ANTONINE UNIVERSITY

Faculty of Engineering

Department of Technology in Computer Science

Computer Security Project

Task 1 Report

Carried out by Lynn Haydar, 202212989

Lynn AbdelKhalek, 202311399

Technology in Computer Science Student

Table of Contents

Project Establishment

Guide: https://www.youtube.com/watch?v=um8BtHFNJBA\

Download XAMPP, download WordPress and extract it in htdocs XAMPP,

Start Apache and MySQL servers in XAMPP.

Go to PHP my admin and create a new database

You can access now your WordPress site by this link: http://localhost/wordpress/

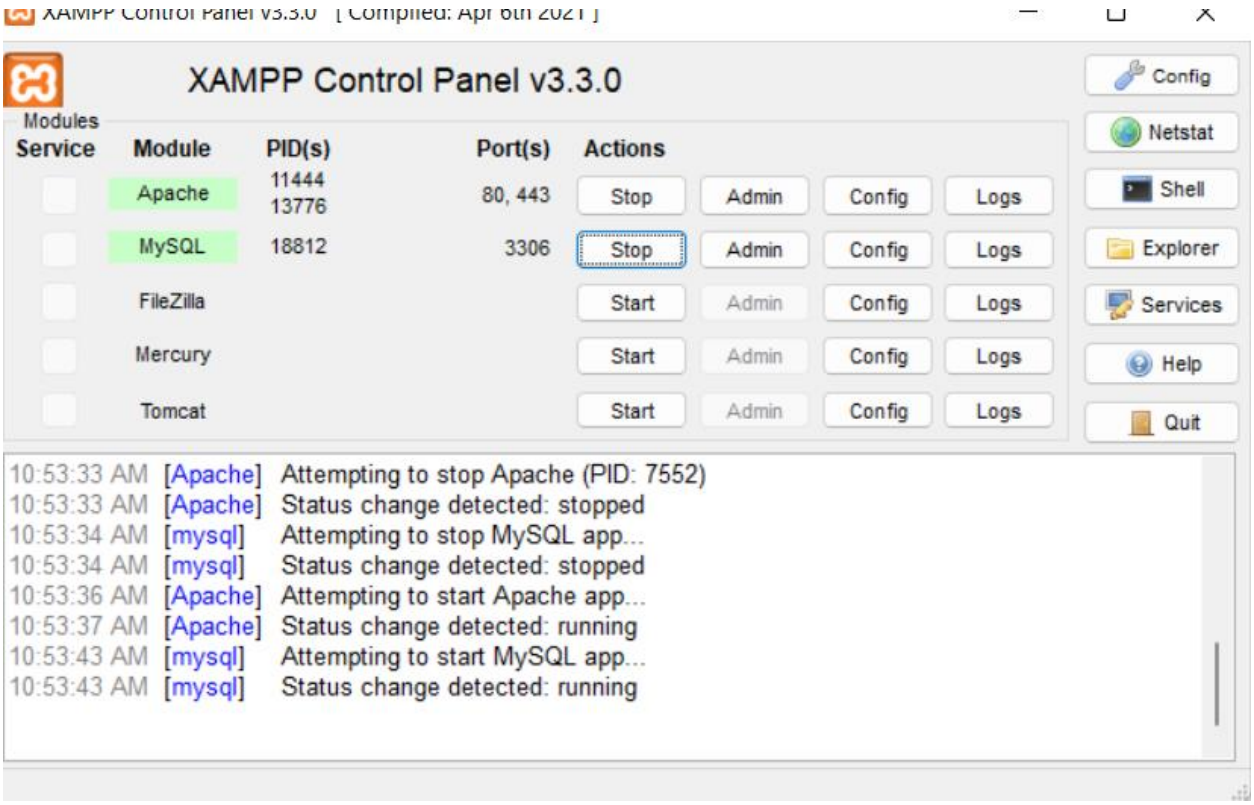


Figure 1_starting services

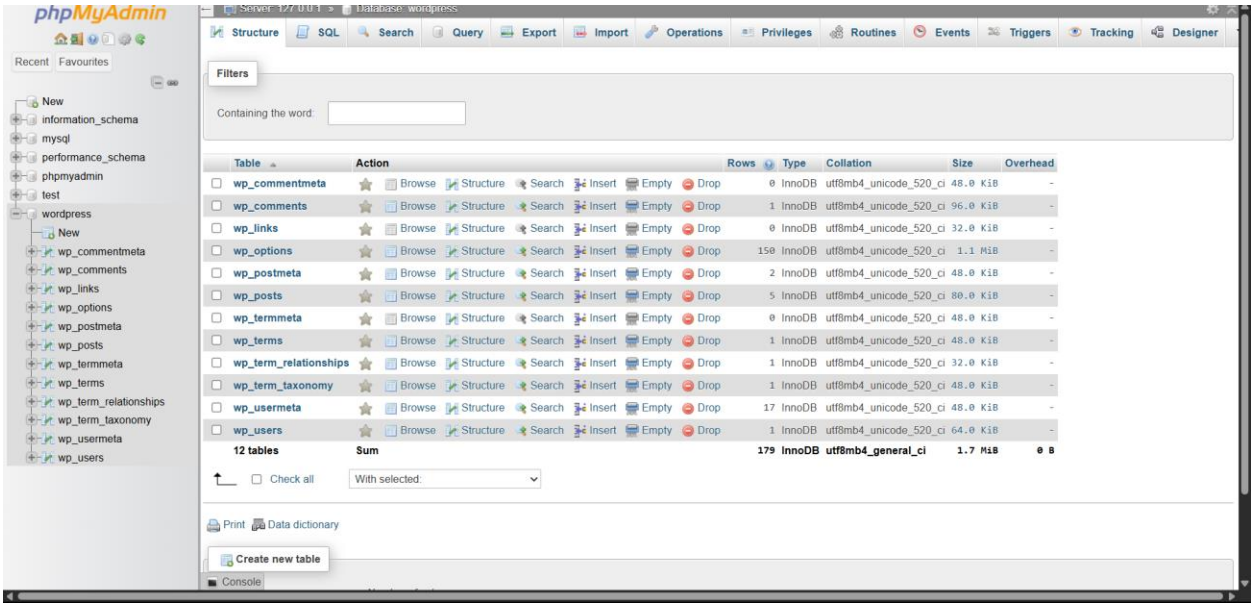


Figure 2_creating database

Installing theme for our website

Open WordPress, go to Apparencies and Themes, we chose Astra

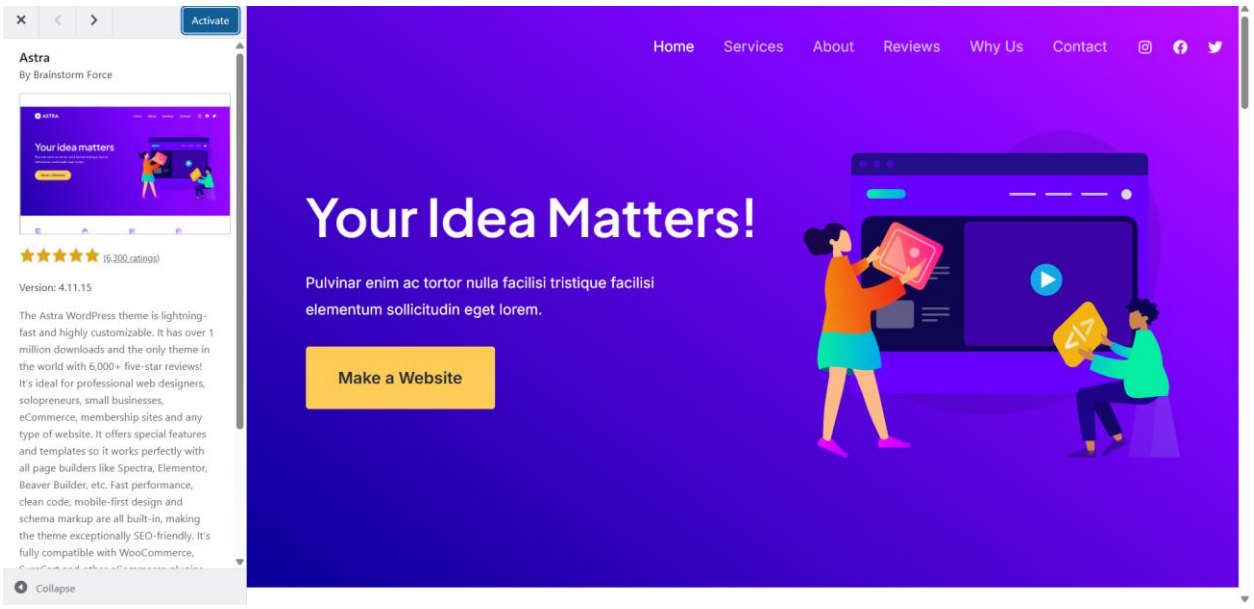


Figure 3_astra theme

Our website needs:

- Homepage
- Register system
- User roles

- Access control
- https: enable SSL on XAMPP
- Security Plugins
- File upload system
- Error logging
- SQL injection protection
- Penetration testing report

So we are going to build a photography website.

Preparing necessary pages

Prepare the necessary pages for our website

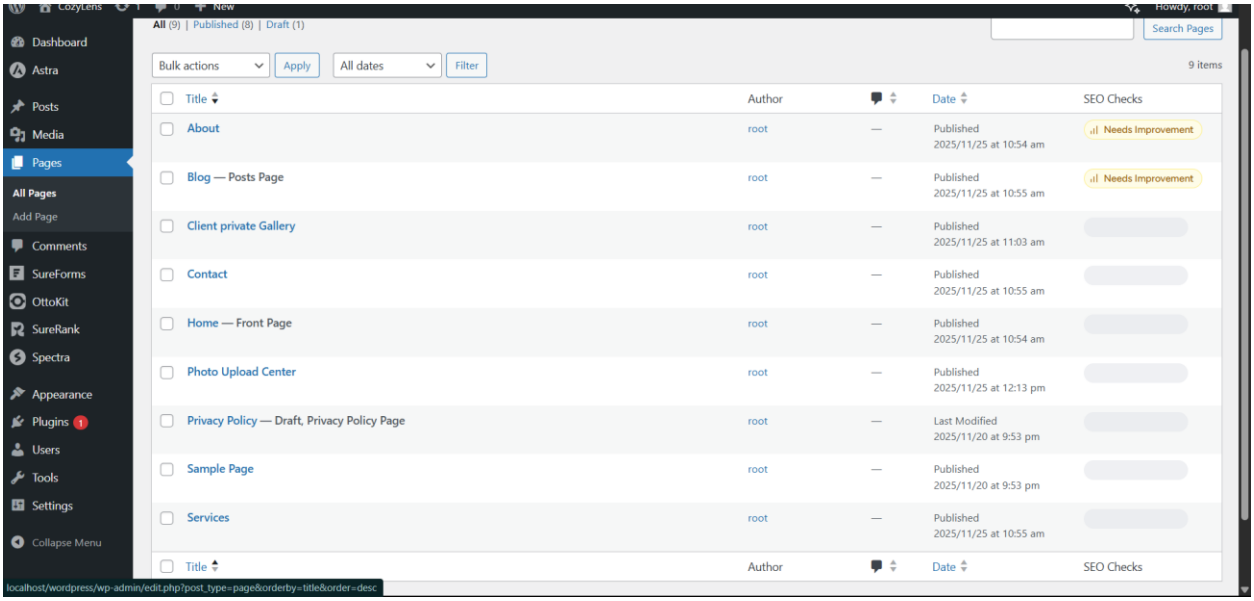


Figure 4_pages

Step 1 Robust User Authentication

Install necessary plugins

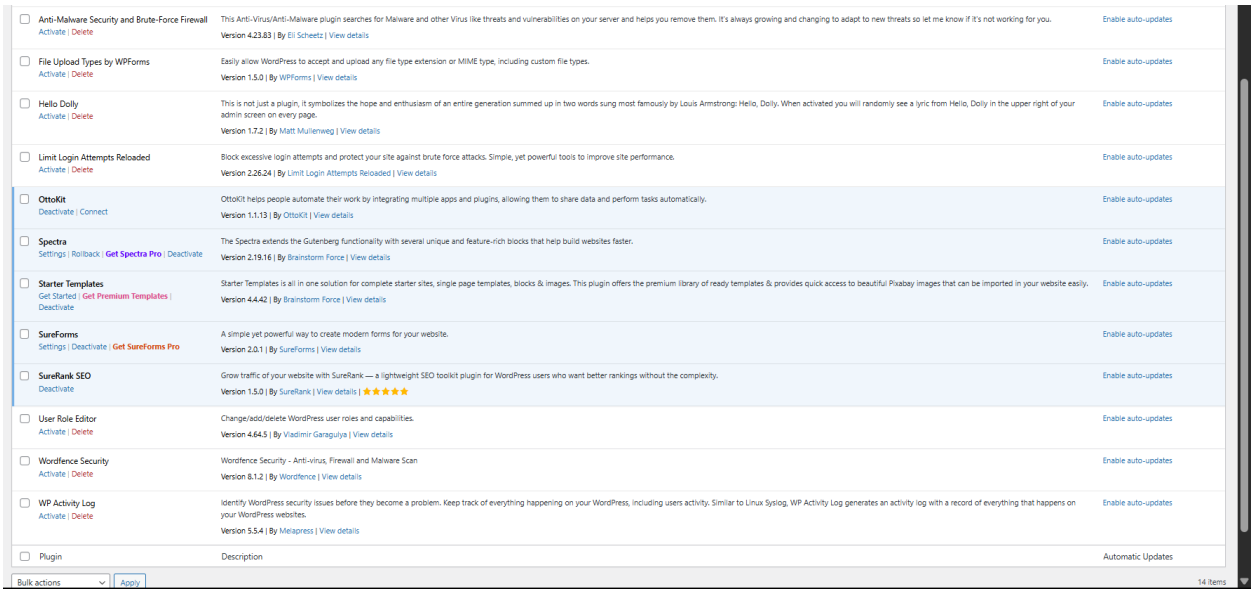


Figure 5_plugins installed

1. **Limit Login Attempts Reloaded** → prevents brute-force attacks
2. **Captcha4wp** → CAPTCHAs on login, registration, contact form

