.txt', print\_r($imageData, true), FILE\_APPEND);

if (!$decodedImage) {

http\_response\_code(400);

echo json\_encode(['success' => false, 'error' => 'Невірне зображення']);

exit;

}

// Маска: screenshot\_<username>\_<мілісекунди>.jpg

$username = preg\_replace('/[^a-zA-Z0-9\_\-]/', '', $\_SESSION['username']);

$current\_time = round(microtime(true) \* 1000);

$filename = "screenshot\_{$username}\_{$current\_time}.jpg";

$savePath = "/uploads/images/" . $filename;

$absolutePath = $\_SERVER['DOCUMENT\_ROOT'] . $savePath;

// Збереження файлу

if (!file\_exists(dirname($absolutePath))) {

mkdir(dirname($absolutePath), 0755, true);

}

/\* Дебаг \*/

file\_put\_contents(\_\_DIR\_\_ . '/debug\_log.txt', "\_\_DIR\_\_: " . \_\_DIR\_\_ . "\n", FILE\_APPEND);

file\_put\_contents(\_\_DIR\_\_ . '/debug\_log.txt', "Абсолютний шлях: $absolutePath\n", FILE\_APPEND);

file\_put\_contents(\_\_DIR\_\_ . '/debug\_log.txt', "Чи існує директорія? " . (is\_dir(dirname($absolutePath)) ? 'Так' : 'Ні') . "\n", FILE\_APPEND);

file\_put\_contents(\_\_DIR\_\_ . '/debug\_log.txt', "Права доступу до директорії: " . decoct(fileperms(dirname($absolutePath)) & 0777) . "\n", FILE\_APPEND);

$result = file\_put\_contents($absolutePath, $decodedImage);

file\_put\_contents(\_\_DIR\_\_ . '/debug\_log.txt', "Результат запису: $result\n", FILE\_APPEND);

if ($result === false) {

file\_put\_contents(\_\_DIR\_\_ . '/debug\_log.txt', "ПОМИЛКА ЗАПИСУ В ФАЙЛ!\n", FILE\_APPEND);

}

// Запис в БД

$pdo = new PDO("mysql:host=" . DB\_HOST . ";dbname=" . DB\_NAME . ";charset=utf8", DB\_USER, DB\_PASS, [

PDO::ATTR\_ERRMODE => PDO::ERRMODE\_EXCEPTION

]);

$stmt = $pdo->prepare("INSERT INTO images (user\_id, image\_path, created\_at) VALUES (:user\_id, :image\_path, NOW())");

$stmt->execute([

':user\_id' => $\_SESSION['user\_id'],

':image\_path' => $savePath

]);

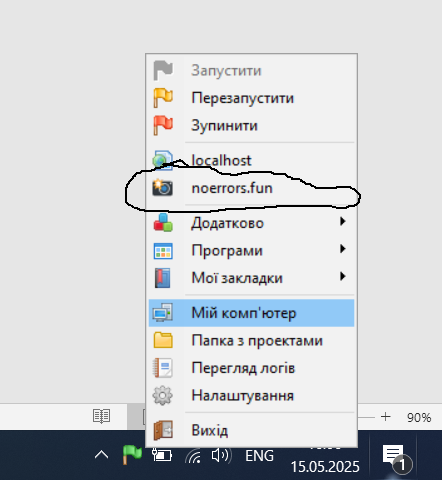
echo json\_encode(['success' => true]);

**Резуьтати**

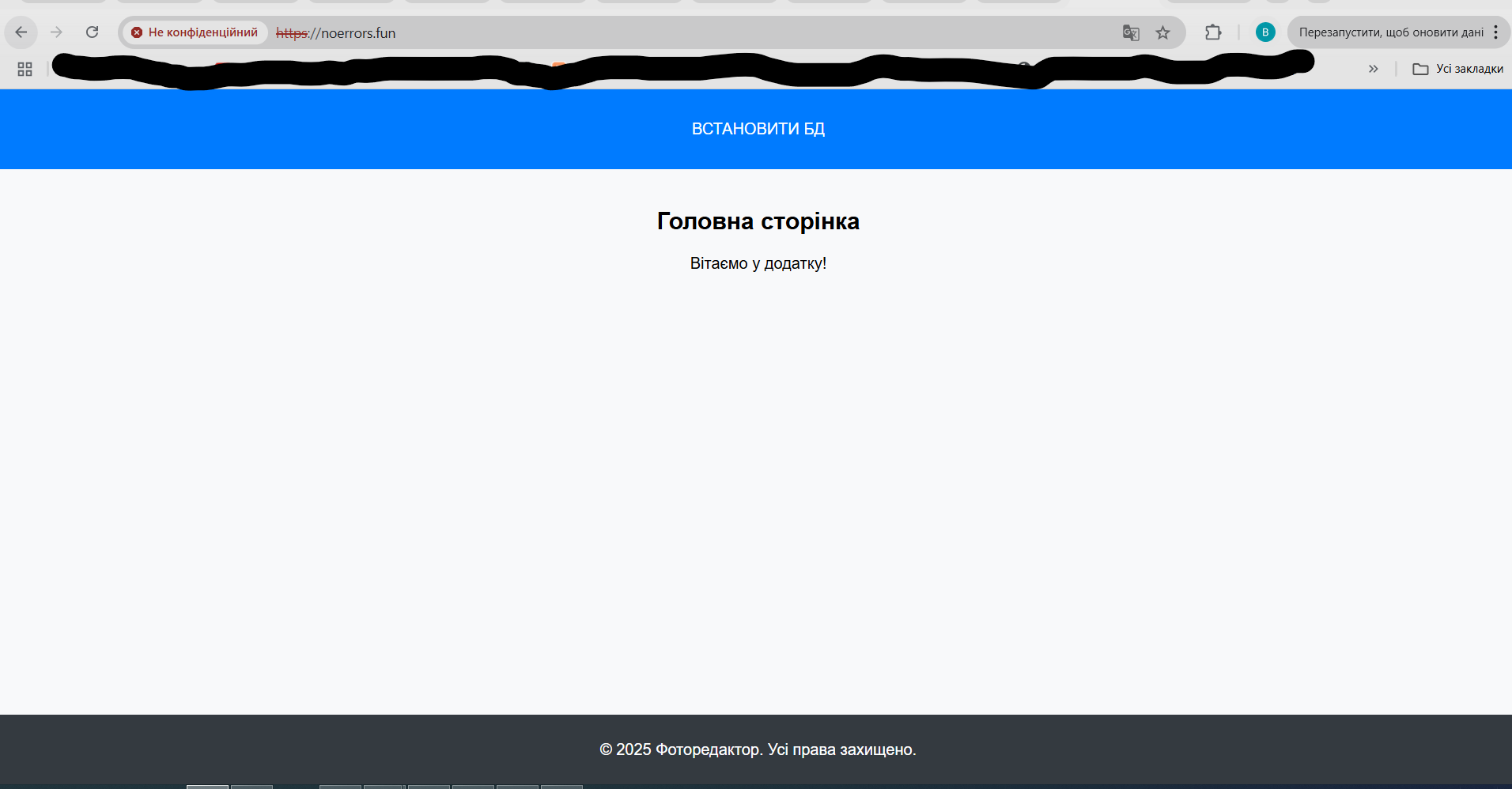
Запускаемо OpenServer:



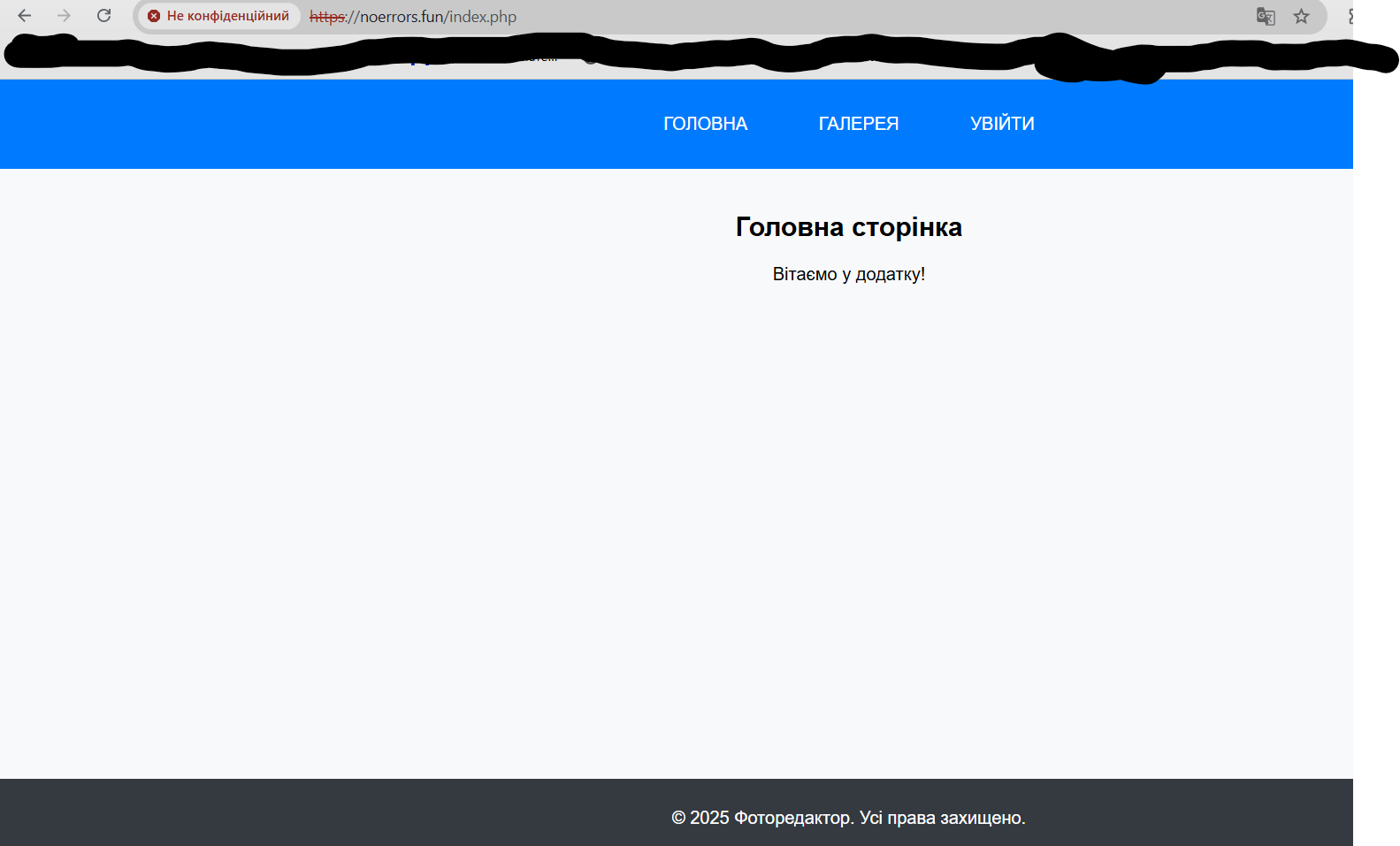
Заходимо на noerrors.fun:



Переходимо на https://noerrors.fun/

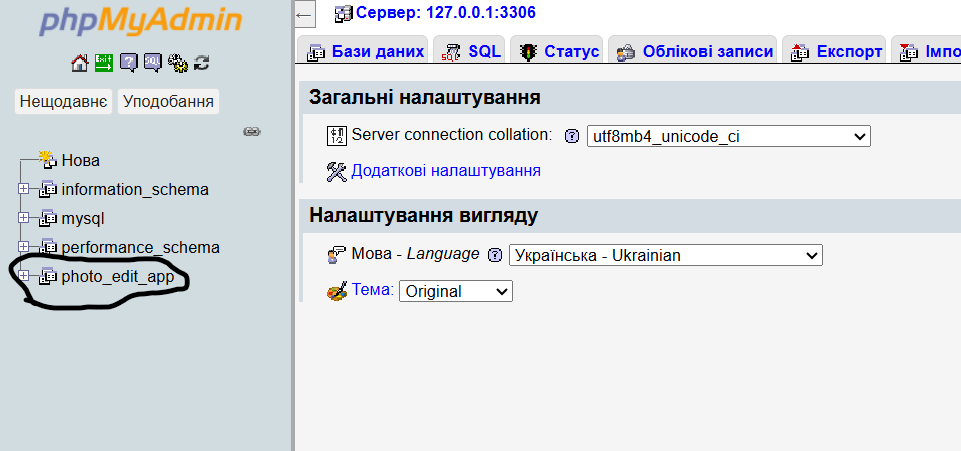


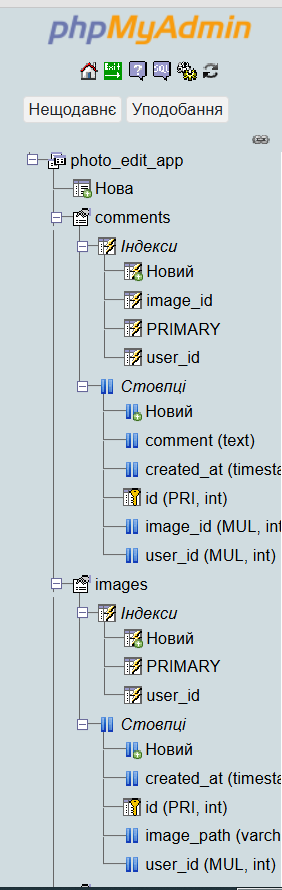
Нажимаємо ***встановити БД:***

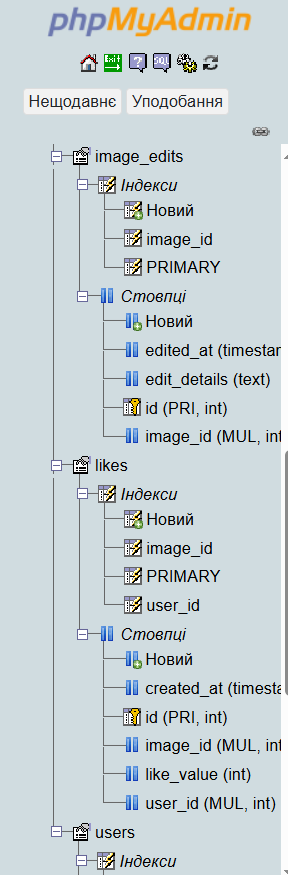


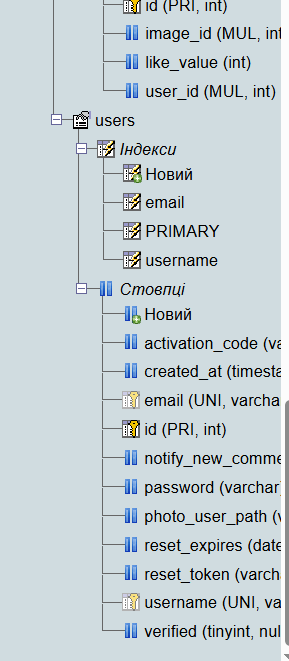
Нас вітає сторінка додатку

Перевіряємо на phpMysql:

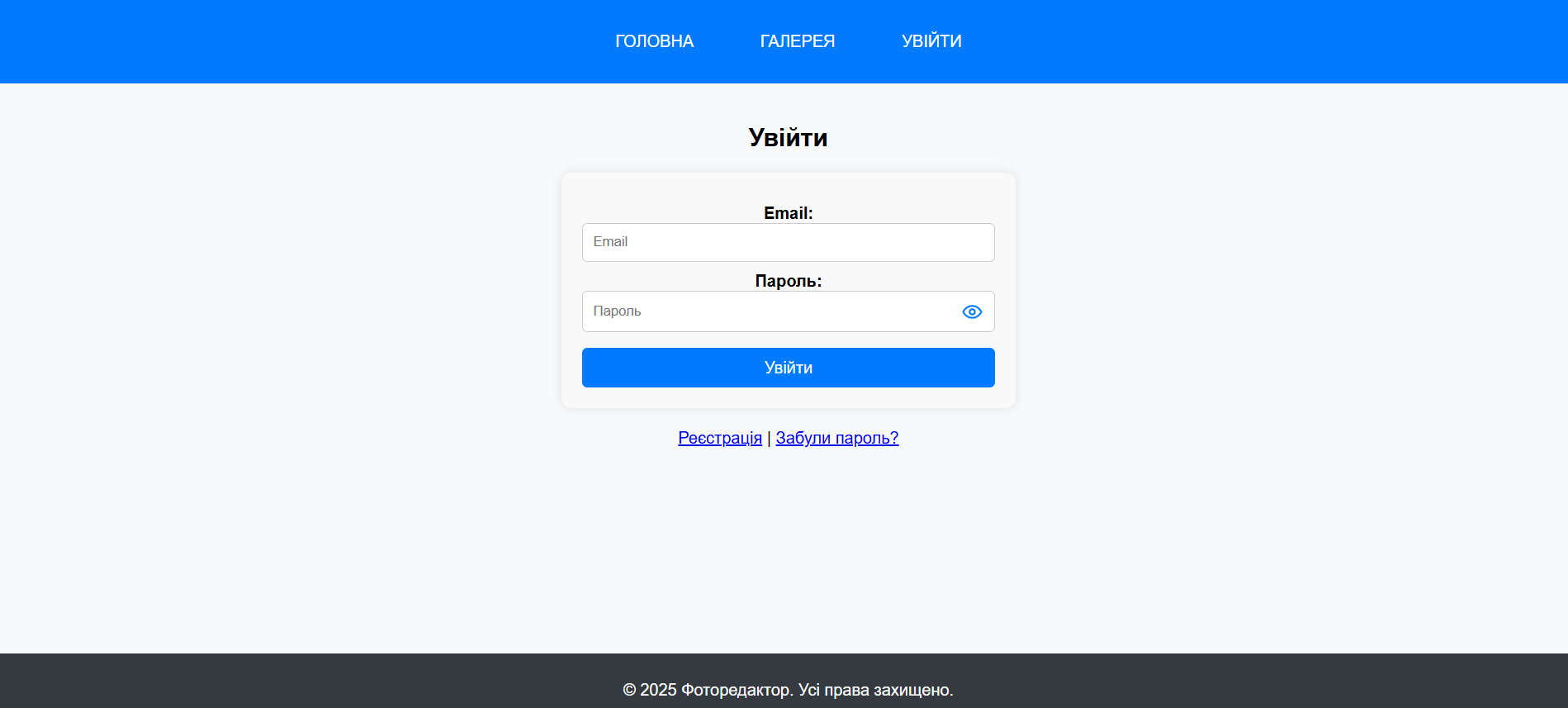




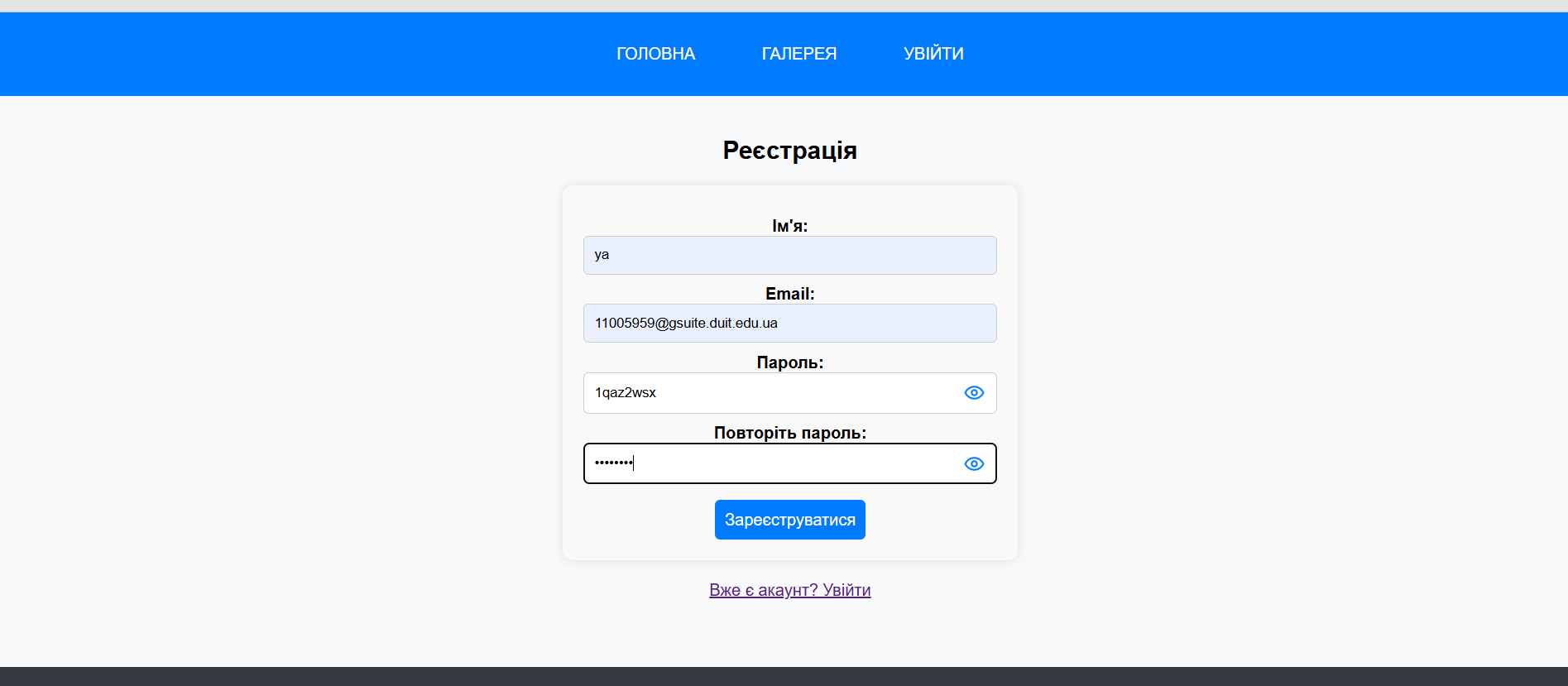




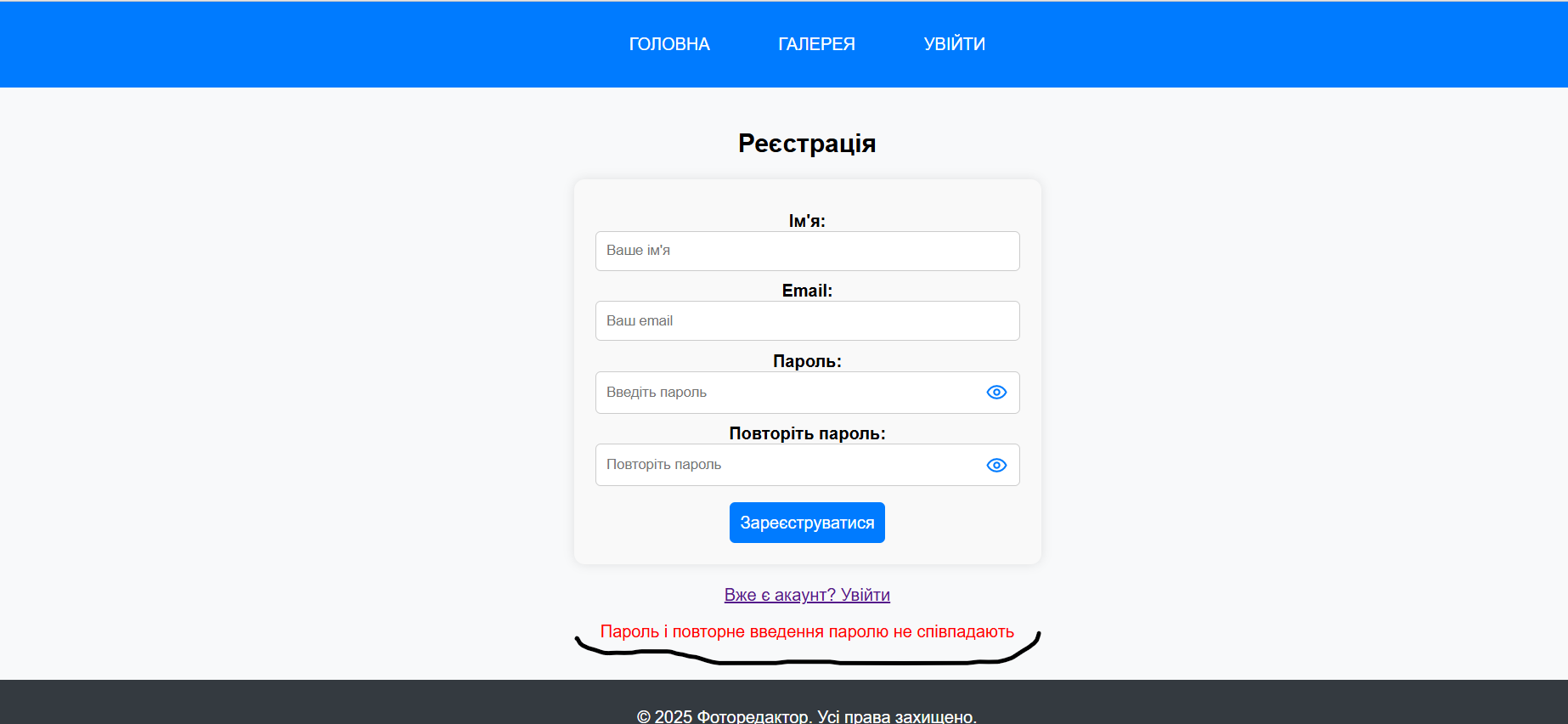
Нажимаємо ***Увійти***:



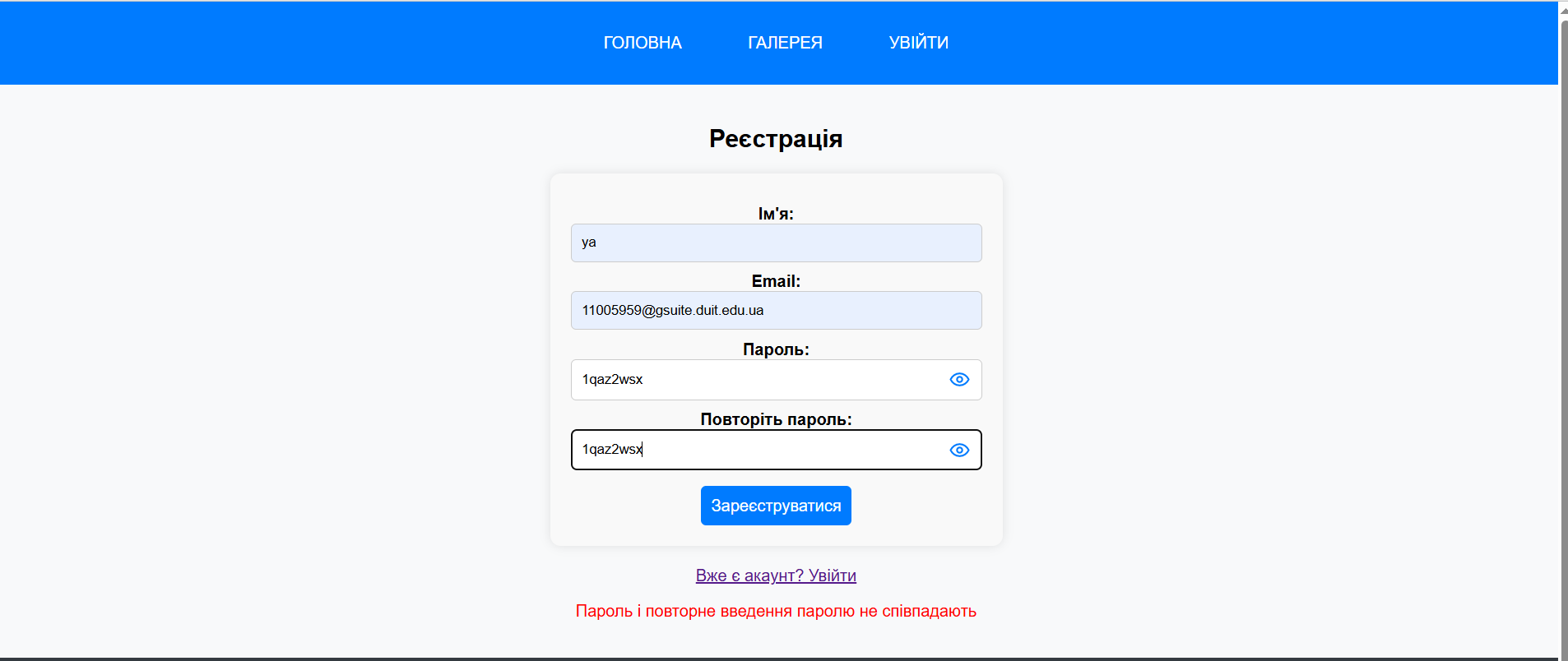
Далі натискаємо ***Реєстрація***:

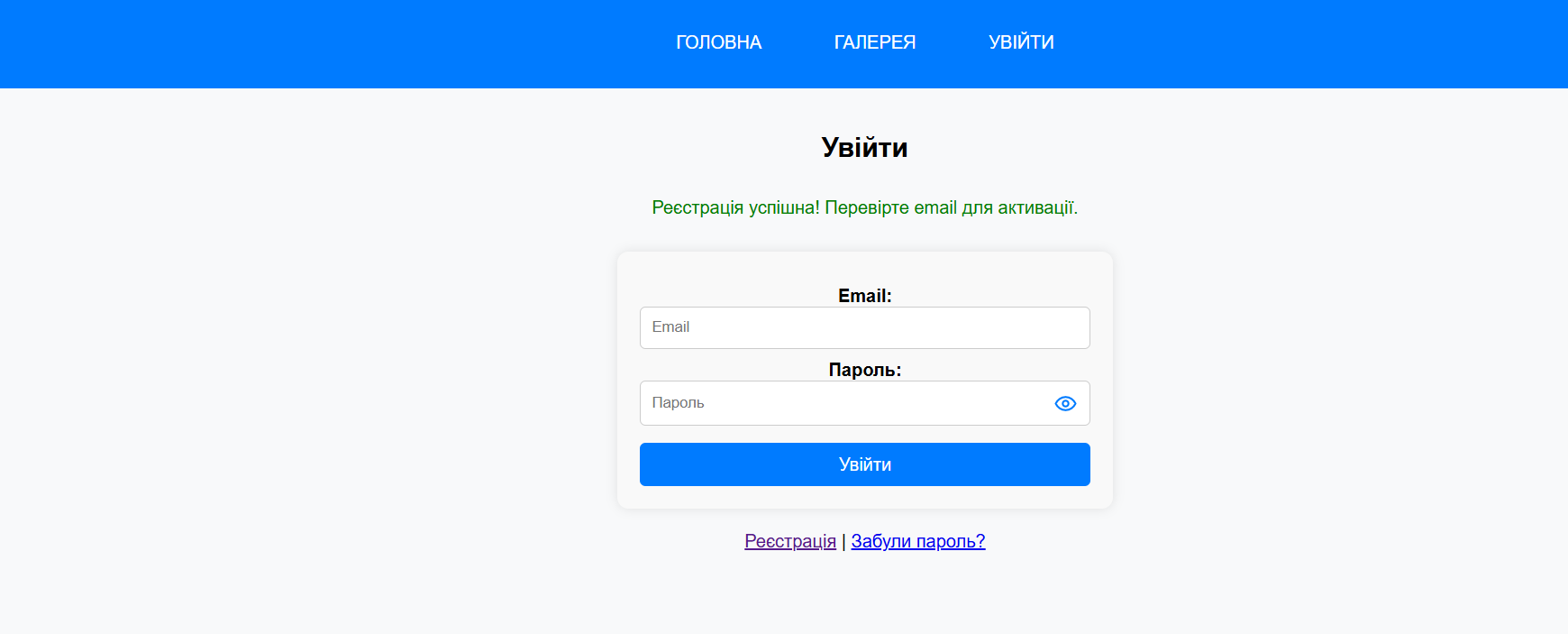


Реєструемося та “випадково” вводимо неправильний другий пароль (*пароль можна сховати та показати*)

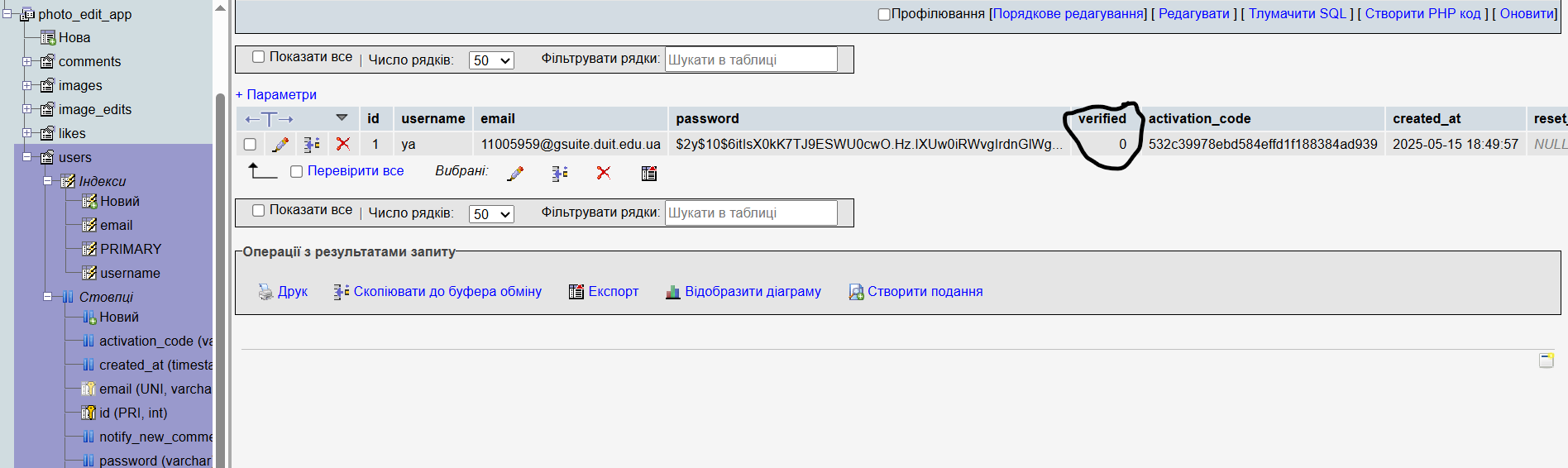


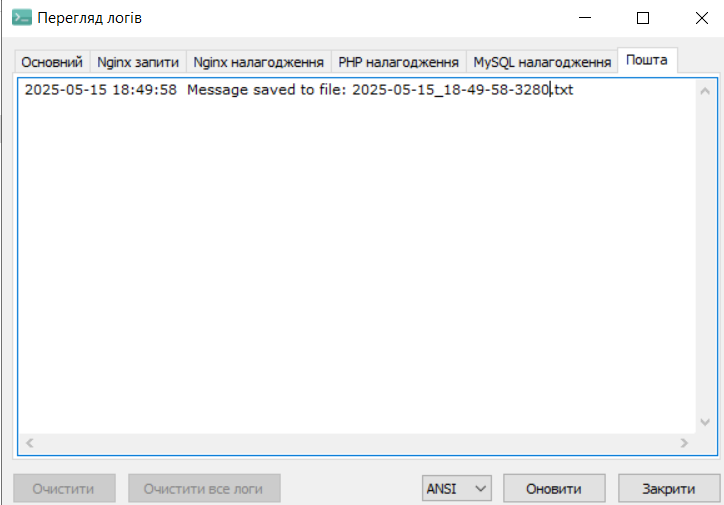
Вводимо все правильно та натискаємо зареєструватися