Limit Log in Attempts

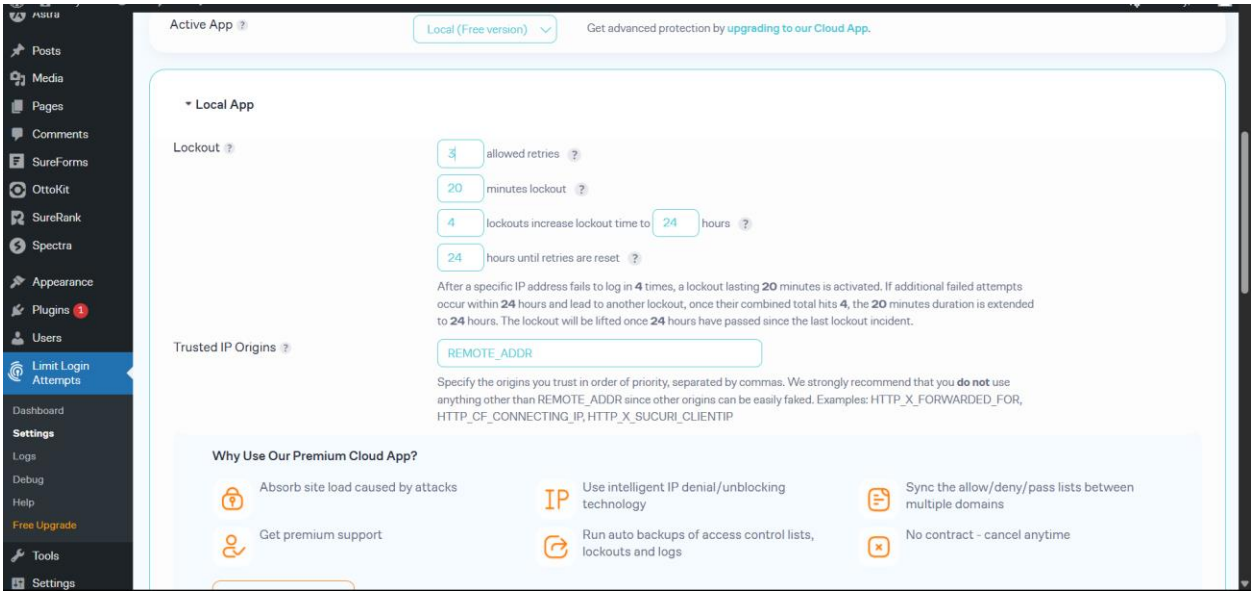


Figure 6_limit login attempts

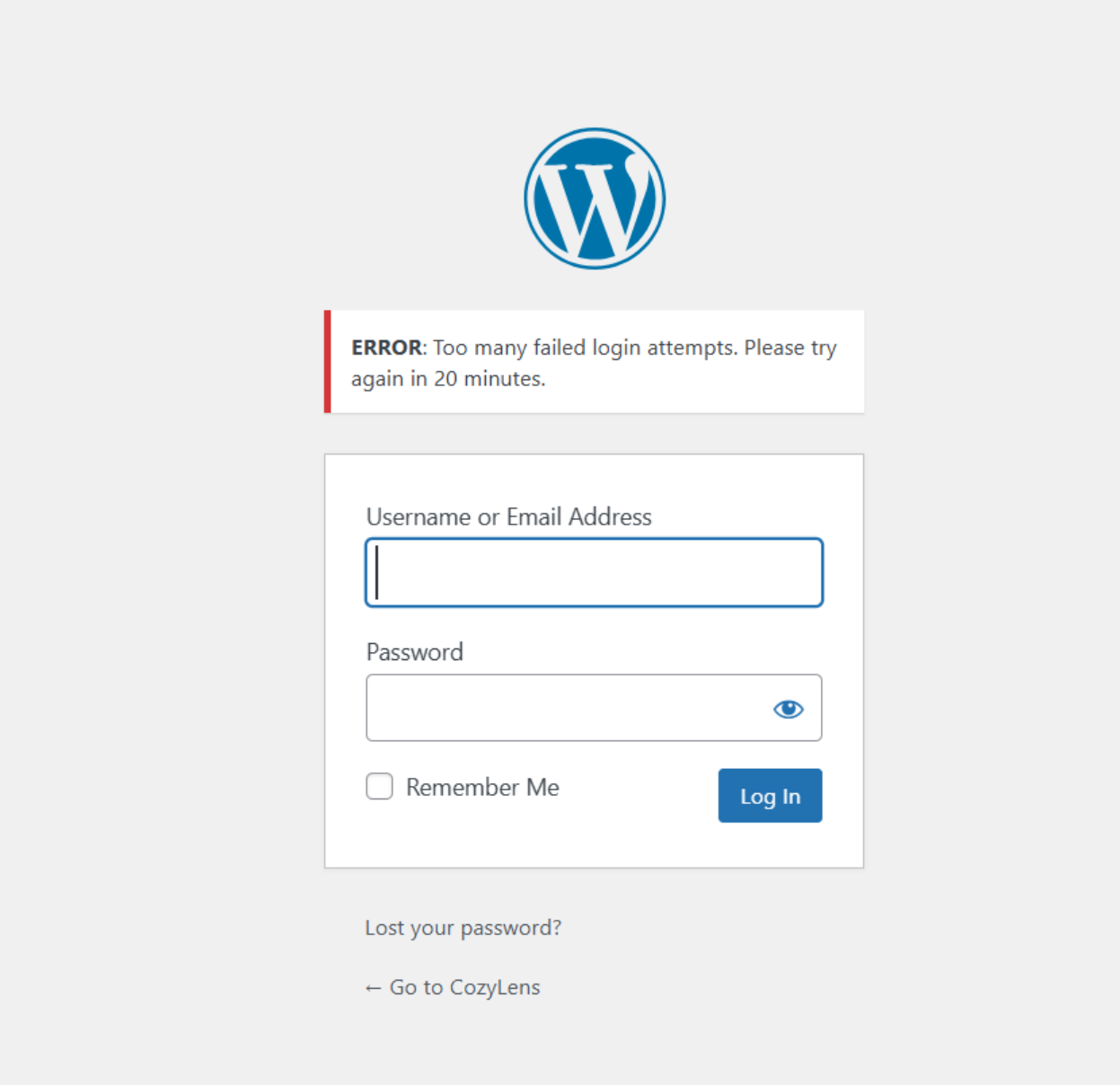


Figure 7_locked after failed attempts

This is brute force protection

Captcha

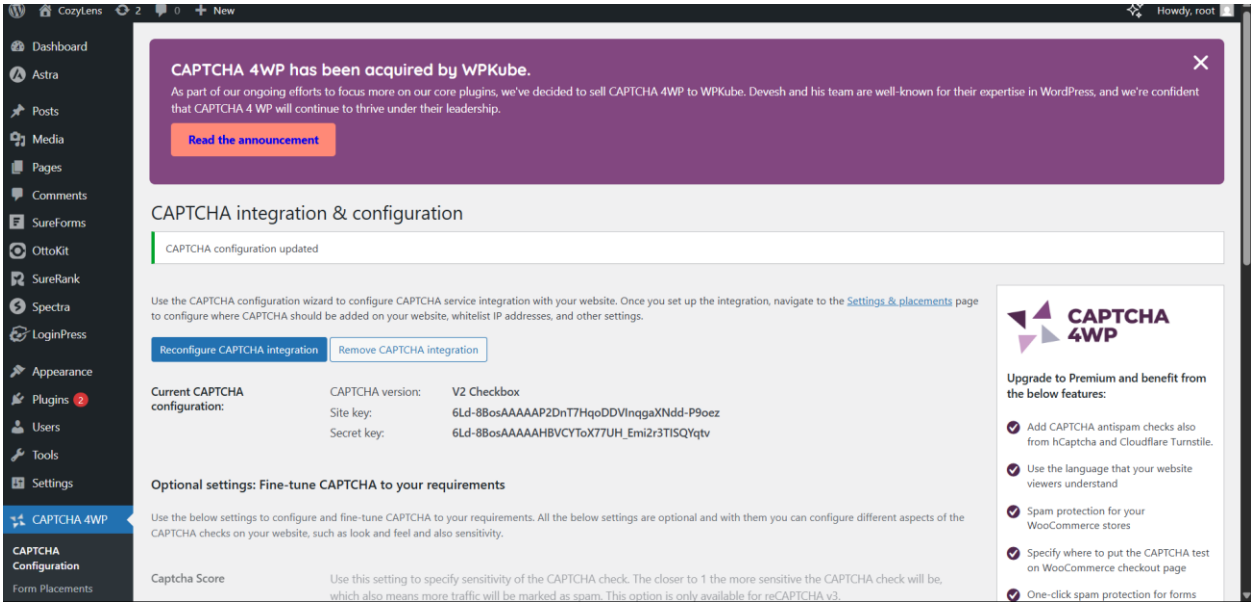


Figure 8_Captcha 4wp

Go to https://www.google.com/recaptcha/admin/create

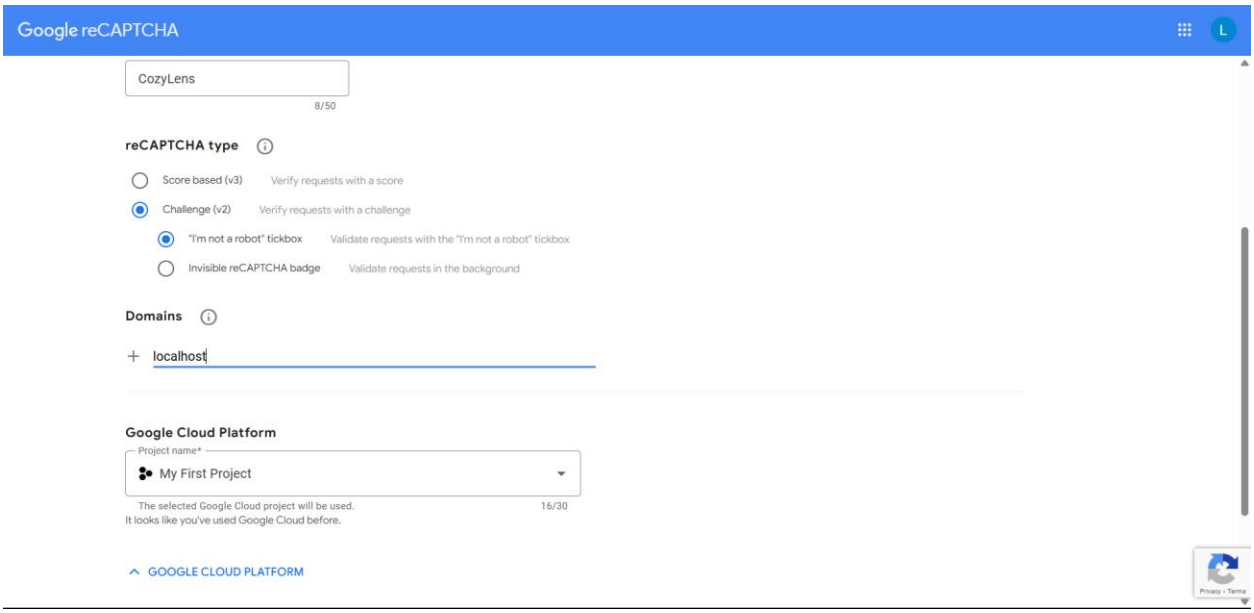


Figure 9_creating new key

Site key: 6Ld-8BosAAAAAP2DnT7HqoDDVInqgaXNdd-P9oez

Secret key: 6Ld-8BosAAAAAHBVCYToX77UH_Emi2r3TISQYqtv

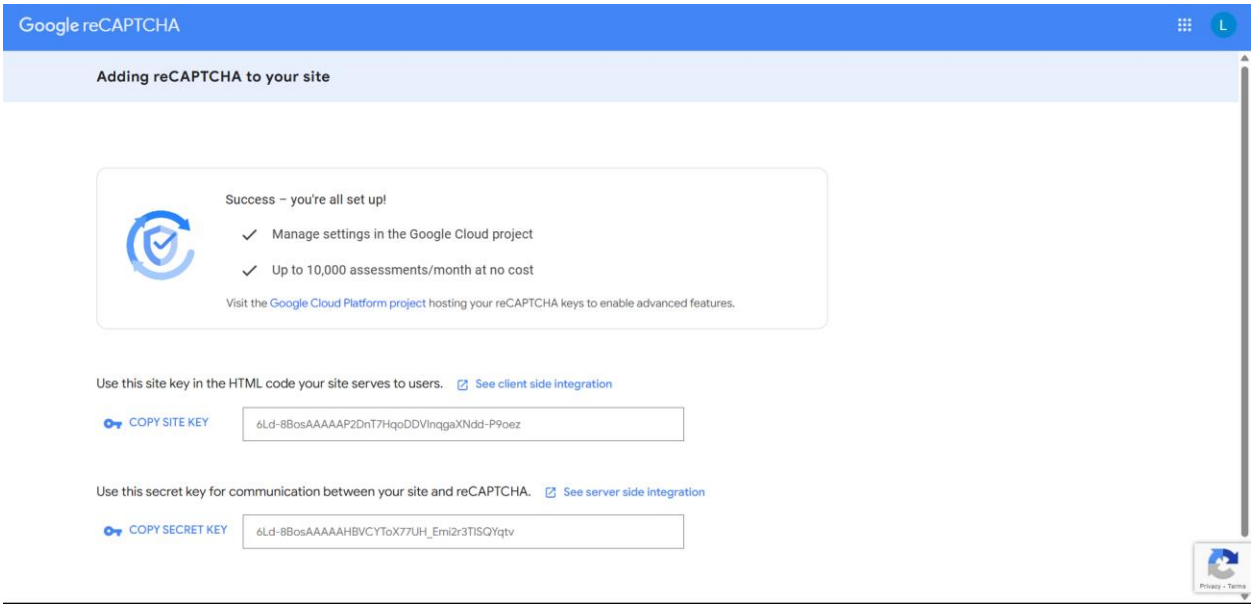


Figure 10_Integration keys

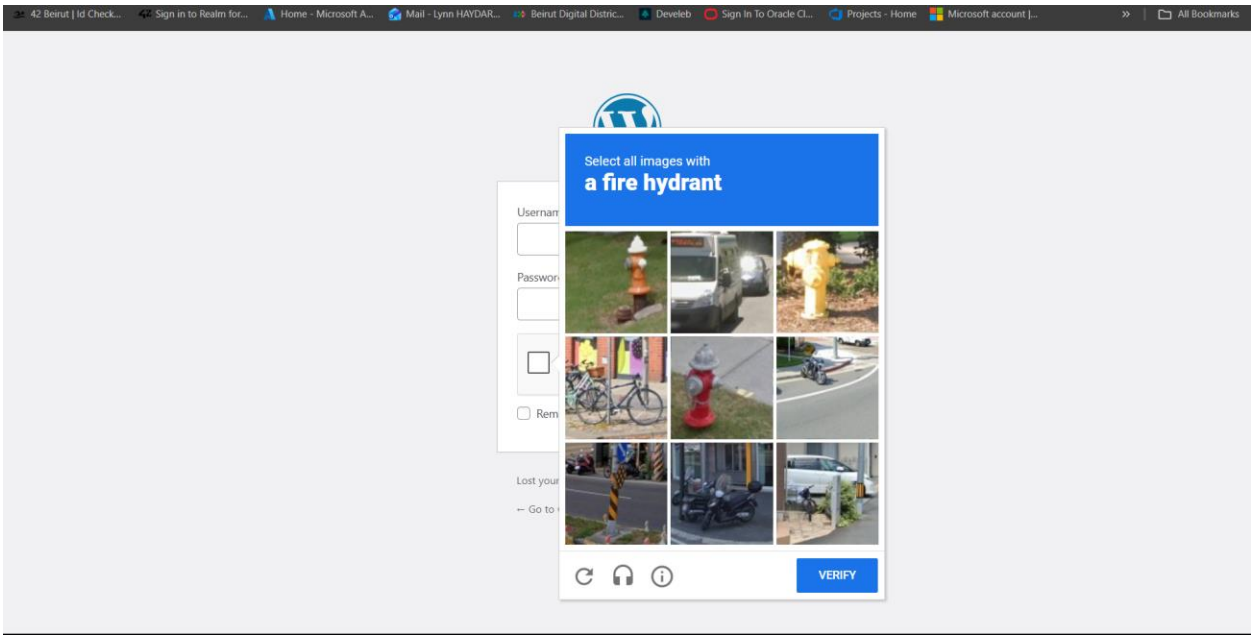


Figure 11_checking results

This block automated login bots

User Registration

Go to settings-> general-> check anyone can register

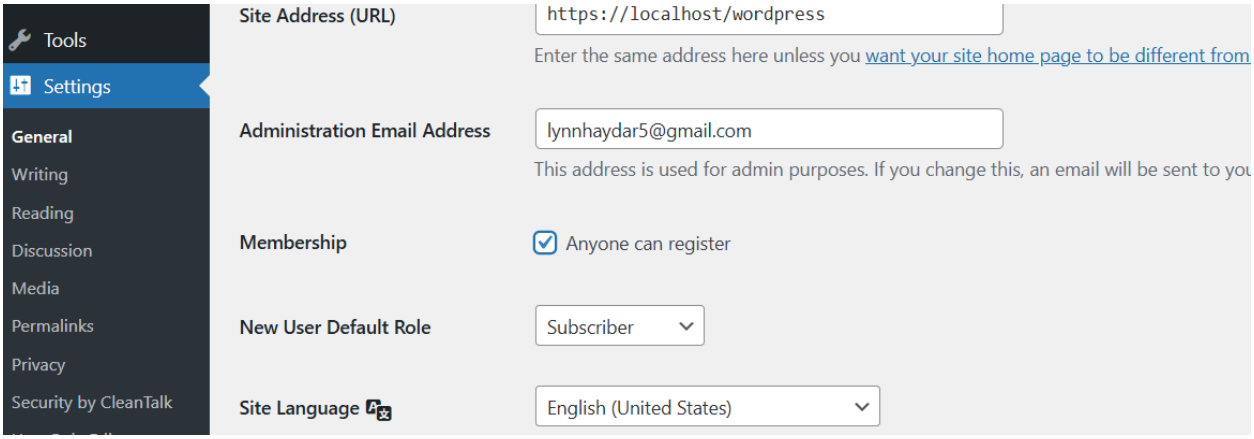


Figure 12_checking

Now download this plugin

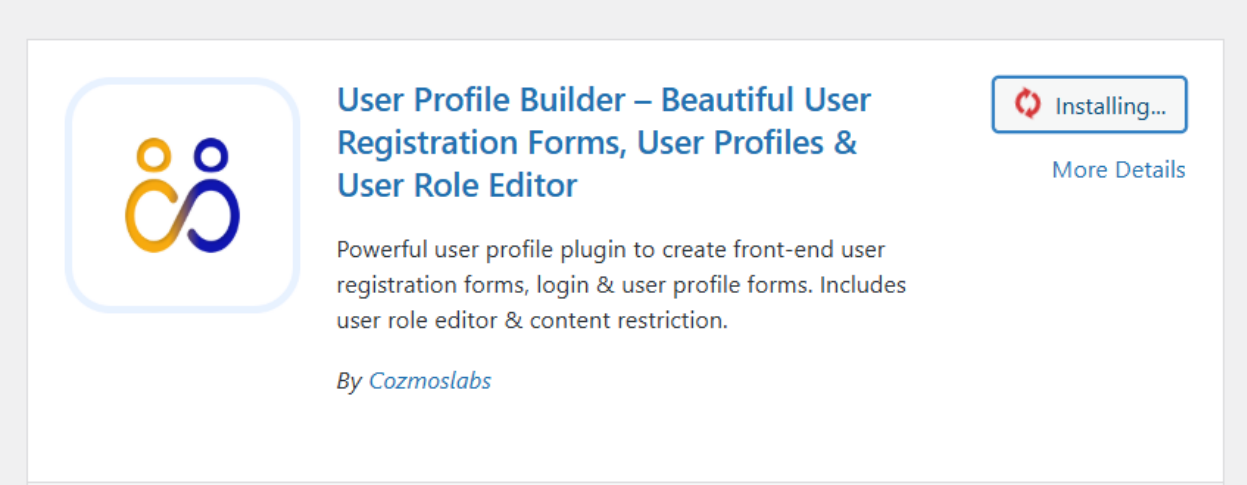


Figure 13_downloading plugin

Test here: https://localhost/wordpress/signip/

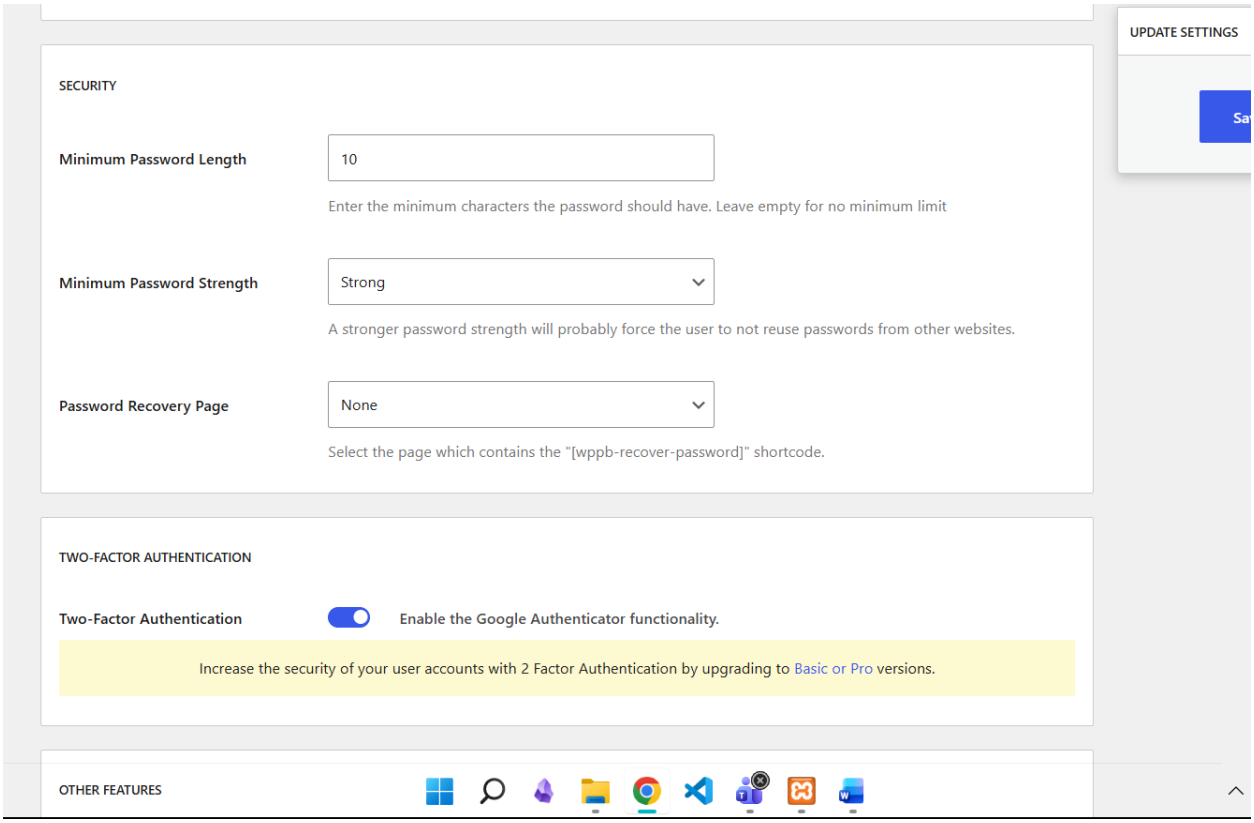


Figure 14_check necessary settings

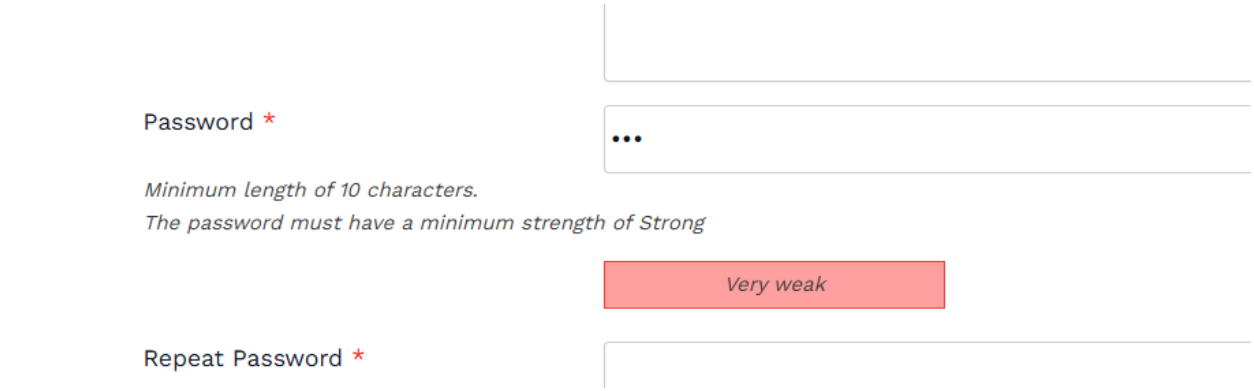


Figure 15_123

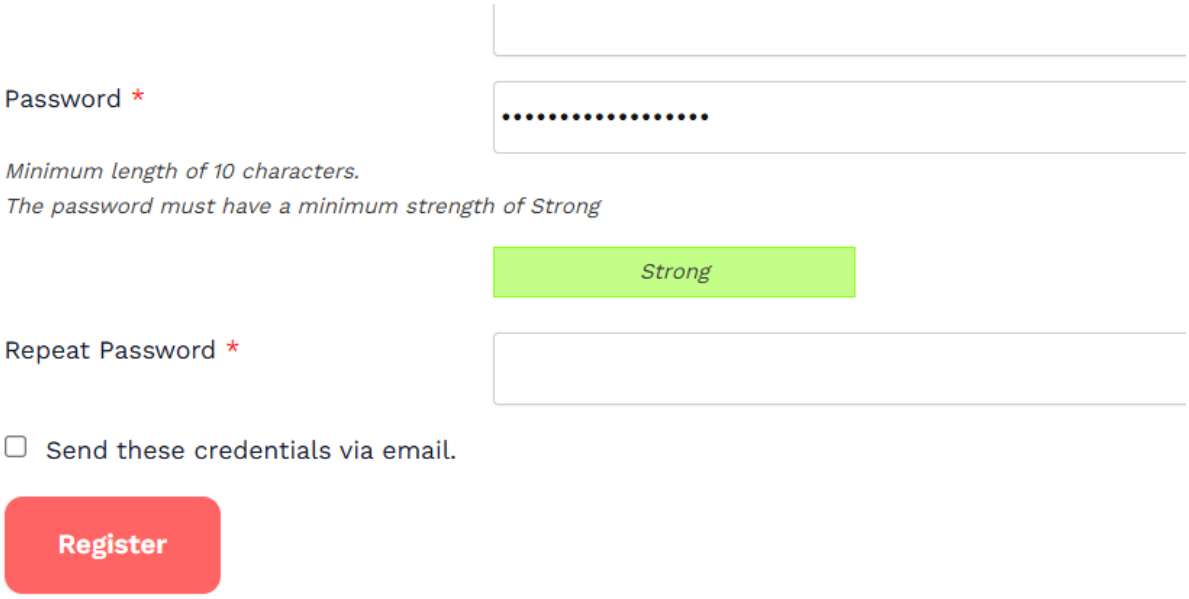


Figure 16_strong

Step 2 Access Control

Download plugin

Download plugin **User Role Editor** → create custom roles

Go to users -> User Role Editor -> add role

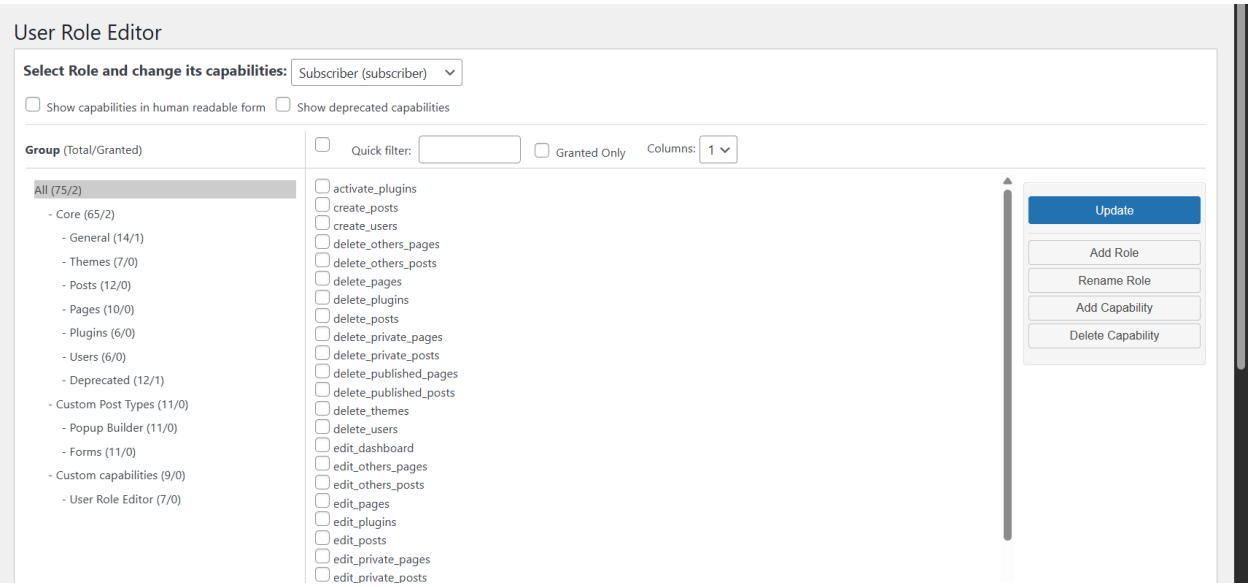


Figure 17_add role

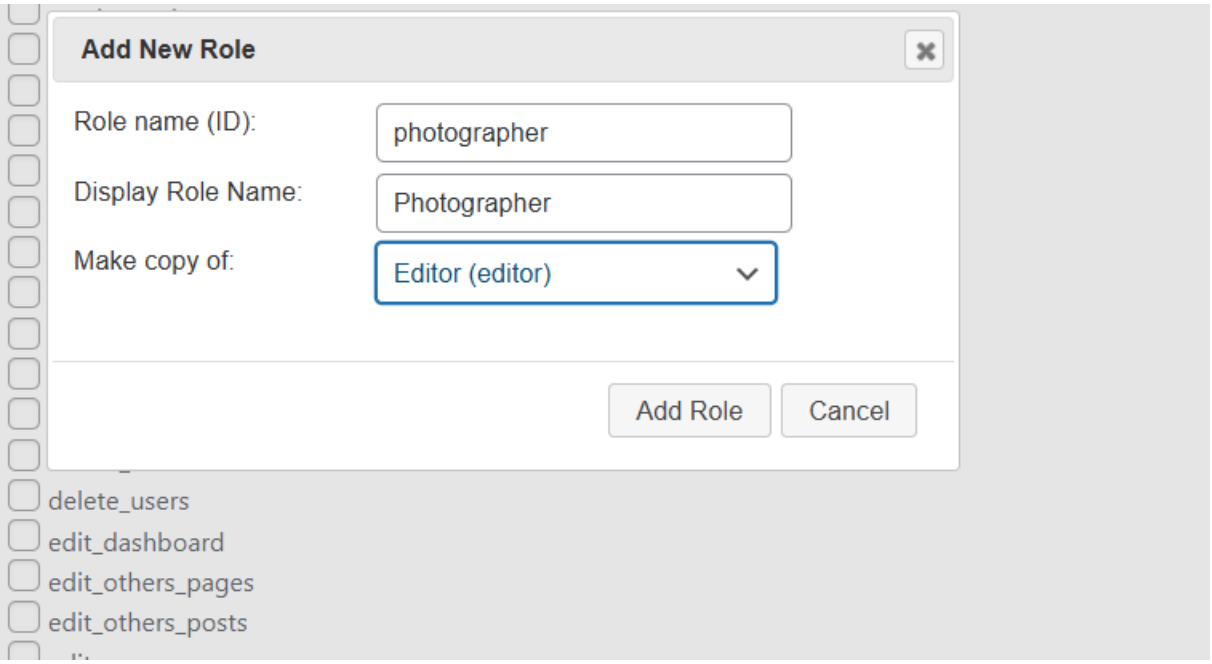


Figure 18_photographer role

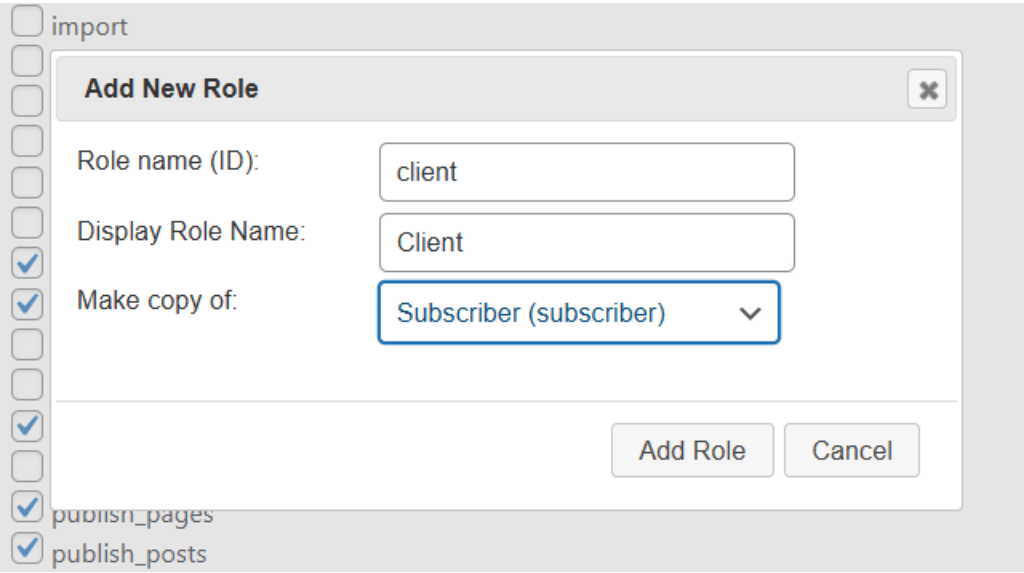


Figure 19_subscriber

Photographer: upload photos

Client: access the private gallery

Now we need test accounts, so we add sample users

Adding sample users

Go to users -> add user

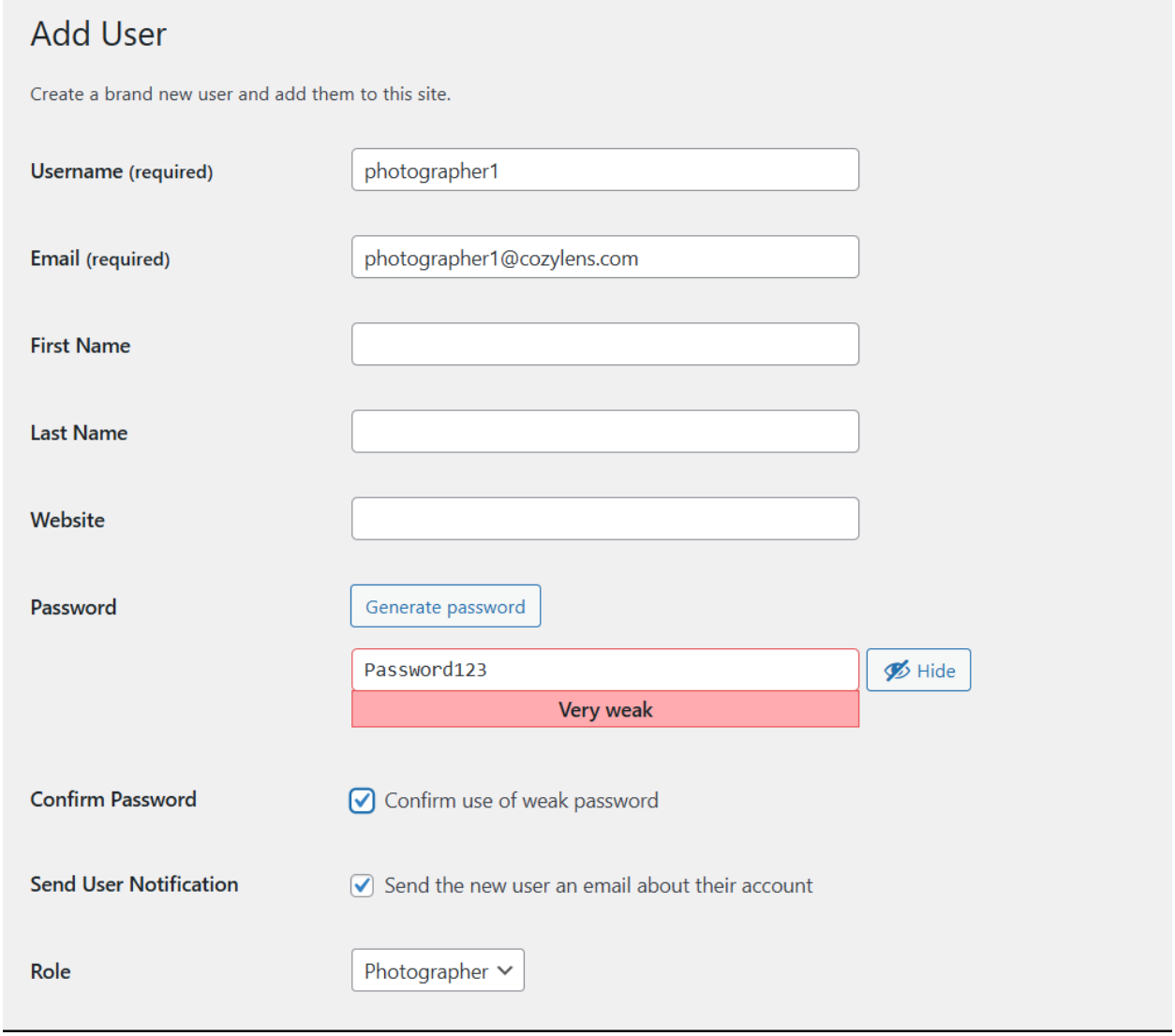


Figure 20_adding new user with role photographer

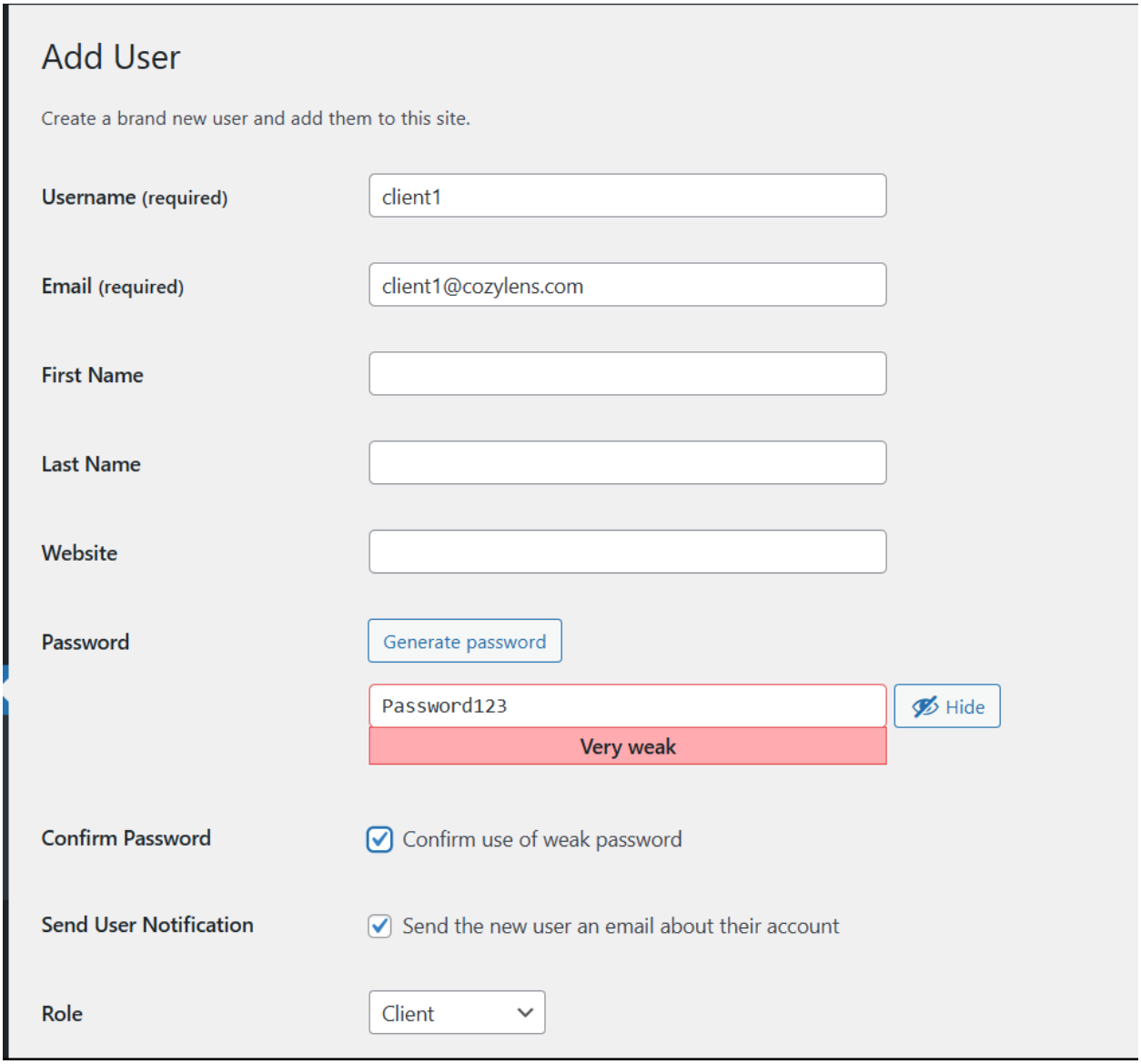


Figure 21_adding new user with role client

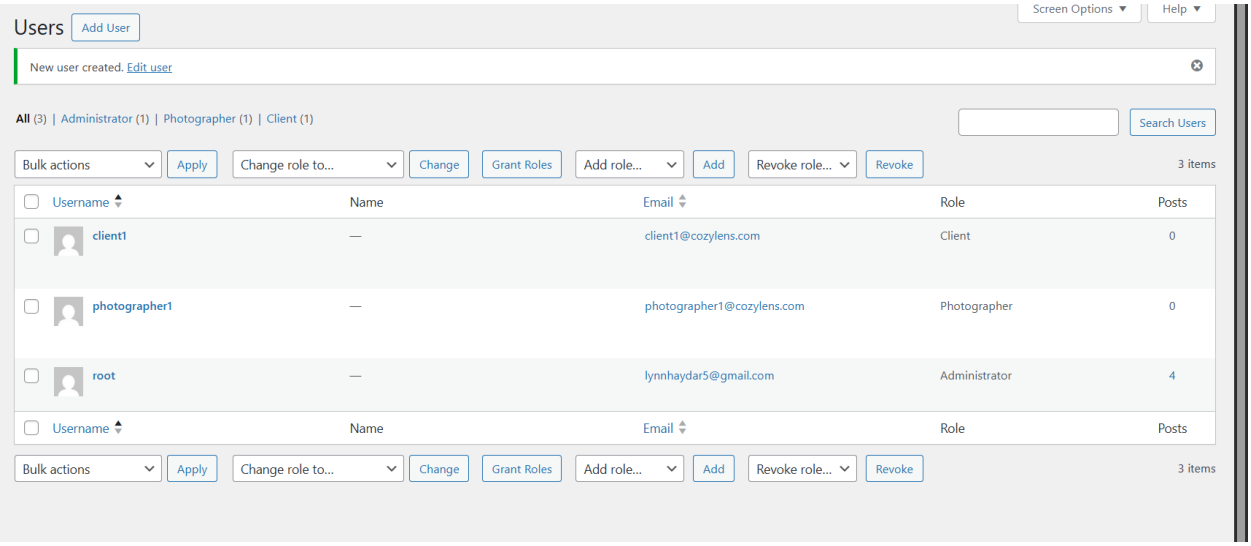


Figure 22_users created

Restrict Pages Based on Roles

Go to the page “Client Private Gallery” and limit access

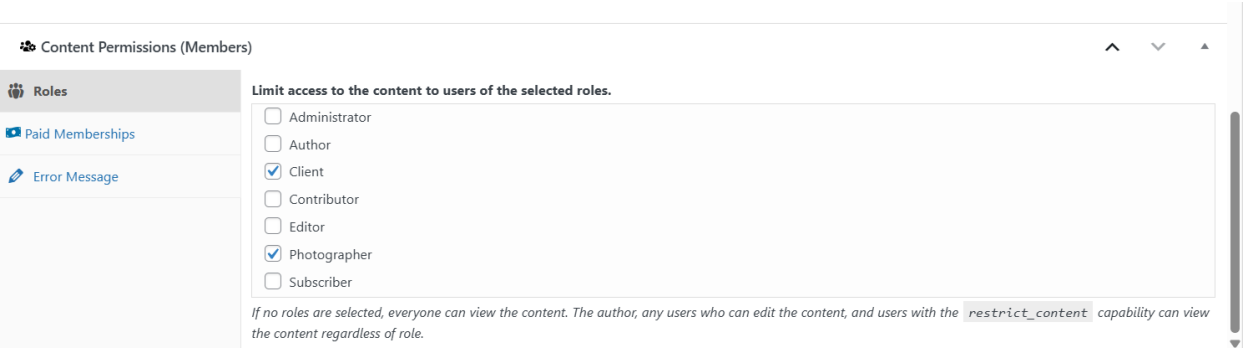


Figure 23_access roles client private gallery

Do the same for the “Photo Upload Center” page

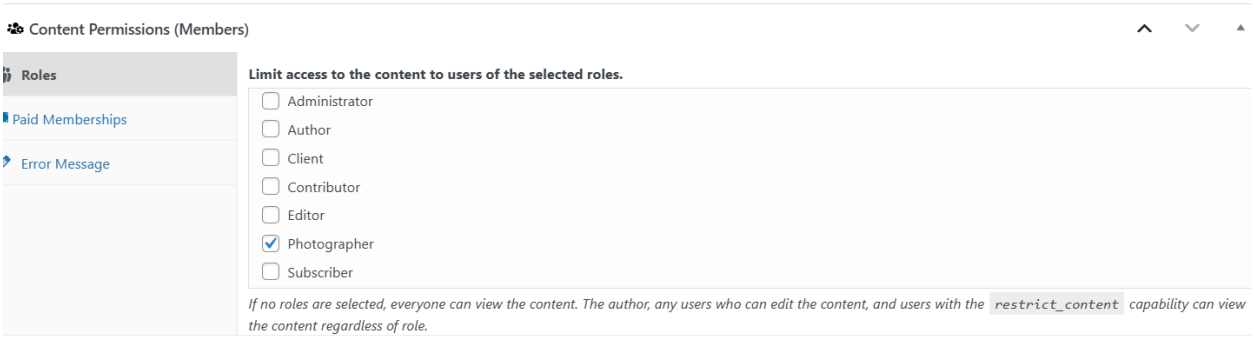


Figure 24_roles photo upload center

Redirecting to login

Go to Apparencies -> Theme File Editor -> functions.php -> add these functions

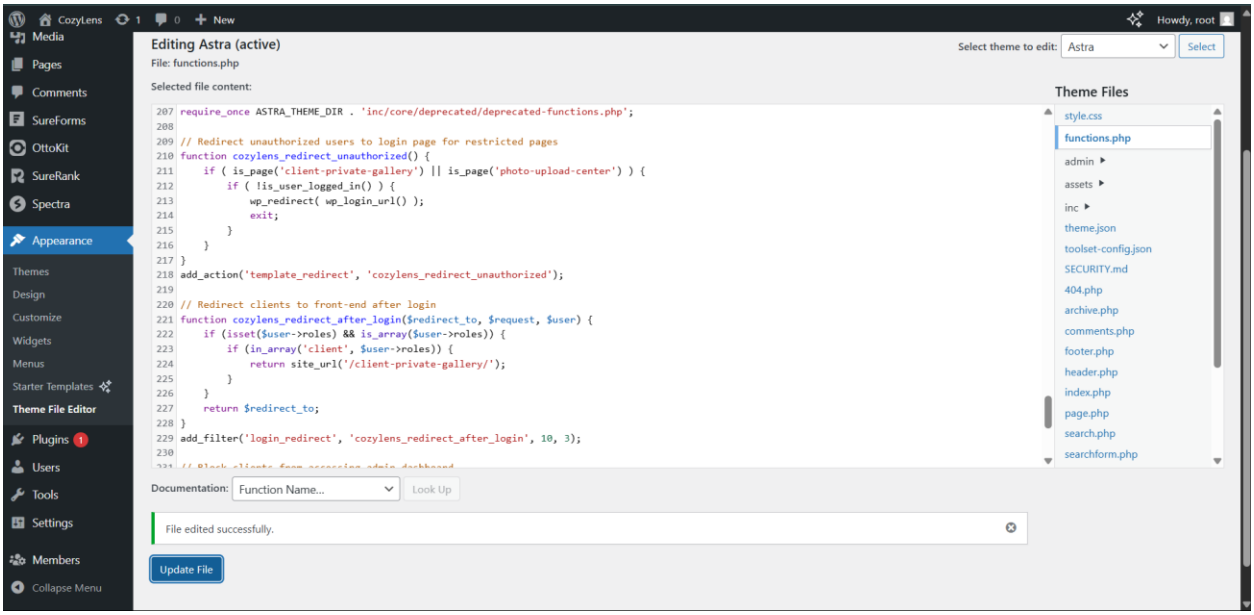


Figure 25_functions.php

```
// Redirect unauthorized users to login page for restricted pages
function cozylens_redirect_unauthorized() {
    if ( is_page('client-private-gallery') || is_page('photo-upload-center') ) {
        if ( !is_user_logged_in() ) {
            wp_redirect( wp_login_url() );
            exit;
        }
    }
}
add_action('template_redirect', 'cozylens_redirect_unauthorized');

// Redirect clients to front-end after login
function cozylens_redirect_after_login($redirect_to, $request, $user) {
```

```
    if (isset($user->roles) && is_array($user->roles)) {
        if (in_array('client', $user->roles)) {
            return site_url('/client-private-gallery/');
        }
    }
    return $redirect_to;
}
add_filter('login_redirect', 'cozylens_redirect_after_login', 10, 3);

// Block clients from accessing admin dashboard
function cozylens_block_wp_admin_access() {
    if (is_user_logged_in() && current_user_can('client') && is_admin()) {
        wp_redirect(site_url('/client-private-gallery/'));
        exit;
    }
}
add_action('init', 'cozylens_block_wp_admin_access');

// Hide admin bar for clients
add_filter('show_admin_bar', function($show){
    if (current_user_can('client')) {
        return false;
    }
    return $show;
});

// Redirect photographers to upload center after login
function cozylens_redirect_photographer($redirect_to, $request, $user) {
    if (isset($user->roles) && is_array($user->roles)) {
        if (in_array('photographer', $user->roles)) {
            return site_url('/photo-upload-center/');
        }
    }
    return $redirect_to;
}
add_filter('login_redirect', 'cozylens_redirect_photographer', 11, 3);

// Block photographers from accessing wp-admin
function cozylens_block_photographer_admin() {
    if (is_user_logged_in() && current_user_can('photographer') && is_admin()) {
        wp_redirect(site_url('/photo-upload-center/'));
        exit;
    }
}
add_action('init', 'cozylens_block_photographer_admin');

```

To test, open incognito mode, go to http://localhost/wordpress/client-private-gallery

You will be redirected to a login page

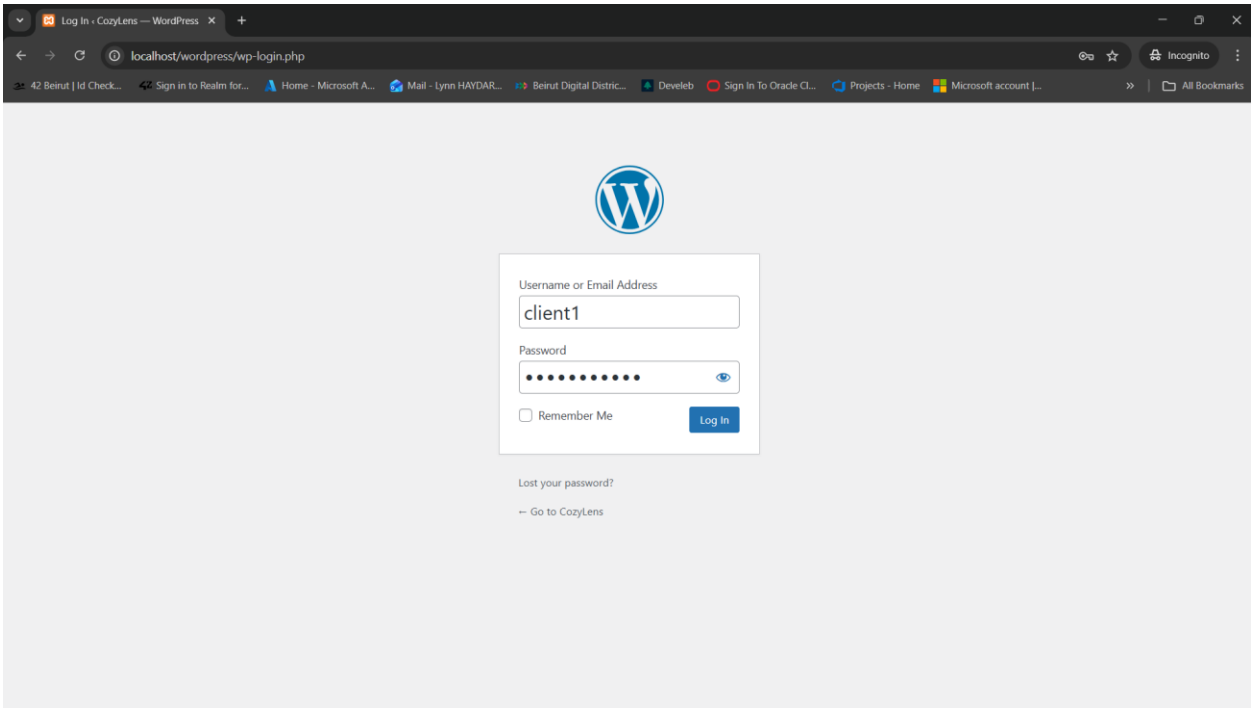


Figure 26_redirected to login page

Log in as client 1

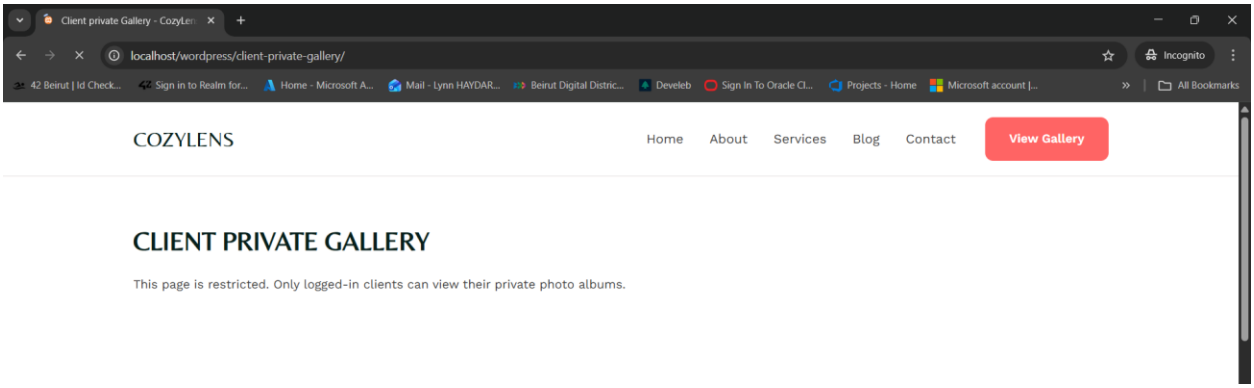


Figure 27_log in successful

Do the same for the photographer

Go to: http://localhost/wordpress/photo-upload-center

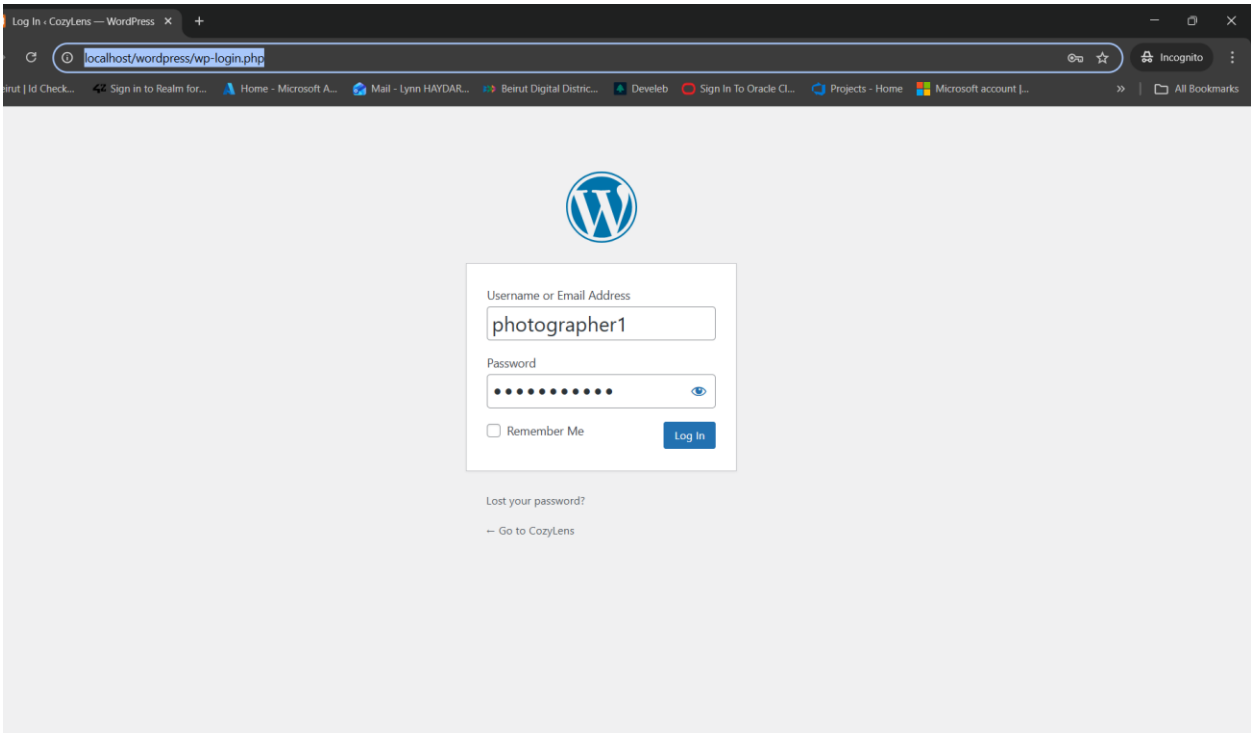


Figure 28_redirected to login

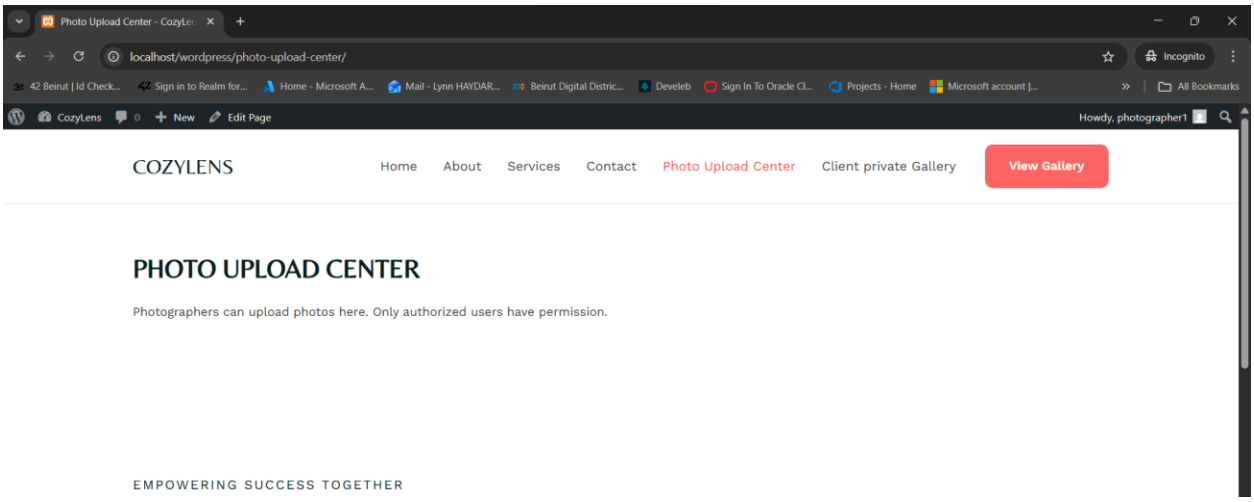


Figure 29_log in successful

This is for the unauthorized access requirement

STEP 3_ Data Encryption and SSL

An SSL (Secure Sockets Layer) certificate is a digital certificate that verifies a website's identity and enables an encrypted connection between a server and a browser. When a website has a

valid SSL certificate, its URL will display https:// and a padlock icon in the browser's address bar, indicating a secure and trustworthy connection.

Install plugin: “Really Simple SSL”.

Figure 30_installing plugin

Visit https://localhost/wordpress

This window appears, but it’s okay it means that the plugin works

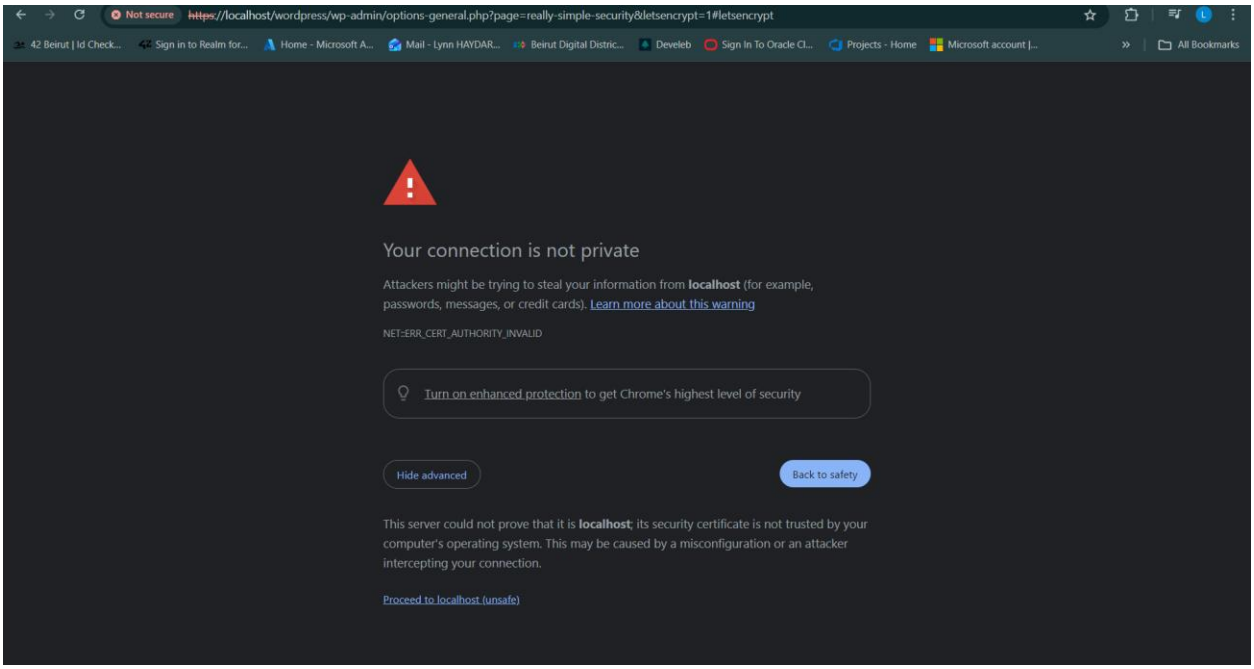


Figure 31_ssl works

Figure 32_https

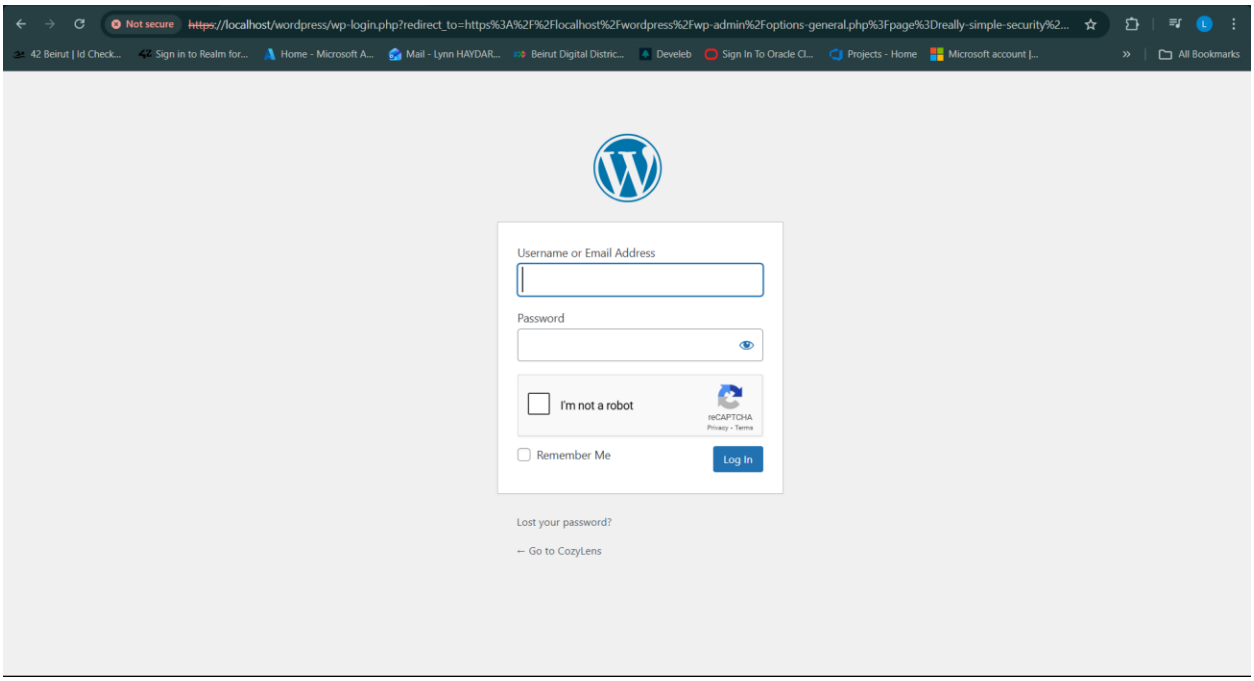


Figure 33_proceding

Step 4 SQL Injection Prevention

Test with this url: http://localhost/wordpress/?rest_route=/wp/v2/users&per_page=abc' OR 1=1—

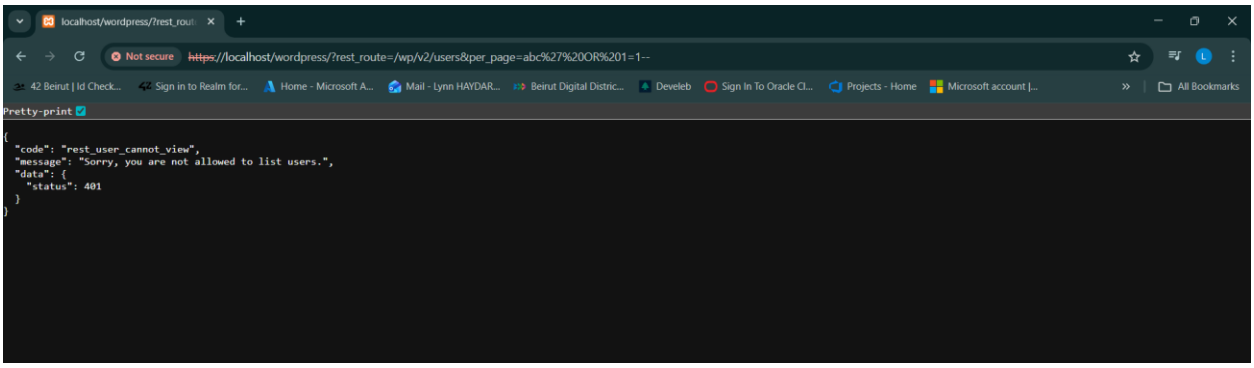


Figure 34_testing

And http://localhost/wordpress/?p=1 UNION SELECT md5(1)—

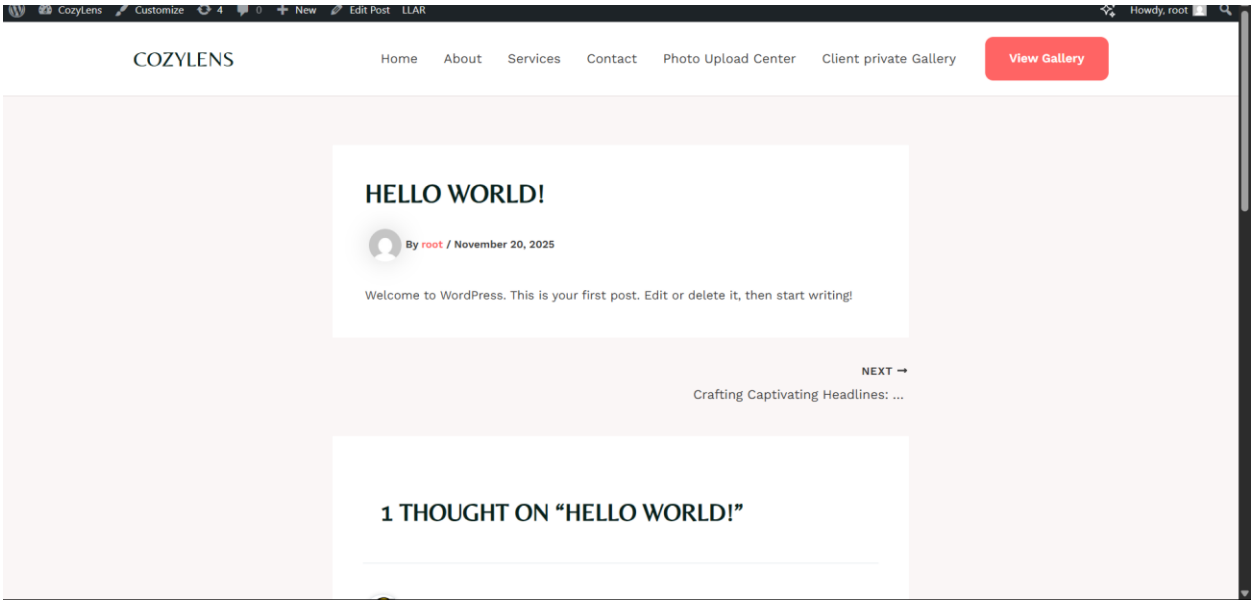


Figure 35_test successful

Website didn't break and show sql data.

Step 5 Session Management

Install this plugin

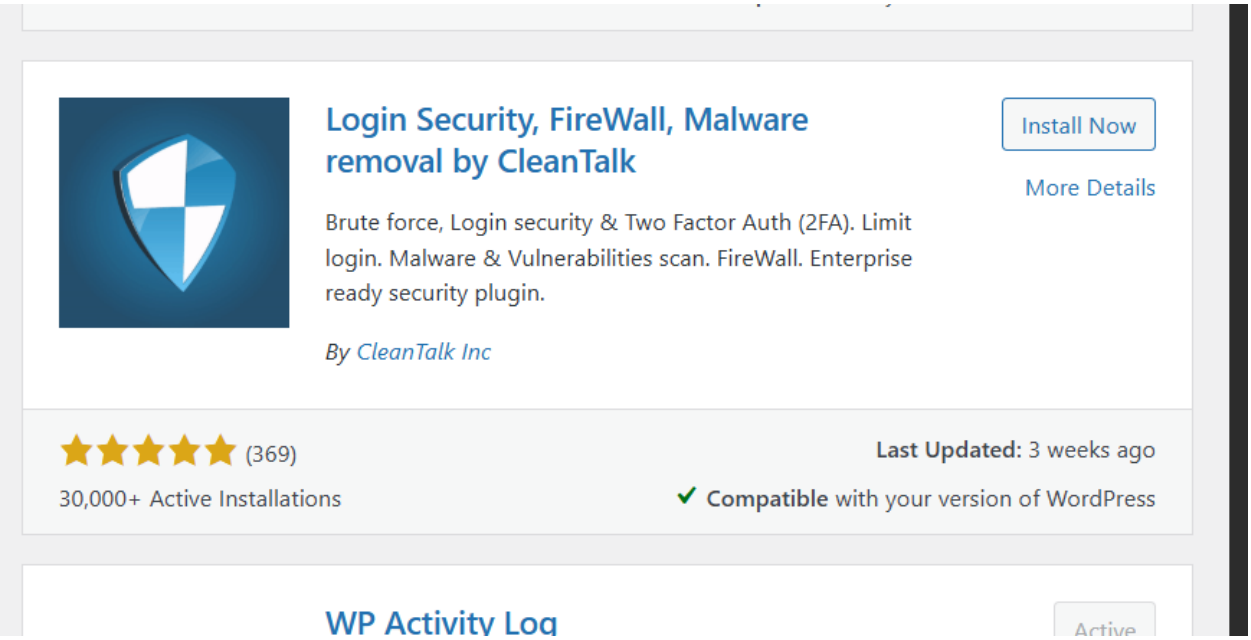


Figure 36_installing plugin

Get manual access key

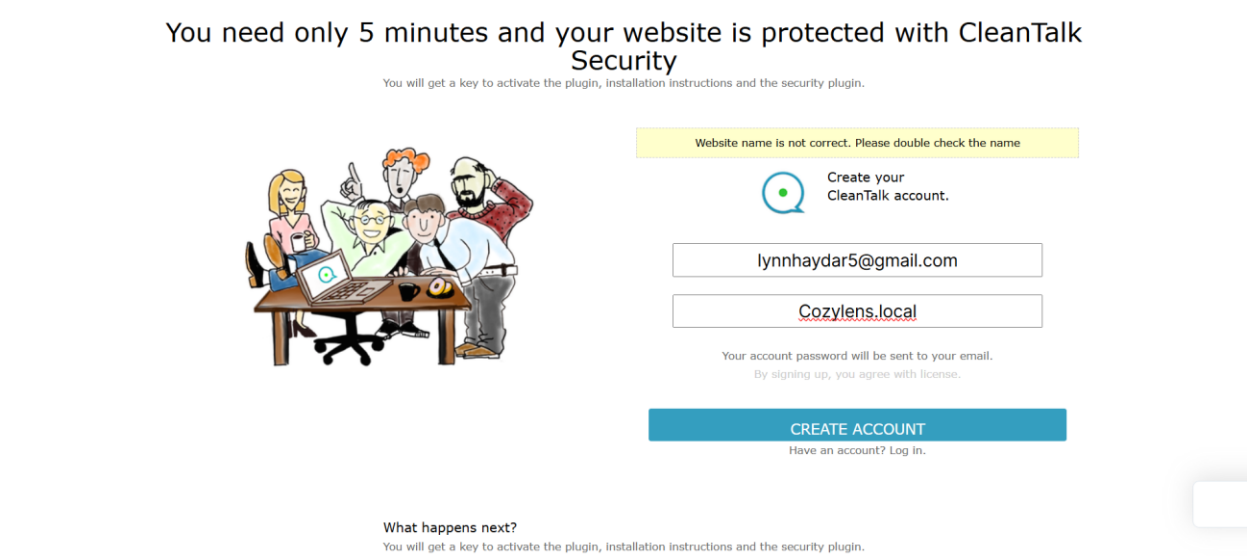


Figure 37_getting access key

And paste it

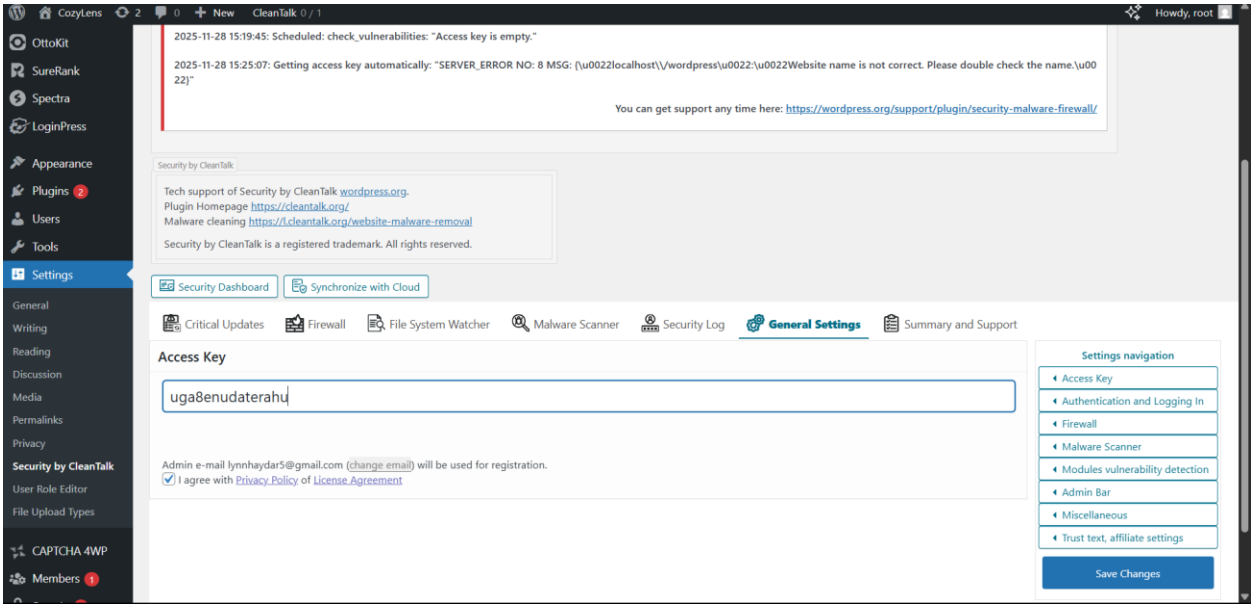


Figure 38_pasting access key

Modify settings and save

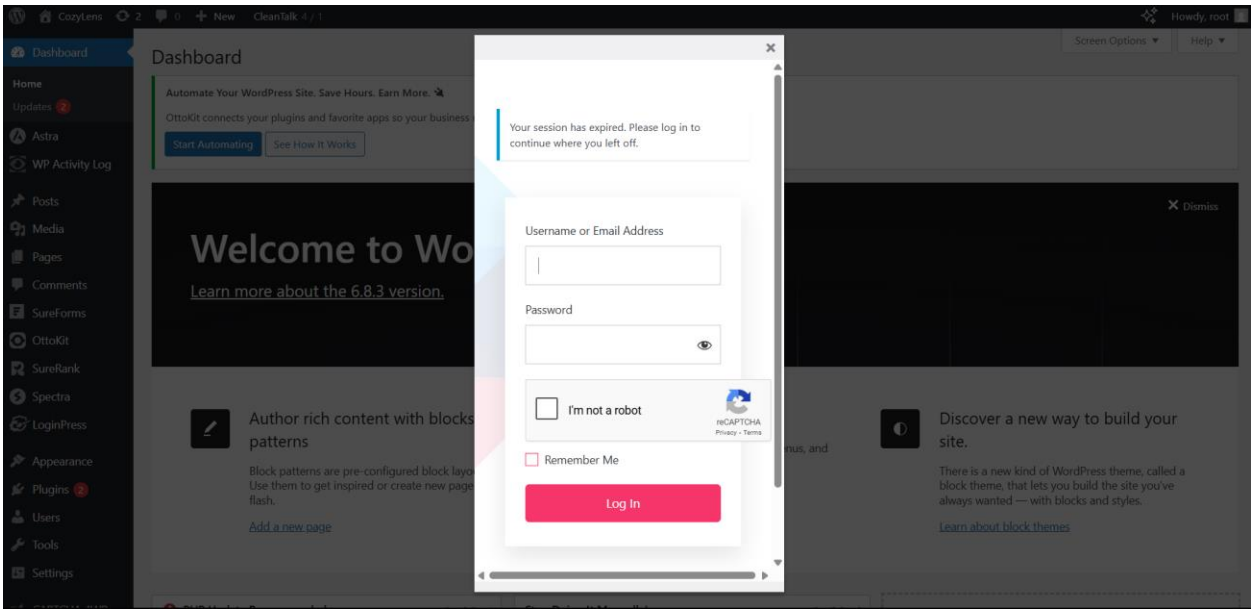


Figure 39_session expired after one minute

STEP 6 File Upload Security (VERY important)

Install this plugin

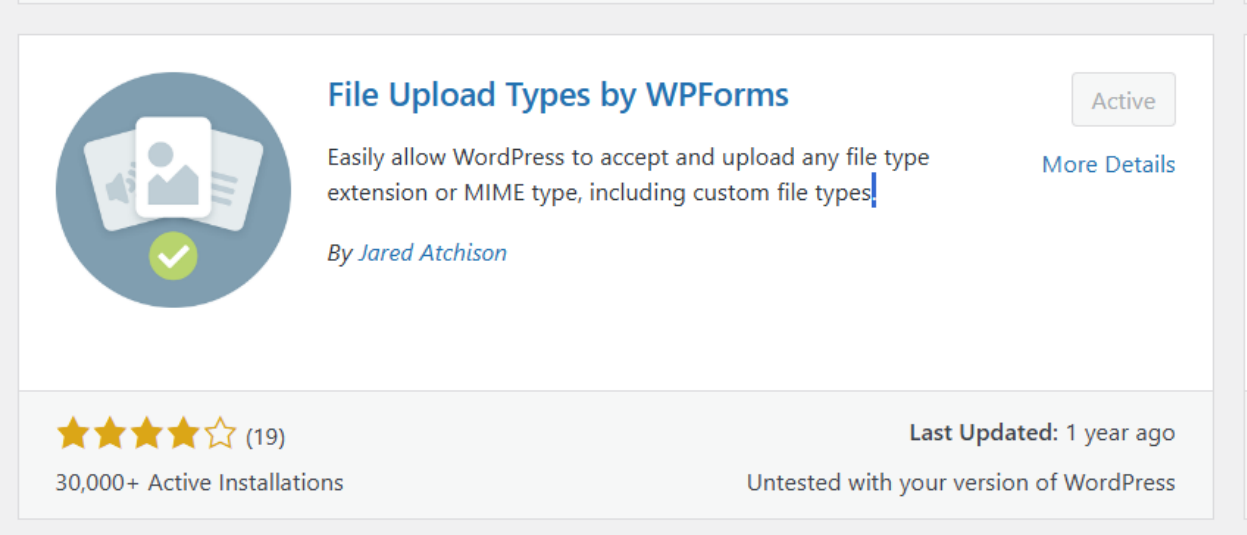


Figure 40_file uploads plugin

We chose to enable these file types:

- jpg
- jpeg
- png
- webp
- gif (optional)
- heic (support iPhone uploads)

PNG, JPG, JPEG, GIF are already allowed by default.

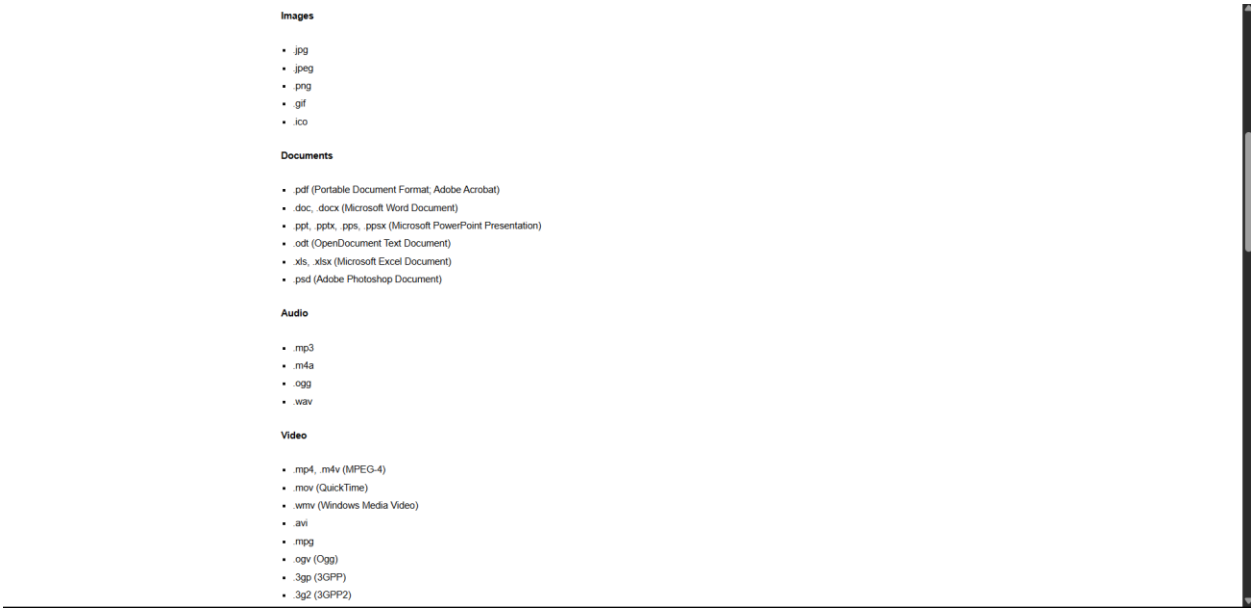


Figure 41_allowed by WordPress

Let's Enable webp and heic

Click on add your custom file type

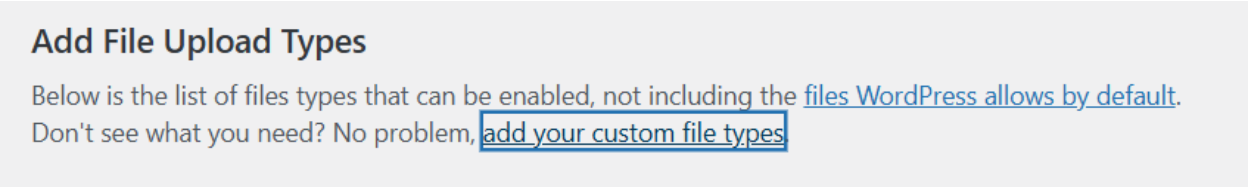


Figure 42_adding custom file type

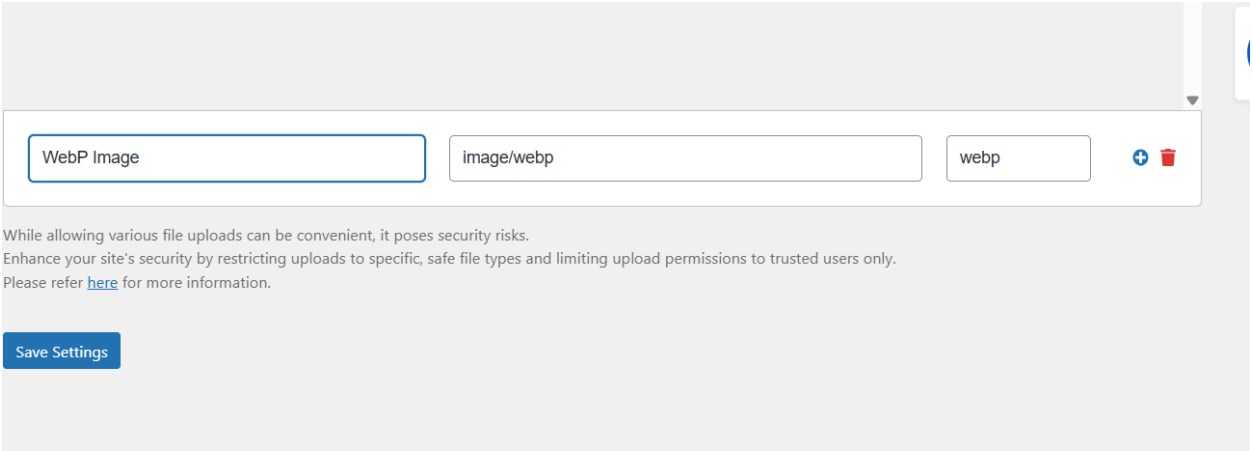


Figure 43_configuring it

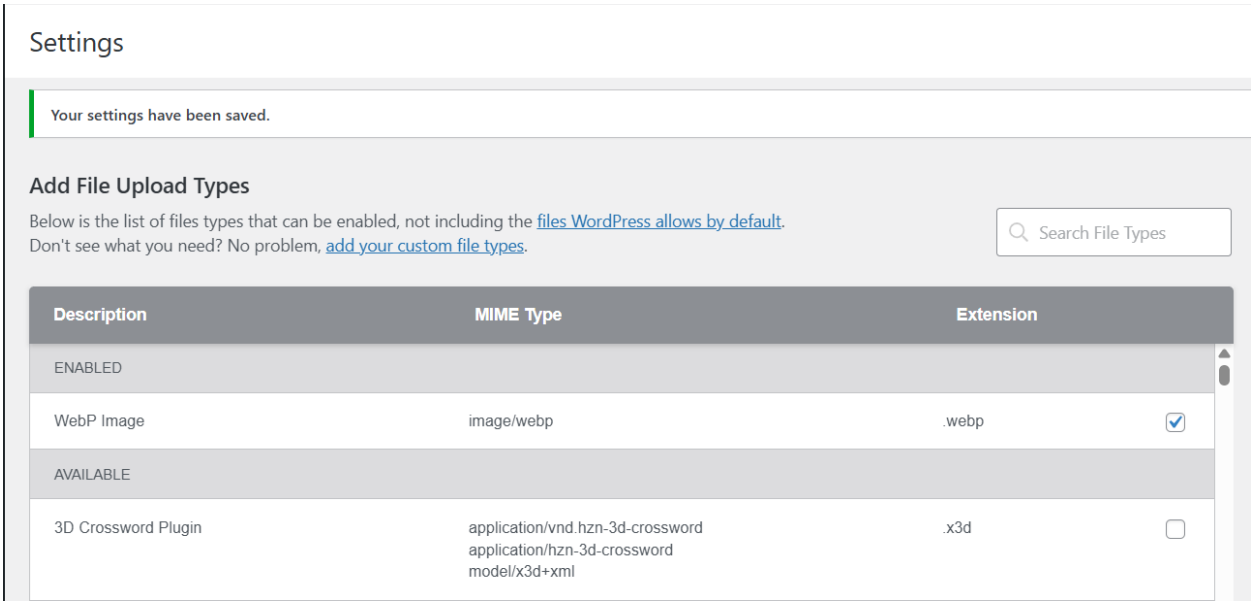


Figure 44_save

Same for heic

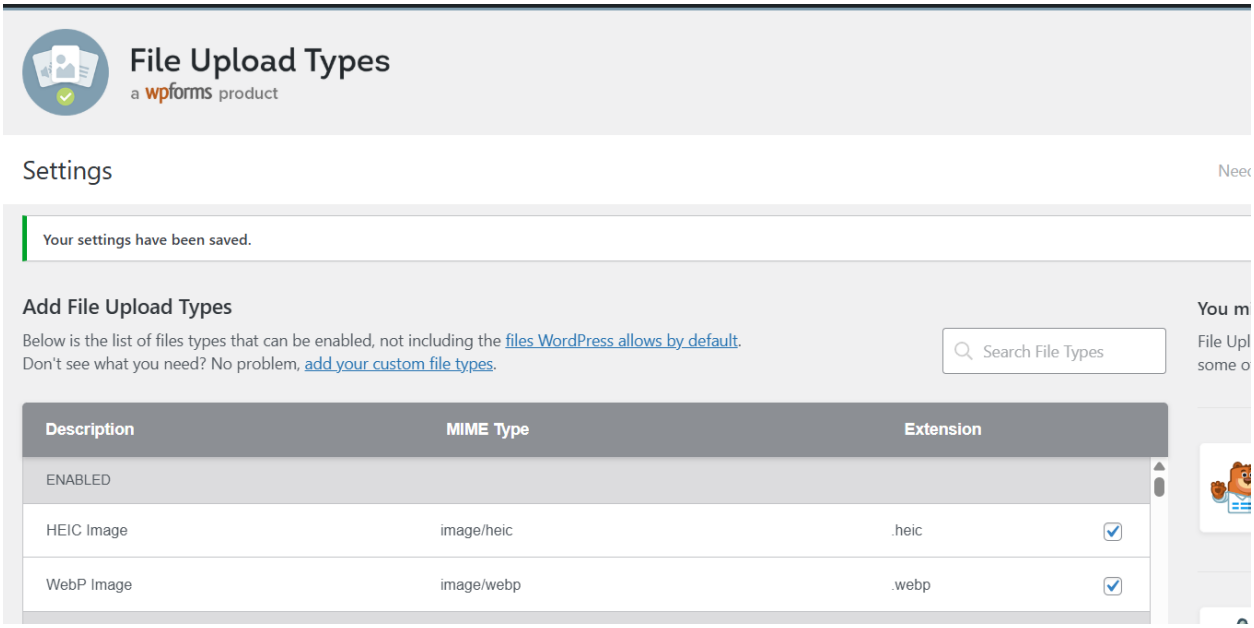


Figure 45_done

To test it go to apparencies, theme file editor -> -> functions.php -> add this functions

```
1.  1. function cozylens_test_upload_form() {
2.  2.     ob_start(); ?>
3.  3.
4.  4. <form method="post" enctype="multipart/form-data">
5.  5.   <label>Select a file to upload:</label><br><br>
6.  6.   <input type="file" name="test_upload" required><br><br>
7.  7.   <input type="submit" name="submit_test_upload" value="Upload File">
8.  8. </form>
```

```php
<?php
if (isset($_POST["submit_test_upload"])) {
    require_once(ABSPATH . 'wp-admin/includes/file.php');

    $upload = wp_handle_upload($_FILES['test_upload'], array('test_form' => false));

    if (isset($upload['error'])) {
        echo "<p style='color:red;'><strong>UPLOAD BLOCKED:</strong> " .
esc_html($upload['error']) . "</p>";
    } else {
        echo "<p style='color:green;'><strong>UPLOAD SUCCESS:</strong> File uploaded to: " .
esc_url($upload['url']) . "</p>";
    }
}

return ob_get_clean();
}
add_shortcode('upload_test', 'cozylens_test_upload_form');

```

This creates a shortcode: [upload_test]

Go to photo upload center page and add a shortcode block and paste it

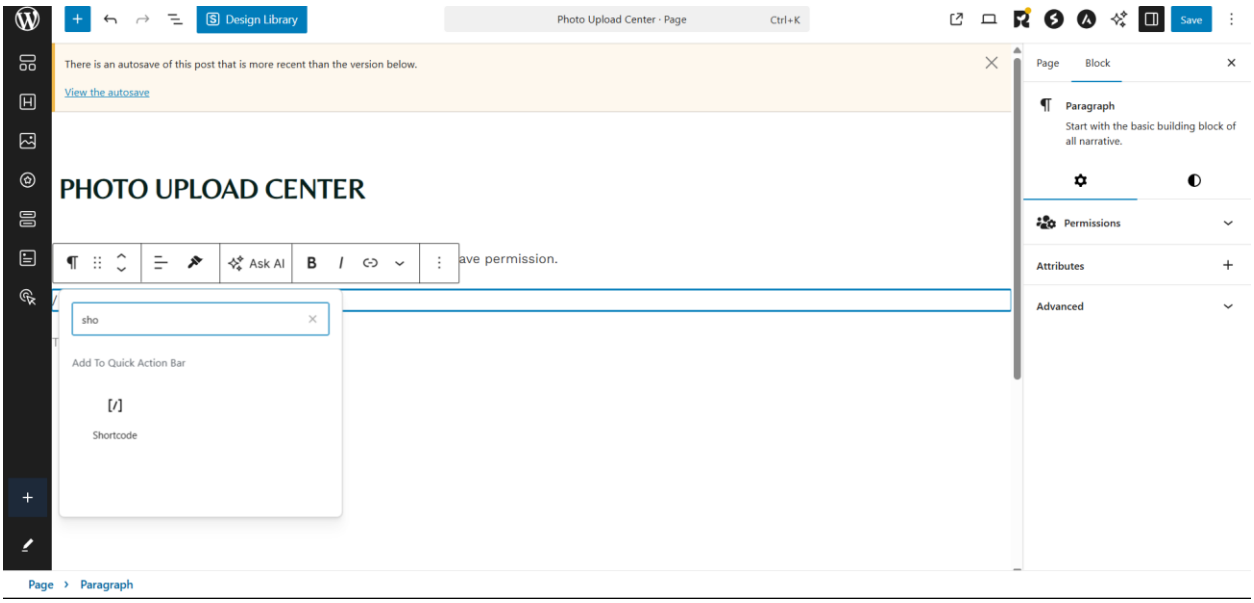


Figure 46_shortcode block

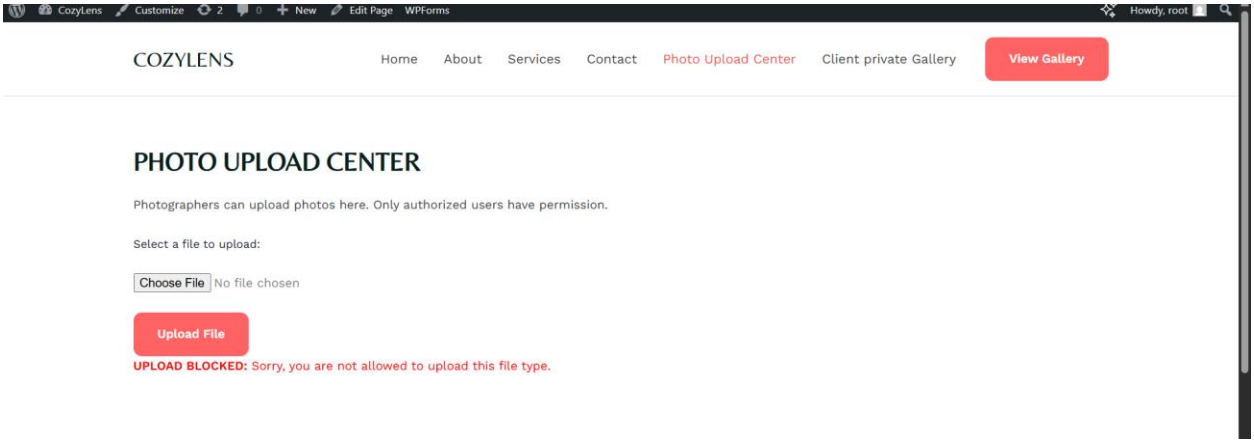


Figure 47_testing

Now for scanning the files

Install this plugin:

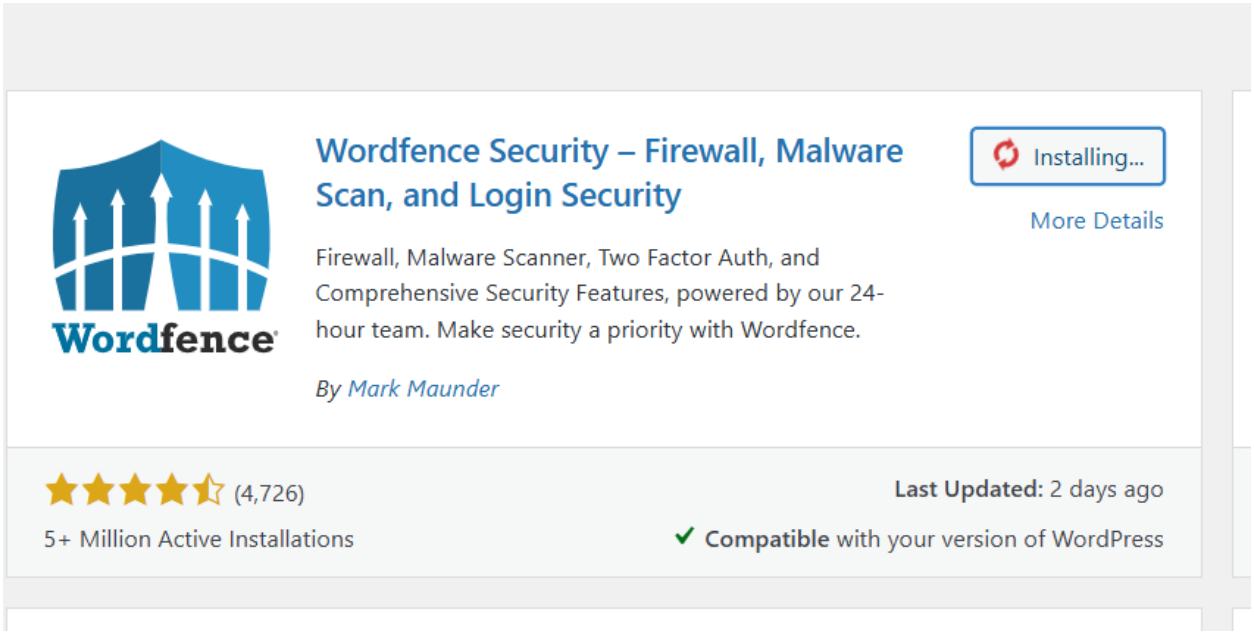


Figure 48_wordfence security installation

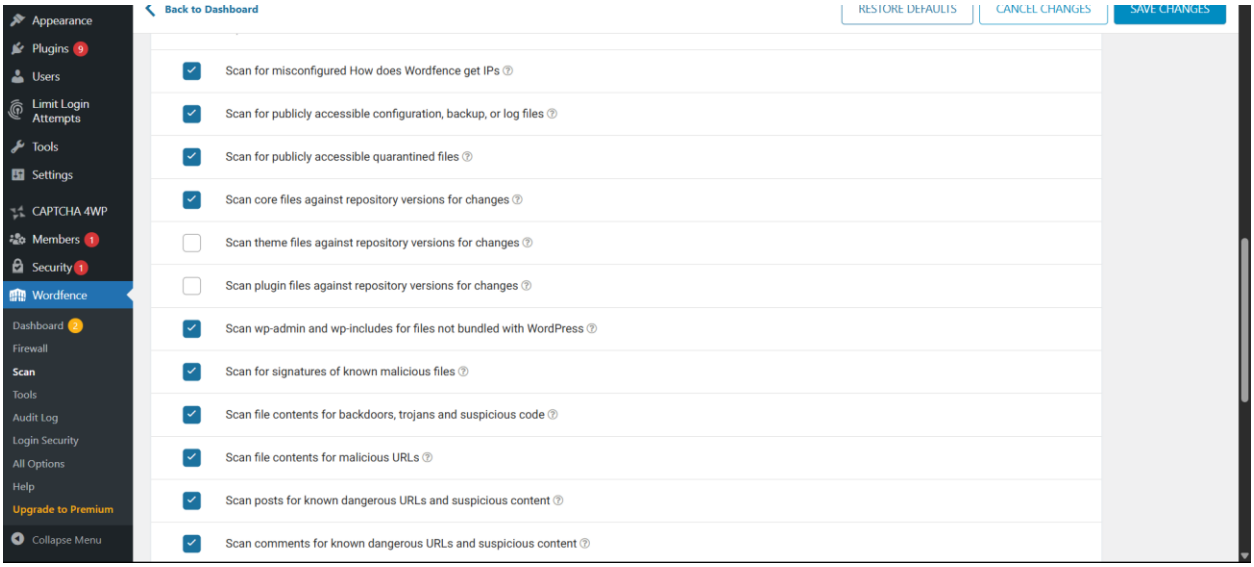


Figure 49_check necessary checkboxes

STEP 7 Error Handling & Logging

WordPress already logs:

- PHP errors

- Login failures

- File upload failures

- Permission errors

But we are going to enhance it:

Enable WordPress logging

Open **wp-config.php**

Find: define('WP_DEBUG', false);

Replace with:

define('WP_DEBUG', true);

define('WP_DEBUG_LOG', true);

define('WP_DEBUG_DISPLAY', false);

Now errors go to: /wp-content/debug.log

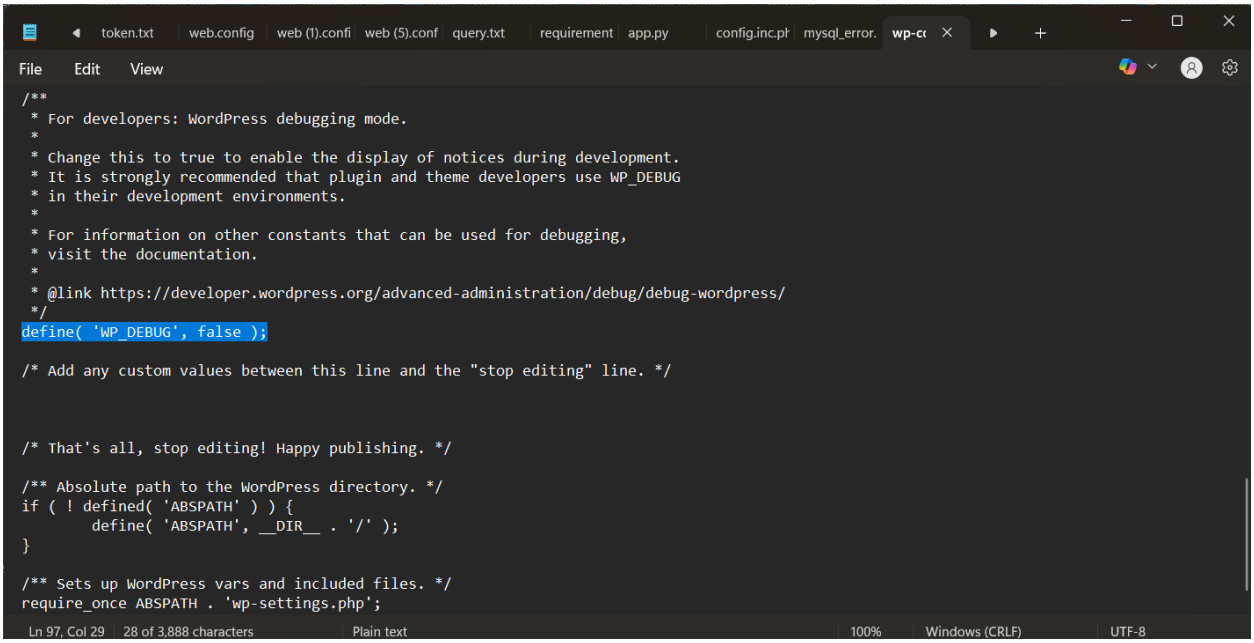


Figure 50_find

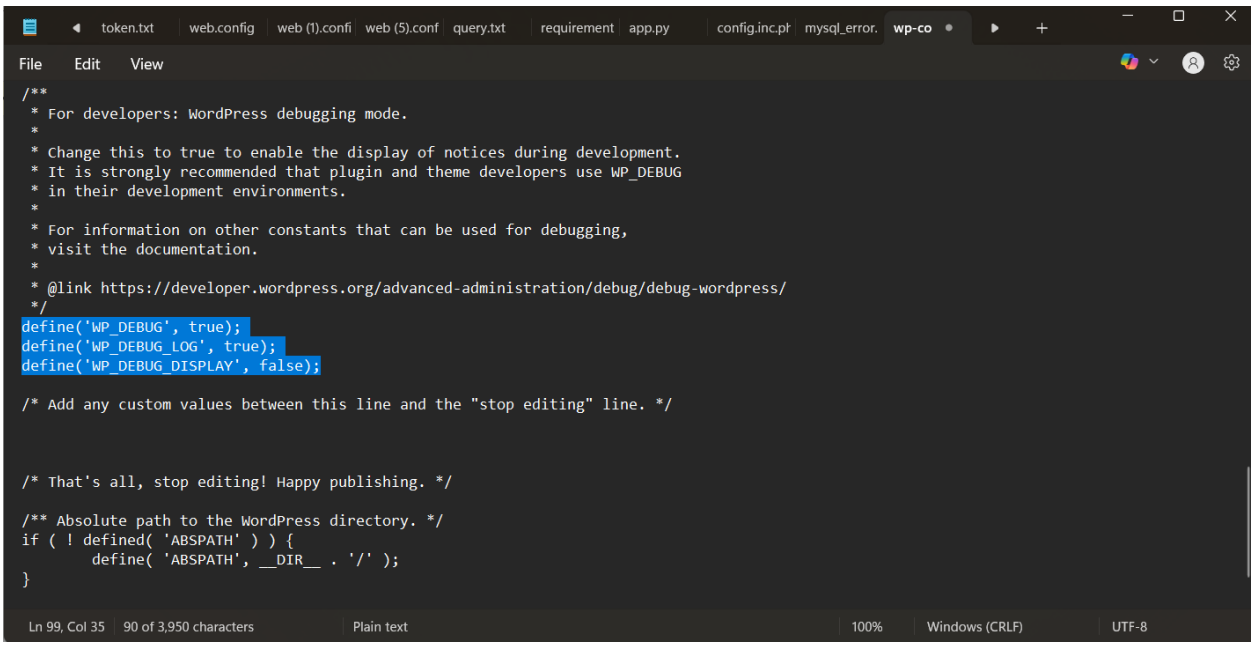


Figure 51_replace

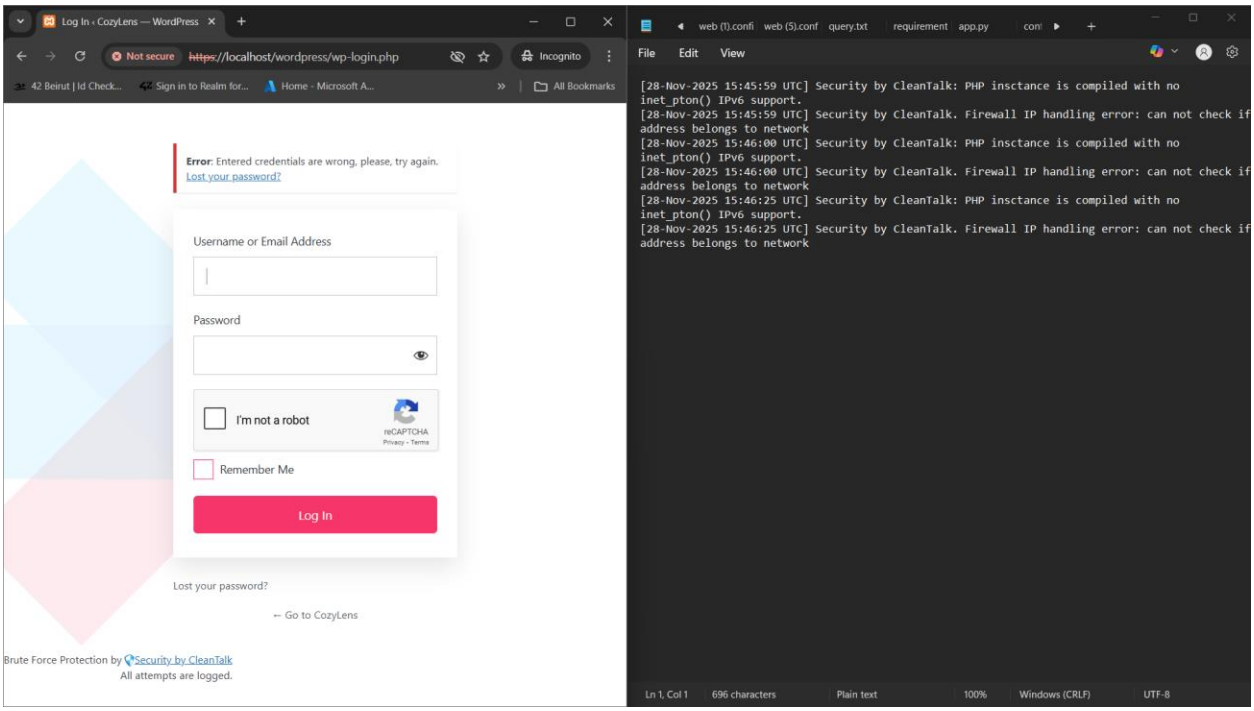


Figure 52_testing

WP Activity Log plugin

Already captures:

✓ logins

✓ failed logins

✓ role changes

✓ file uploads

✓ access to restricted pages

Using WP Activity Log, every login, logout, failed login, and role assignment is recorded

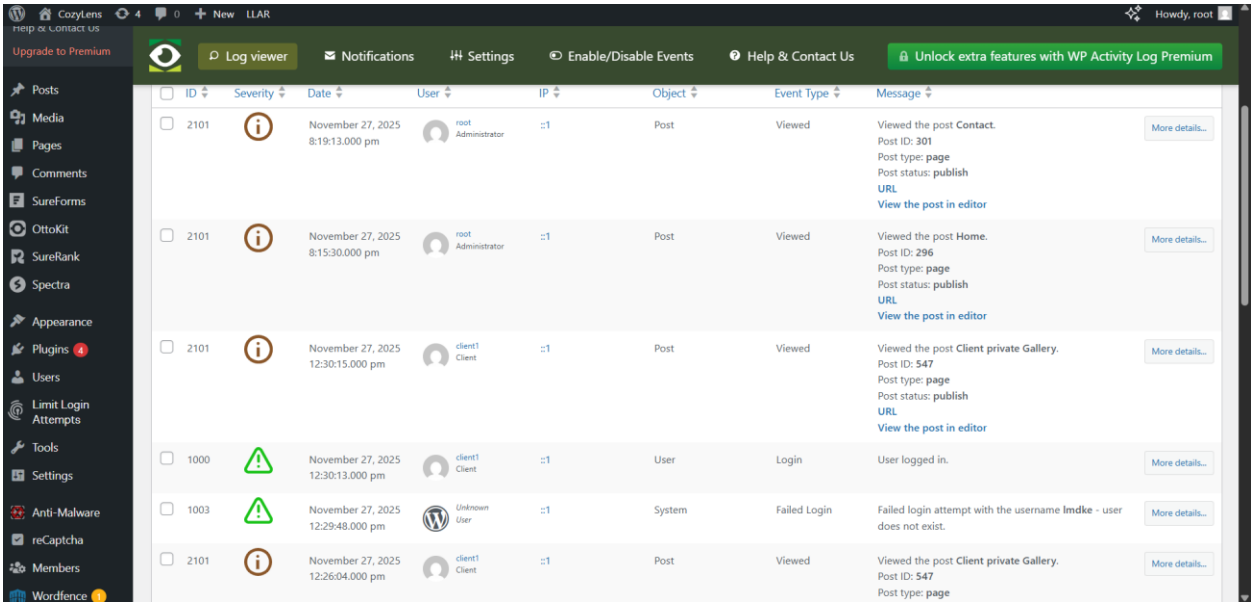


Figure 53_wp login

Step 8 Penetration Testing

Download Owasp Zap

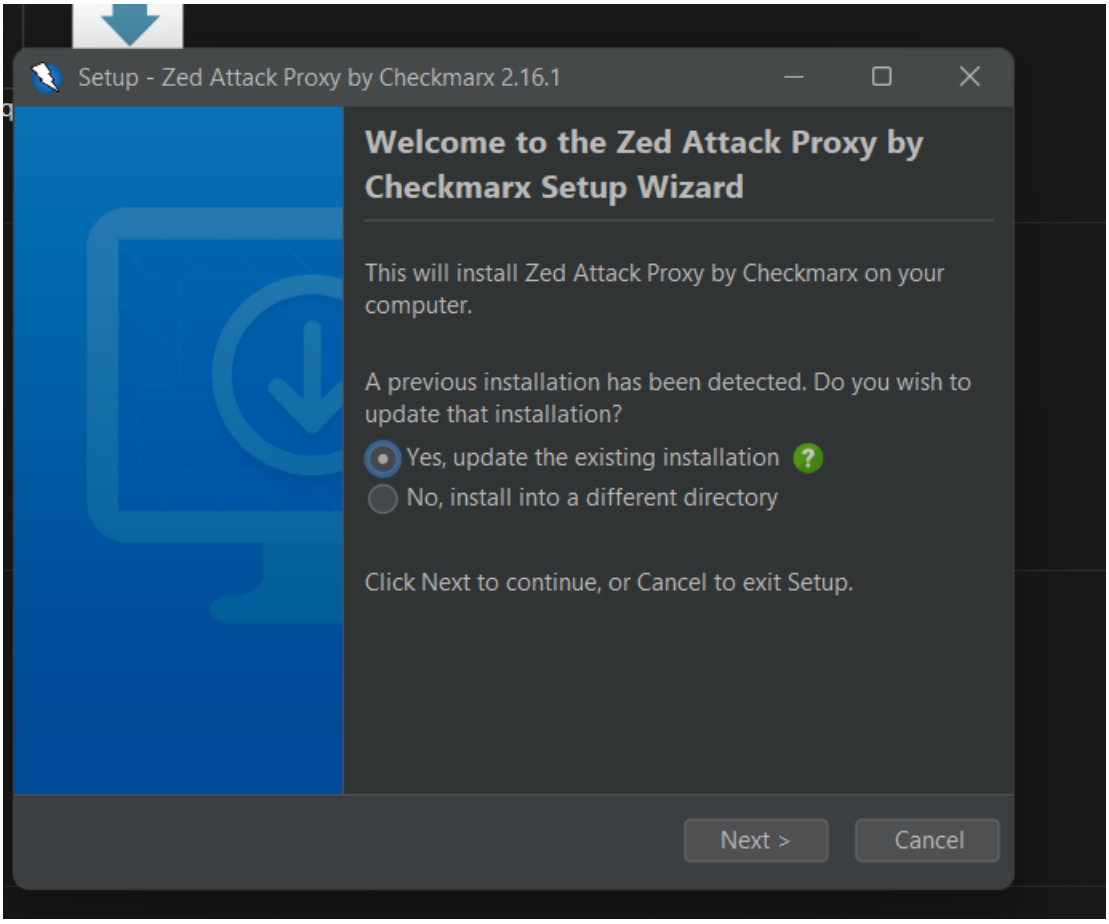


Figure 54_downloading ZAP

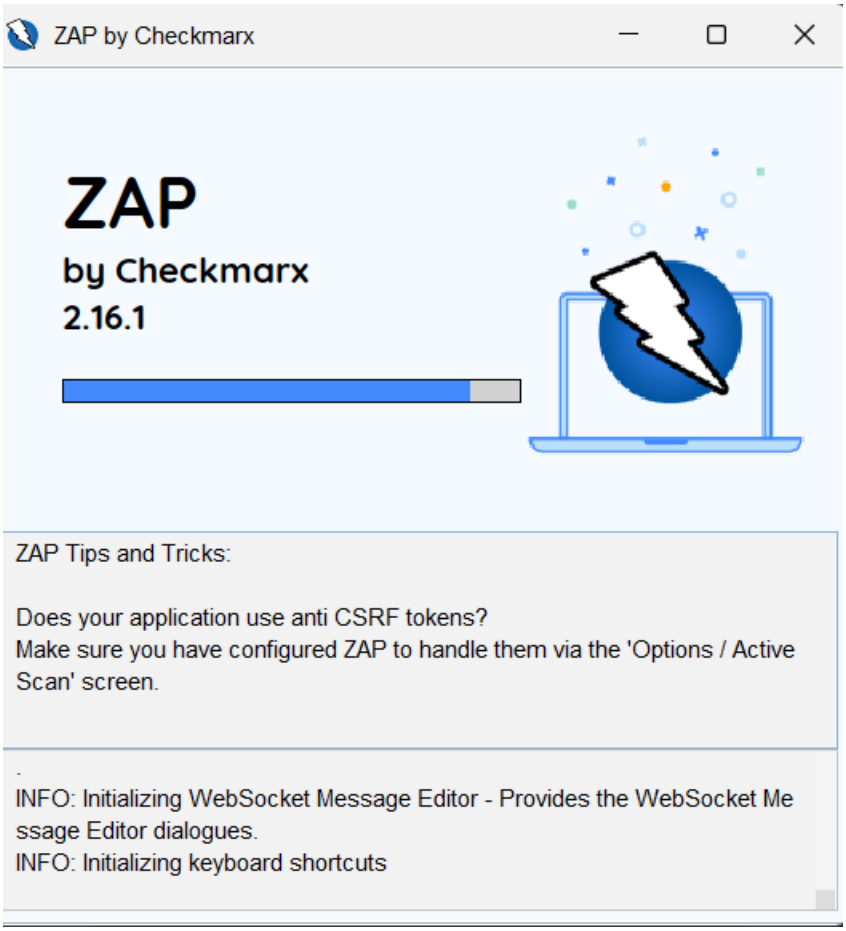


Figure 55_opening zap

Click on quickstart and automated scan

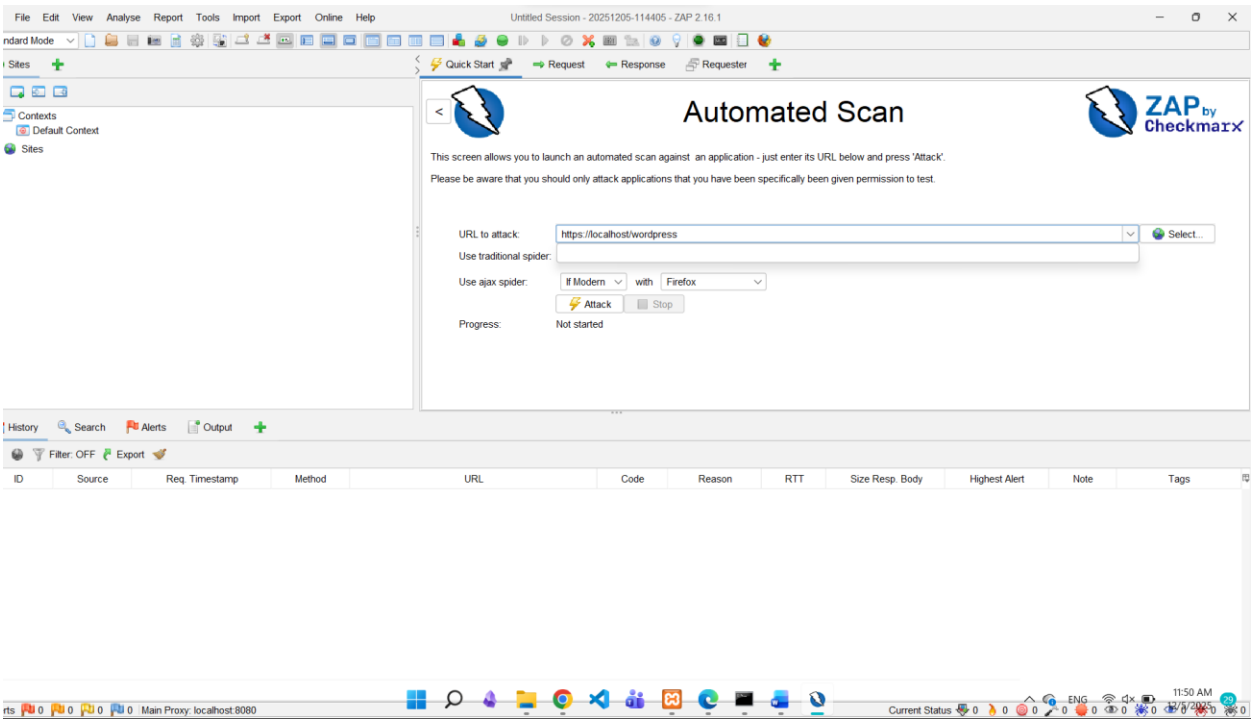


Figure 56_automated scan

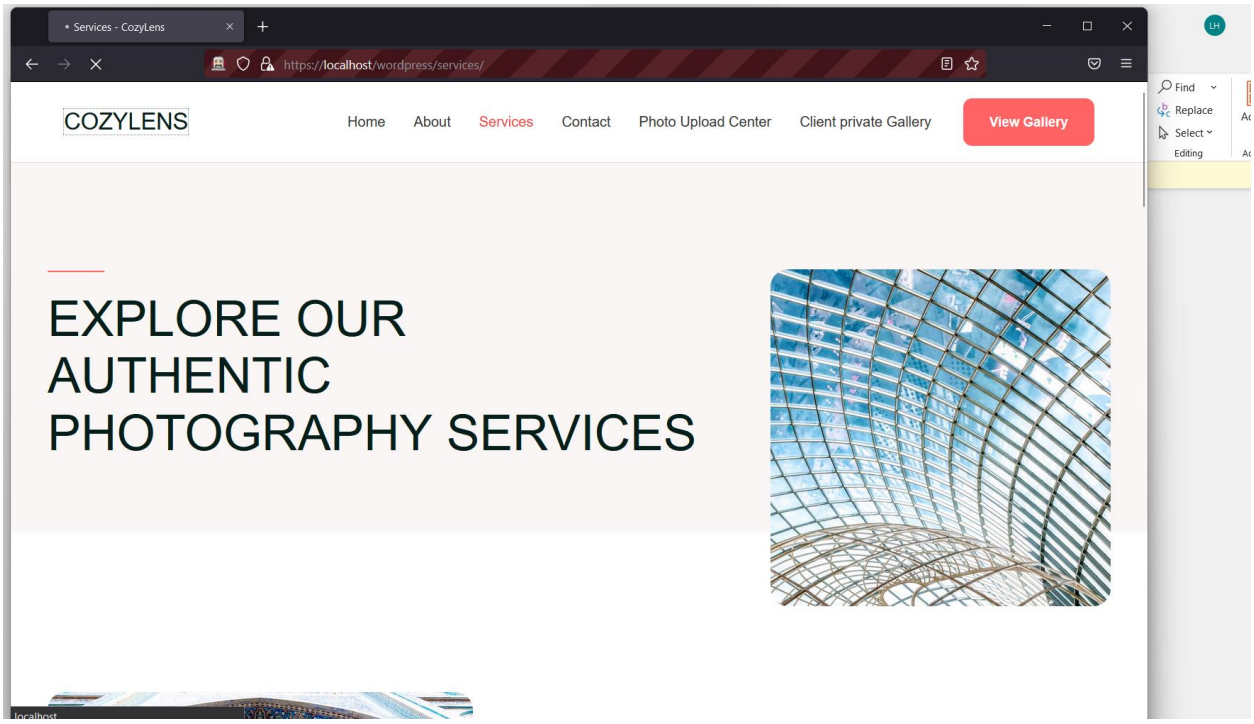


Figure 57_testing in process

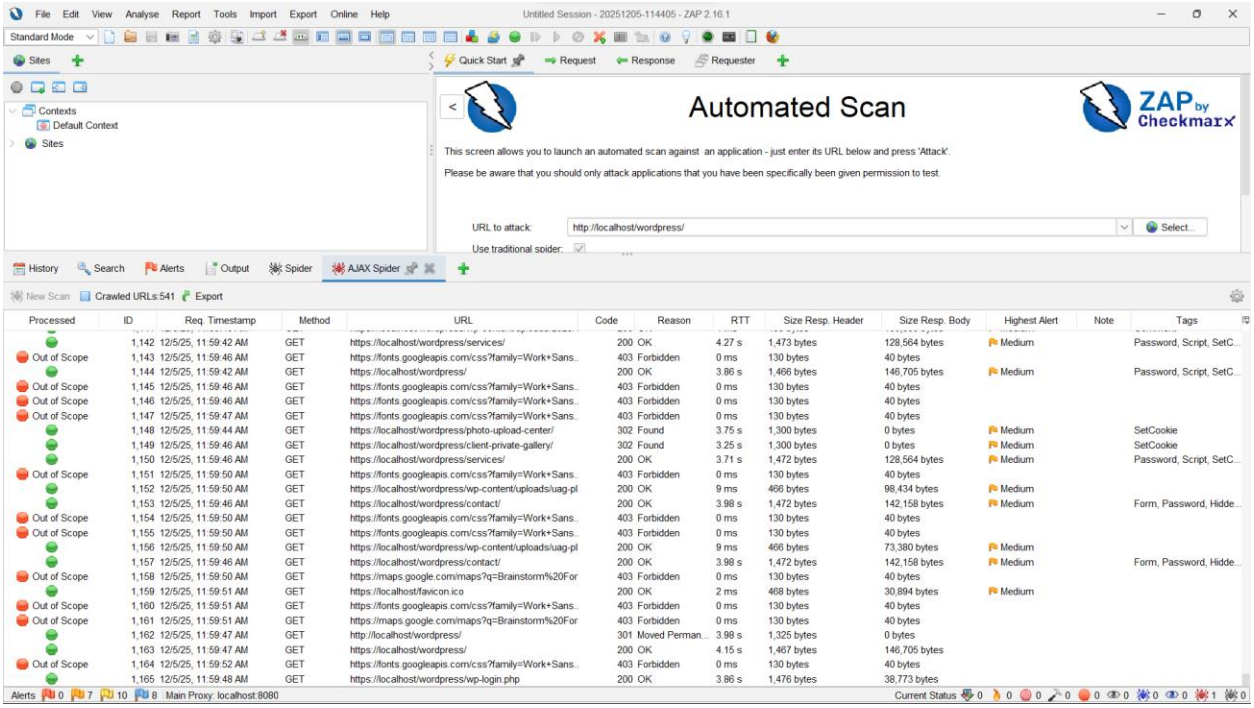


Figure 58_attack in progress

The scanner crawled the entire WordPress installation.

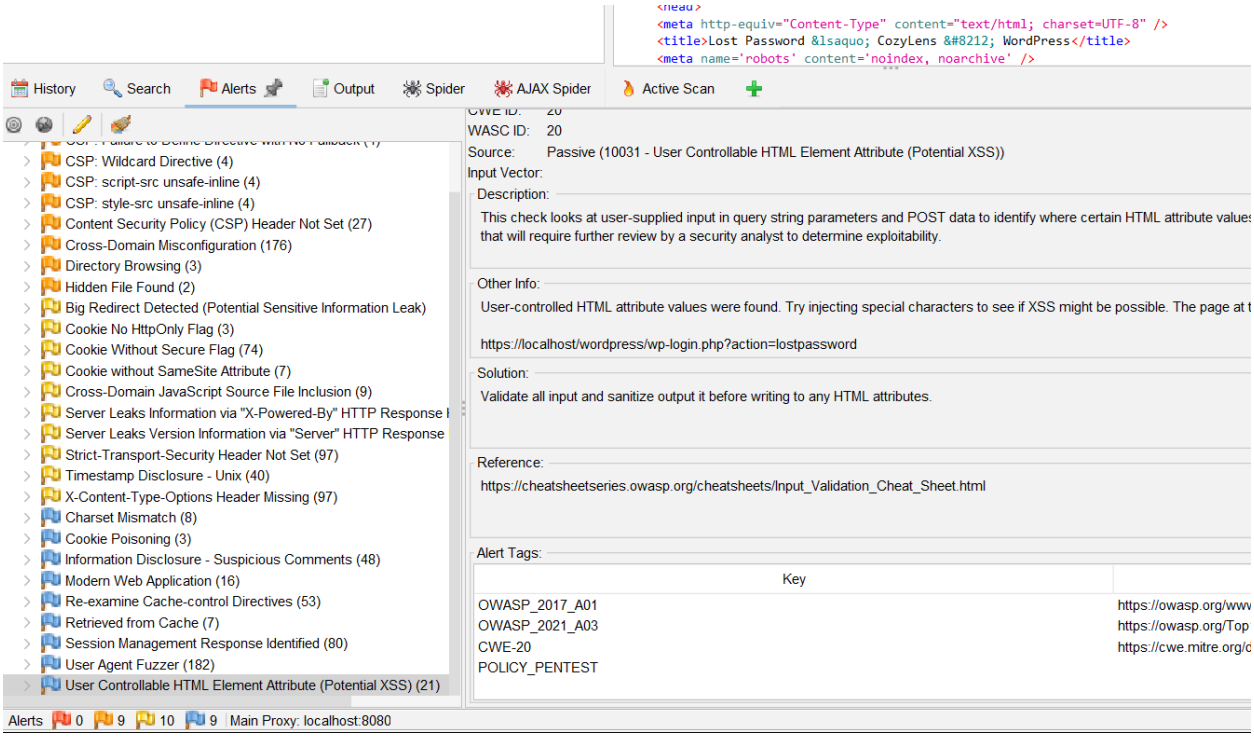


Figure 59_alerts

The most important security findings were:

1. Missing CSRF tokens

Attackers could potentially submit forms on behalf of authenticated users → CSRF attack.

A CSRF token (Cross-Site Request Forgery token) is a unique, secret, random code generated by a web application and sent to a user's browser, which must be included in subsequent requests for sensitive actions (like changing passwords or making payments) to prove the request is legitimate and not from a malicious third-party site, thus preventing attackers from tricking users into performing actions they didn't intend. It works by matching the token sent with the request against a token stored server-side for that user's session, ensuring the request originates from the act

2. Directory browsing & hidden file exposure

Files like /readme.html publicly expose version info which increases attack surface.

Solution:

1- ZAP cannot see the internal WordPress security check happening on the server. It only reads HTML

So its logic is:

"I don't see a CSRF token in the HTML → must be vulnerable."