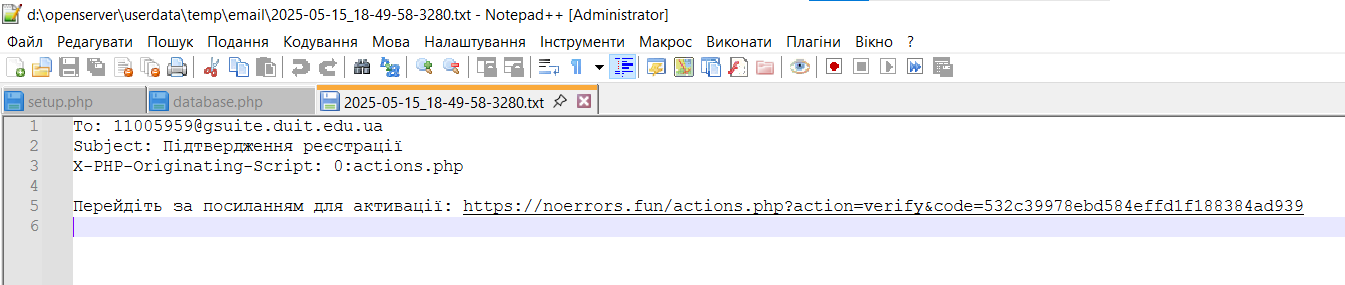




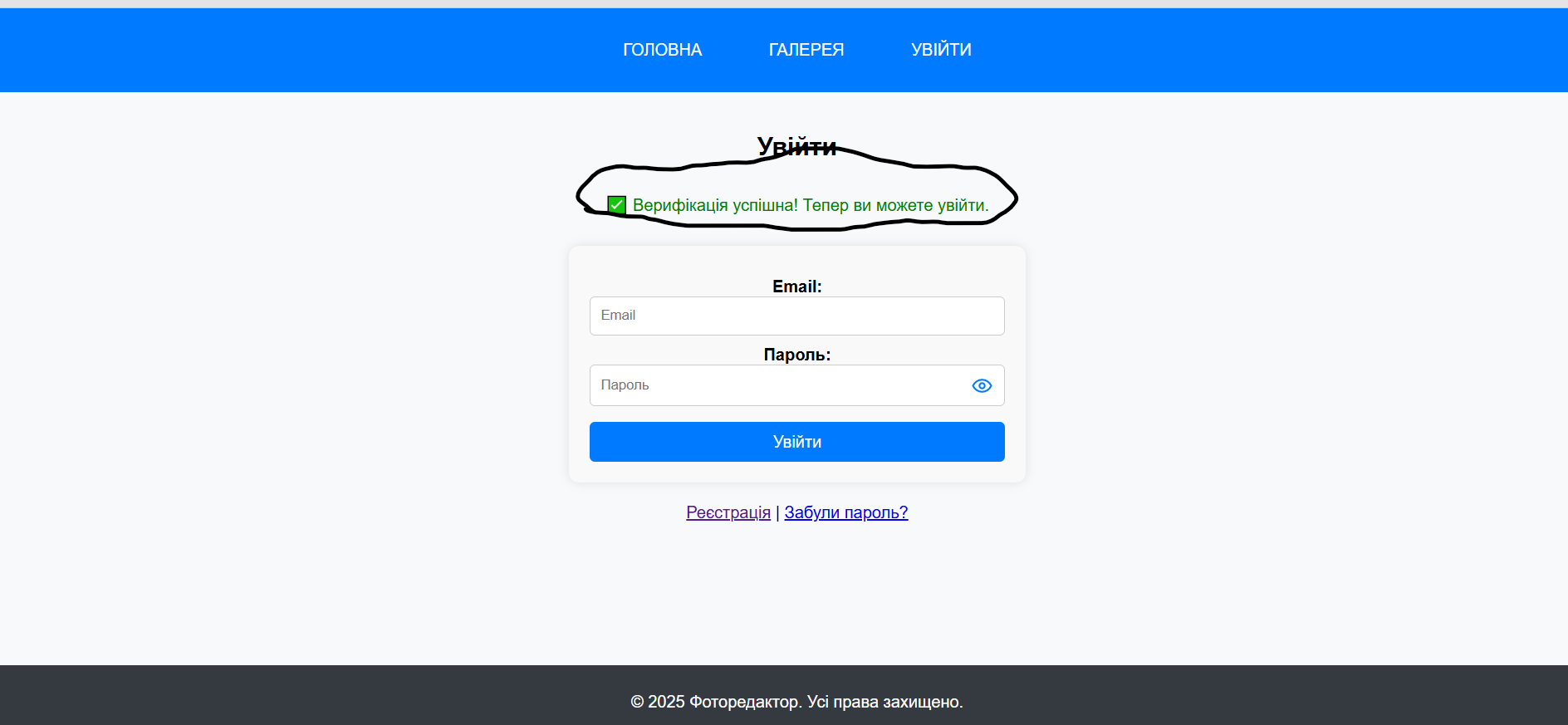
У PhpMySql у таблиці users з’явился новий юзер але поки він неверифікований виділене поле буде дорівнювати 0:

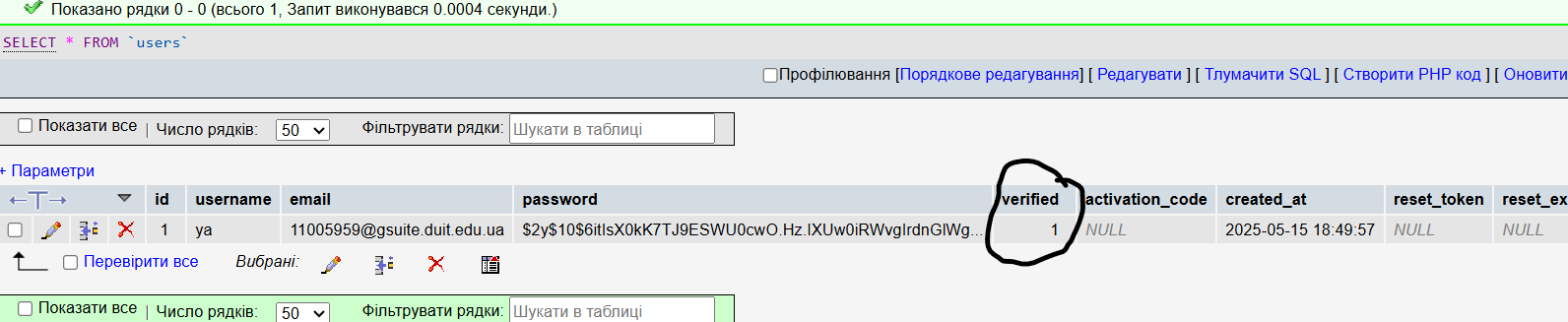


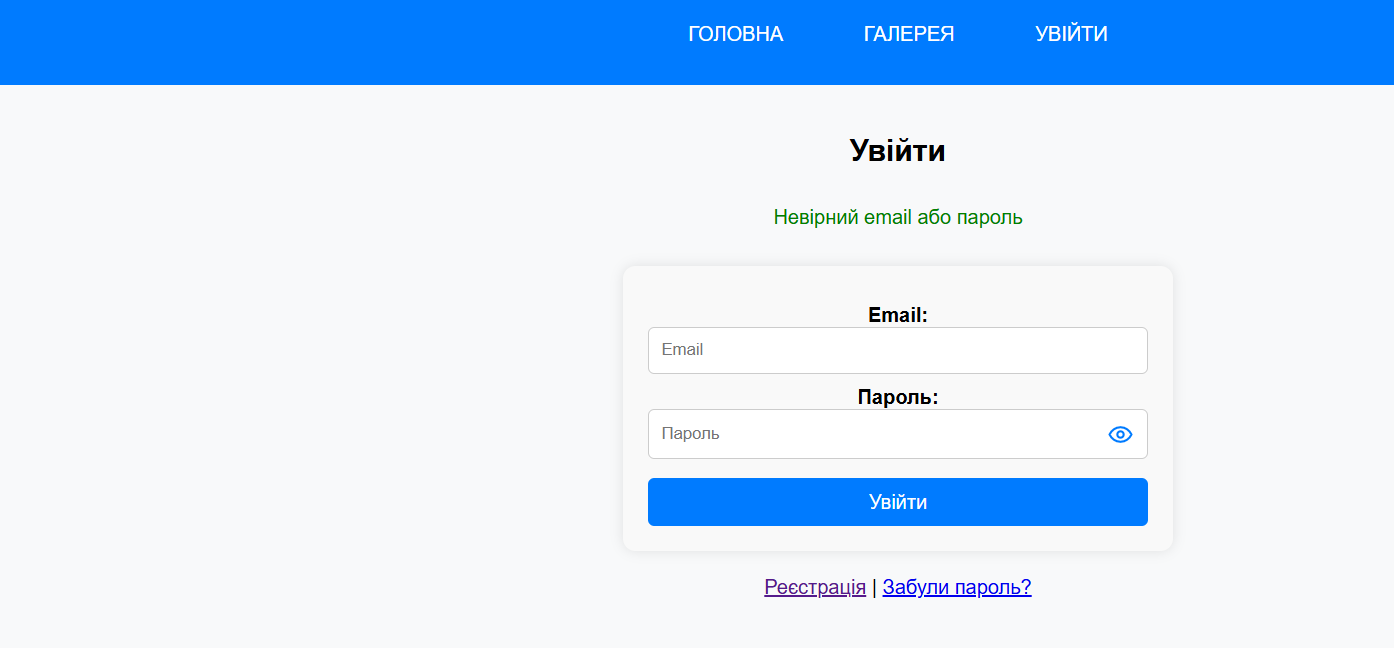




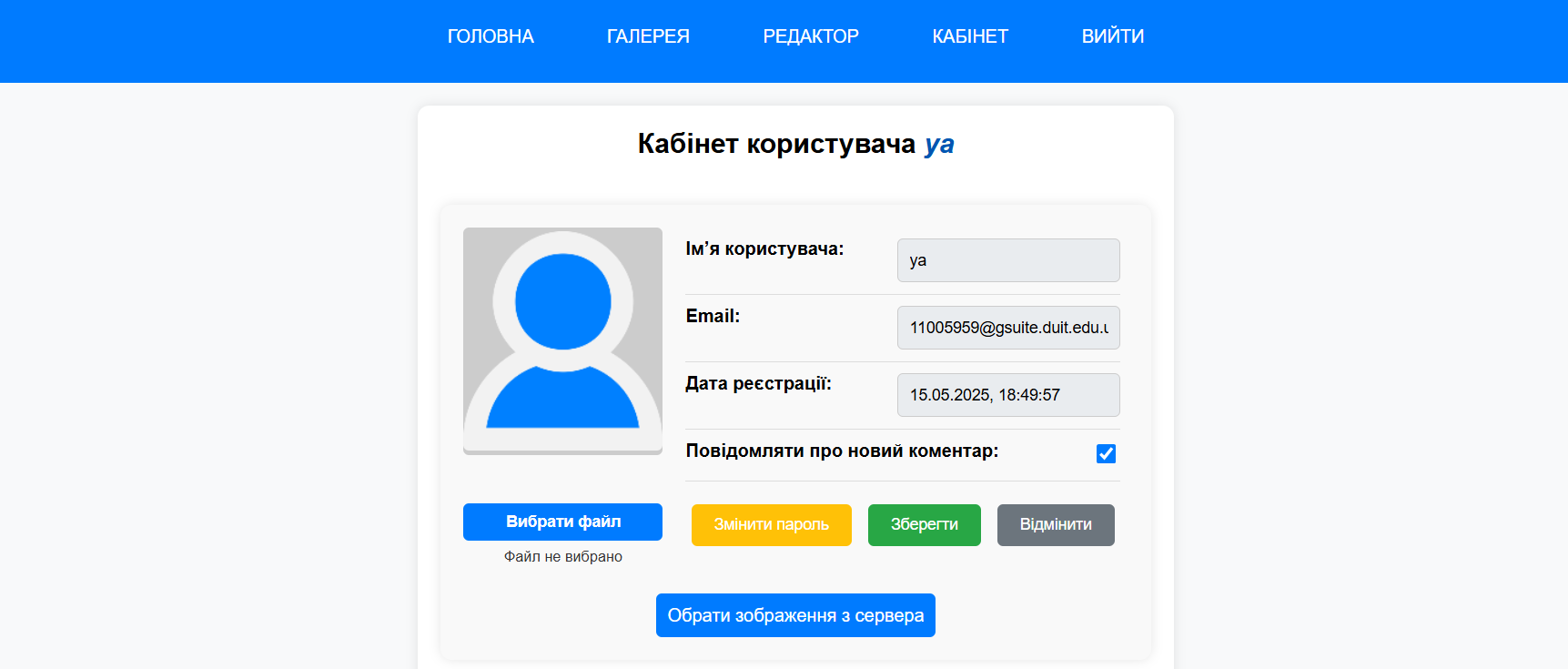
Верифікуємо:





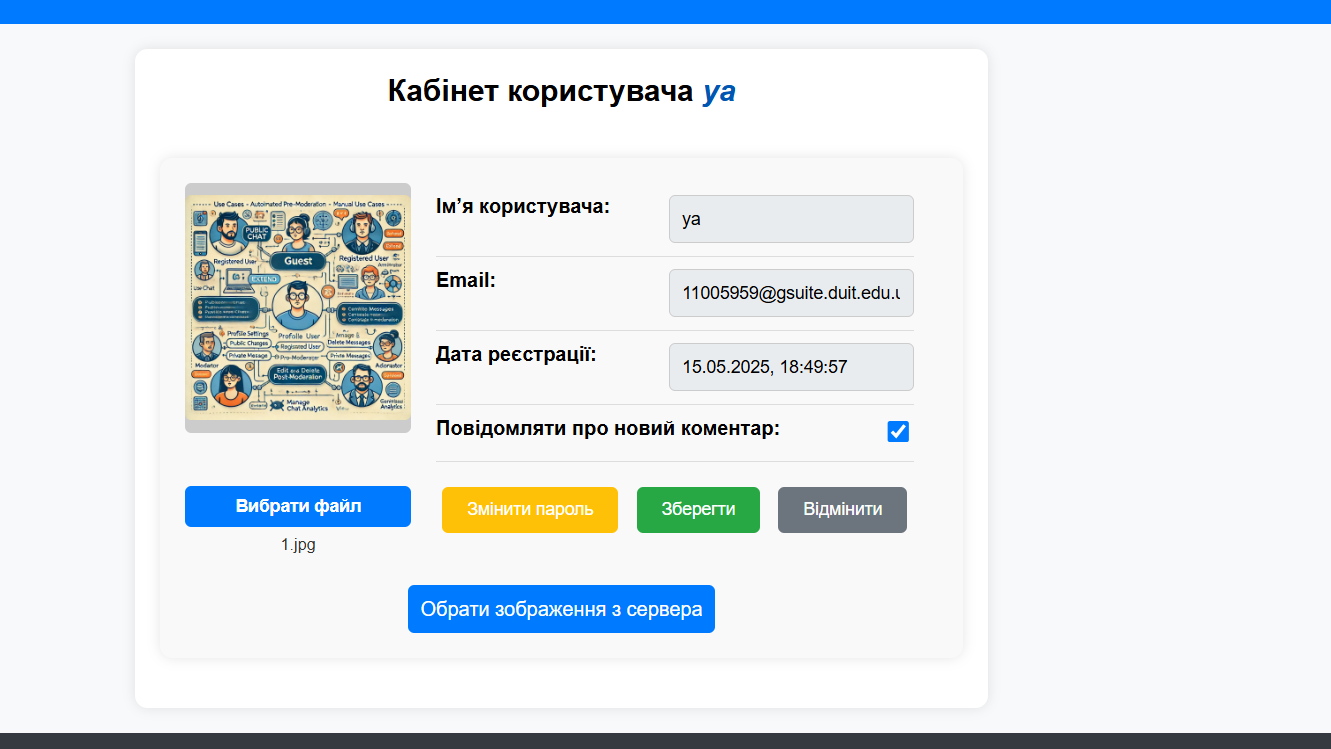
Знову “випадково” вводимо неправильний пароль:  


Вводимо правильний:

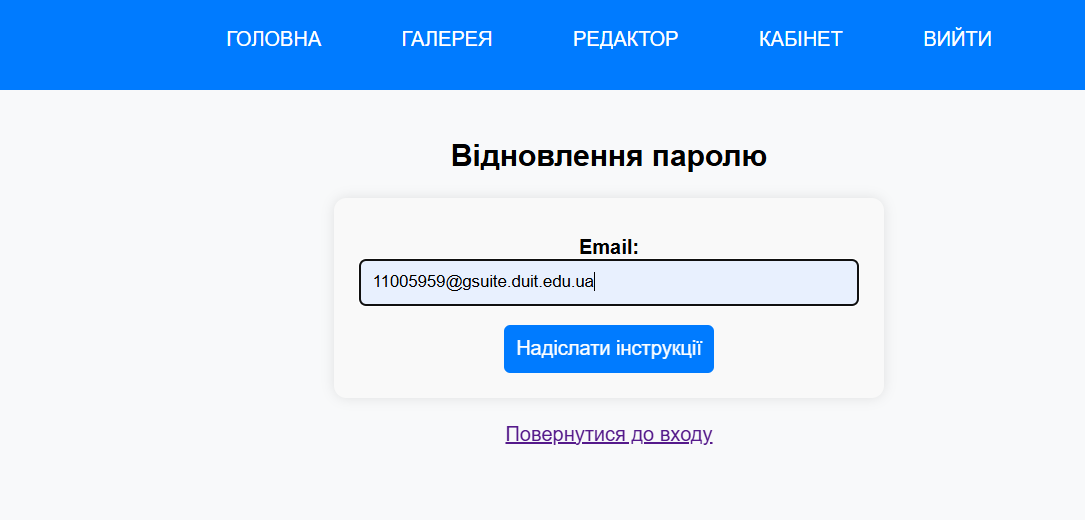


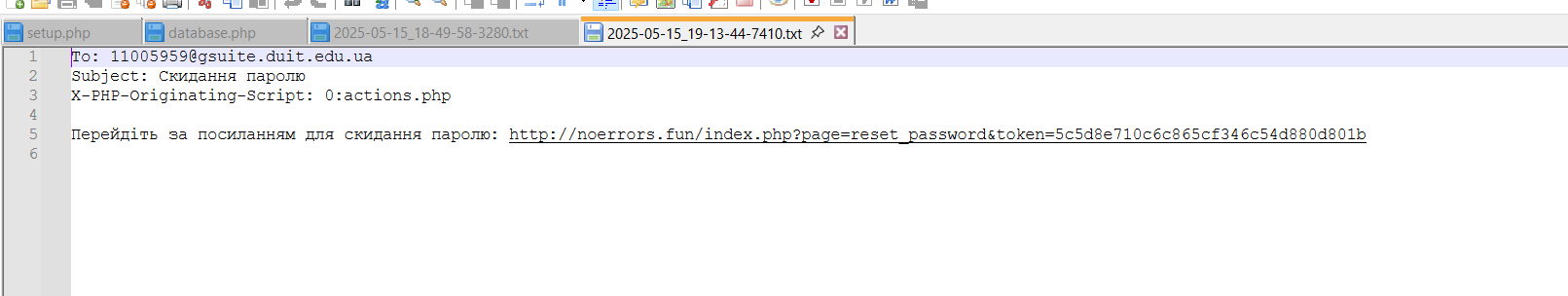
Тут ми можемо:

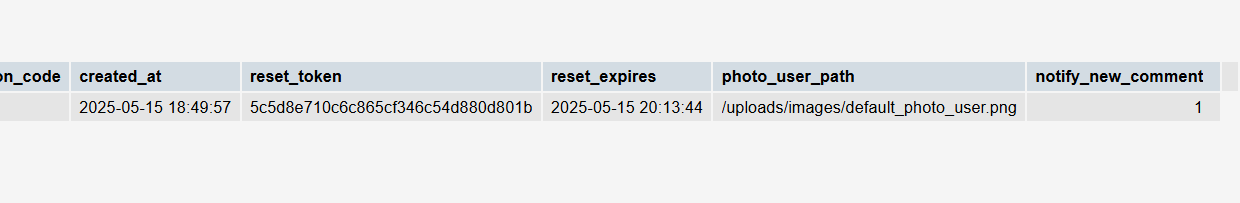
* Вибрати файл для аватарки



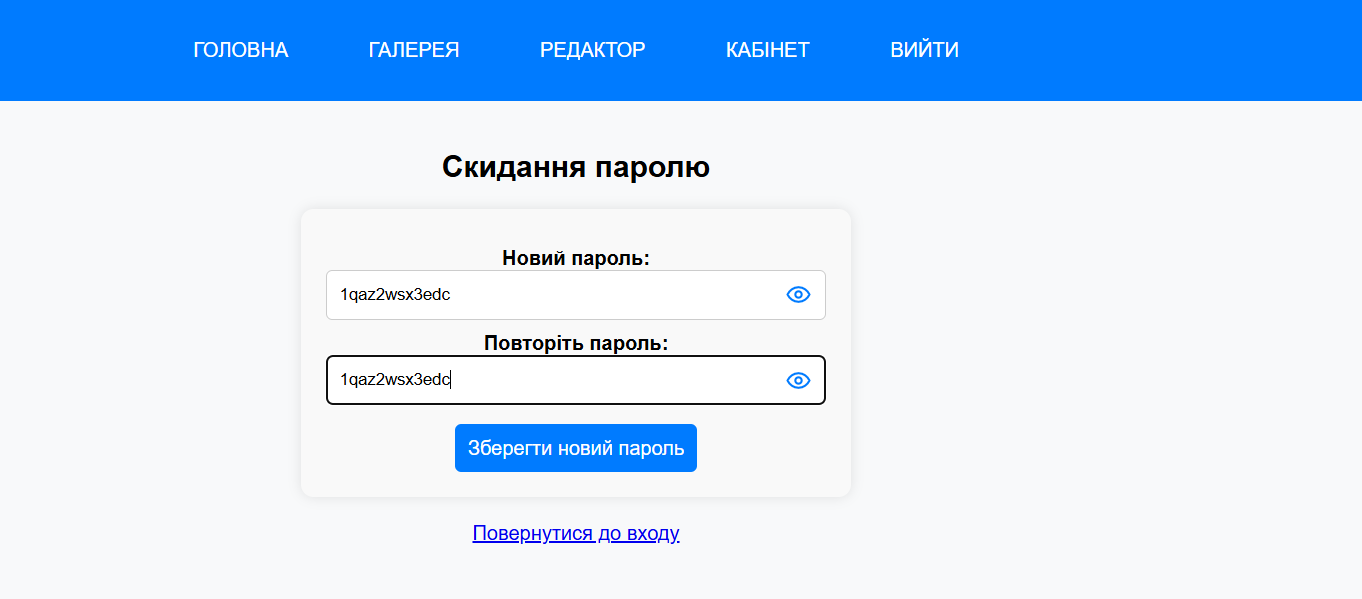
* Змінити пароль:



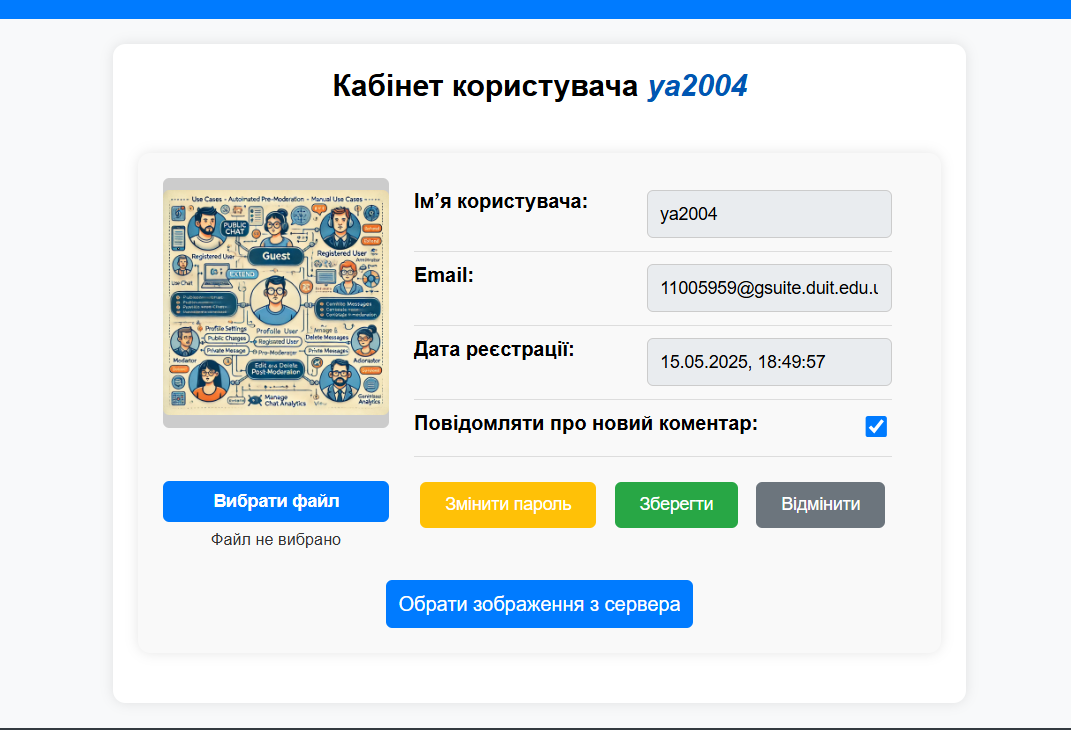


Формується токен який буде дійсний 1 годину:  


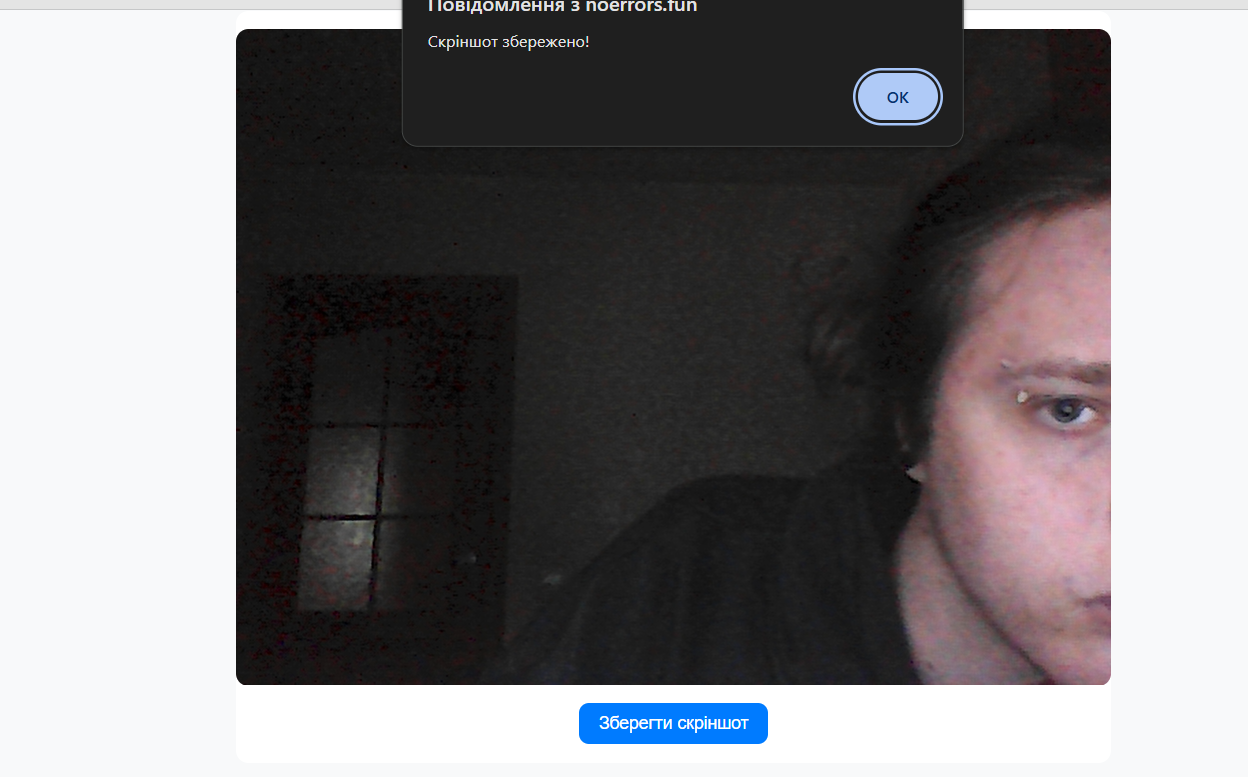
Вводимо новий пароль:



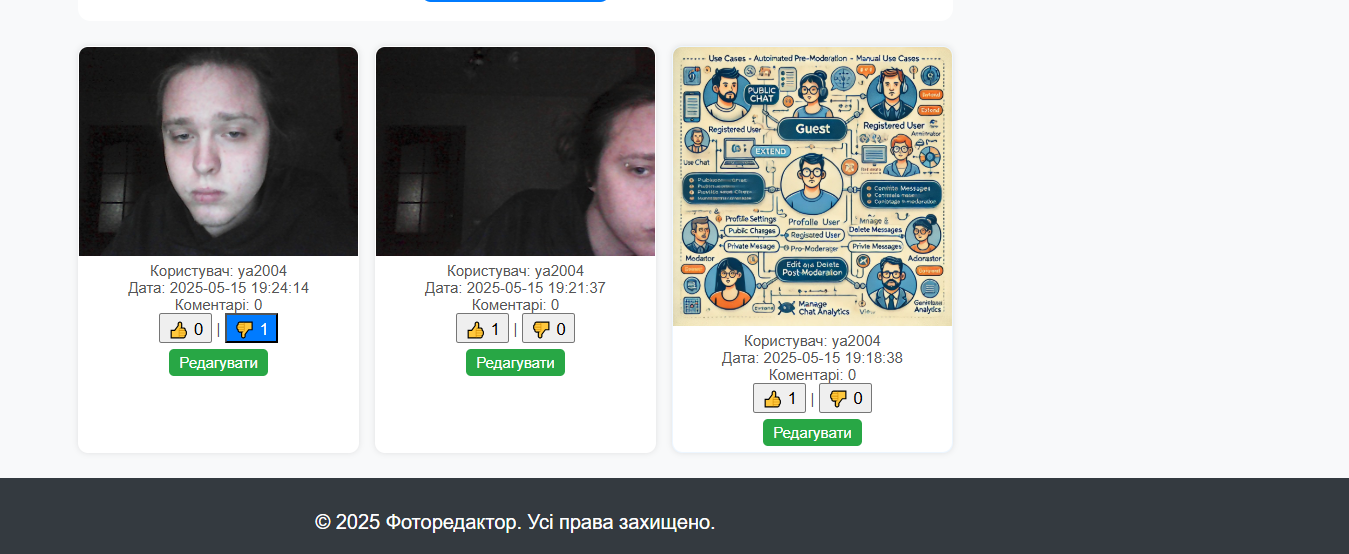
* Можемо змінити аватарку та ім’я



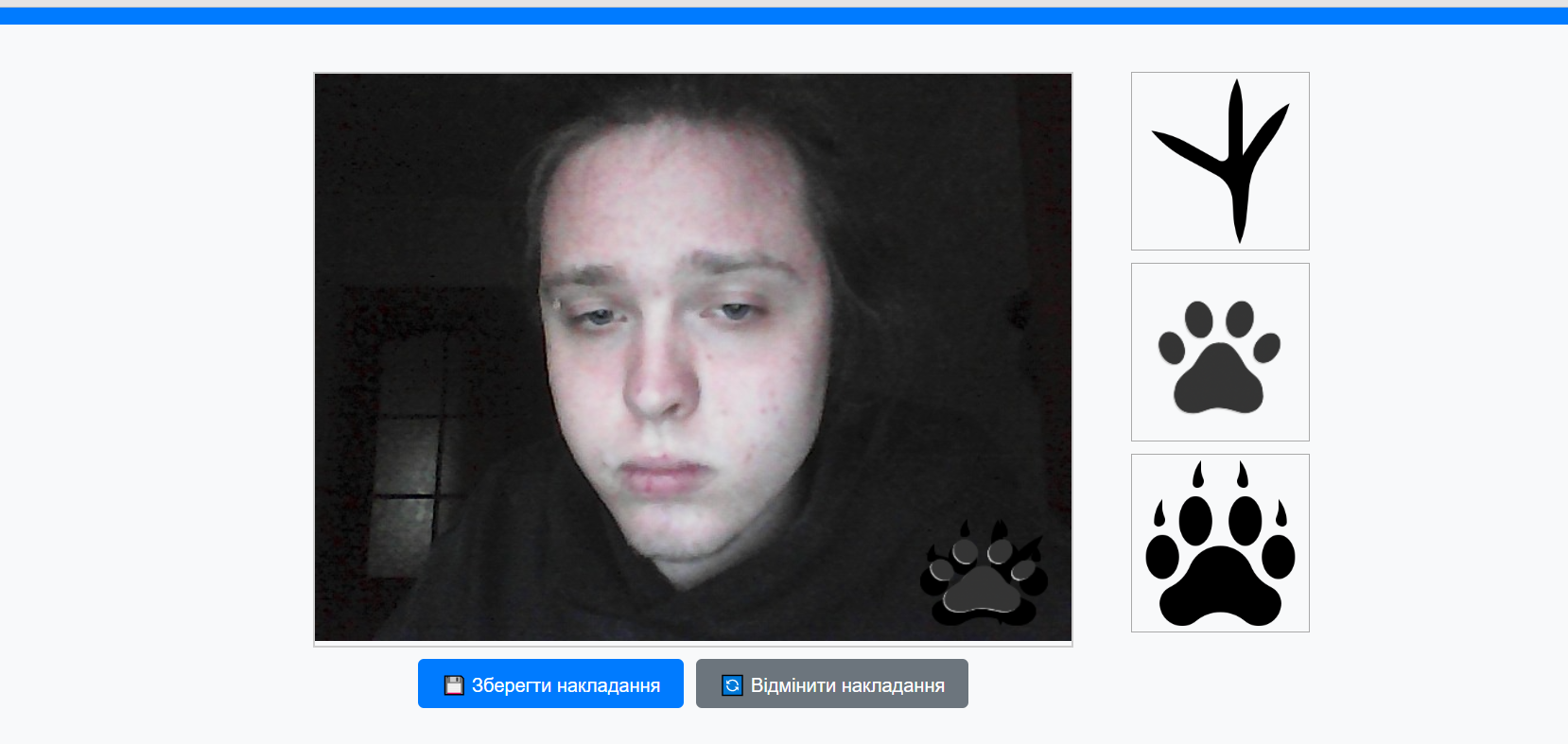
У вкладці галерея можна робити знімки з камери :



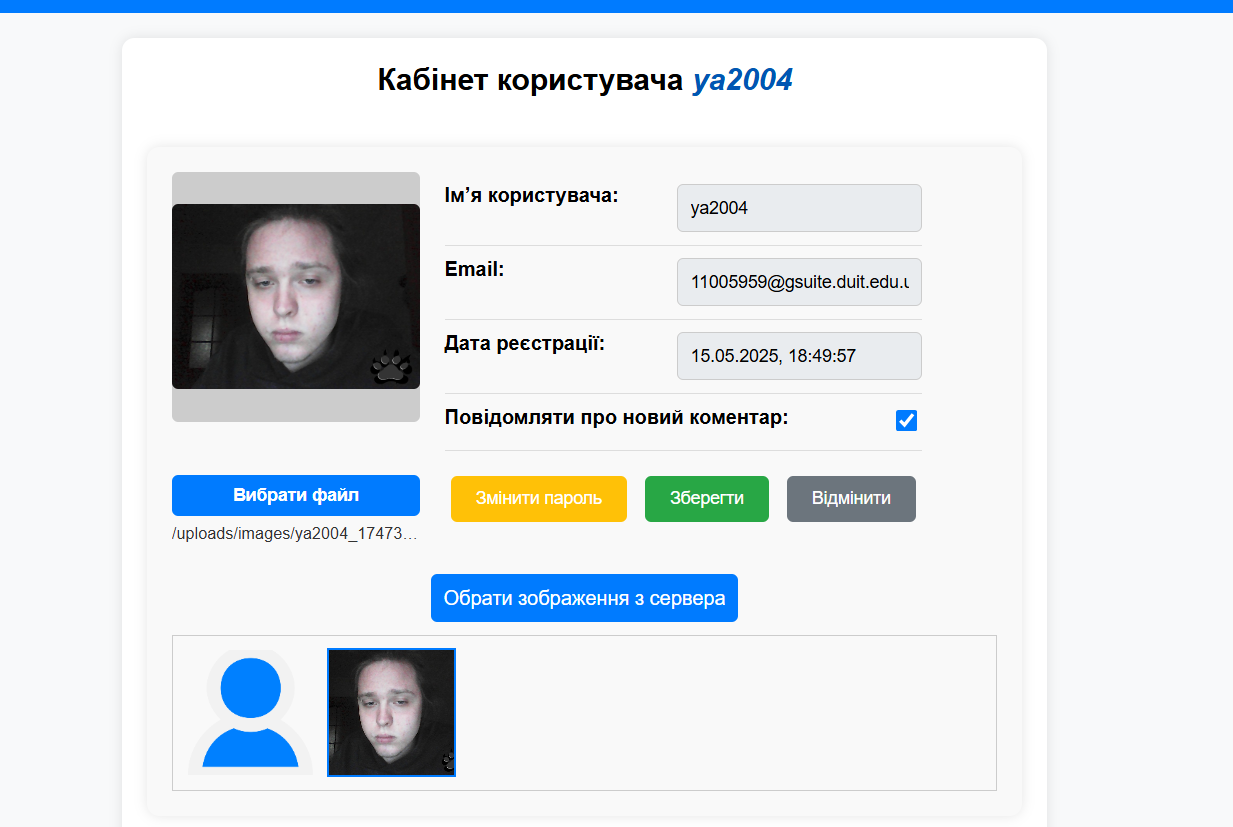
Та ставити лайки/діслайки фотографіям:



Можна натиснути на редагувати та накласти зображення:



Можна обрати файл з сервера:



Можна додавати коментарі:

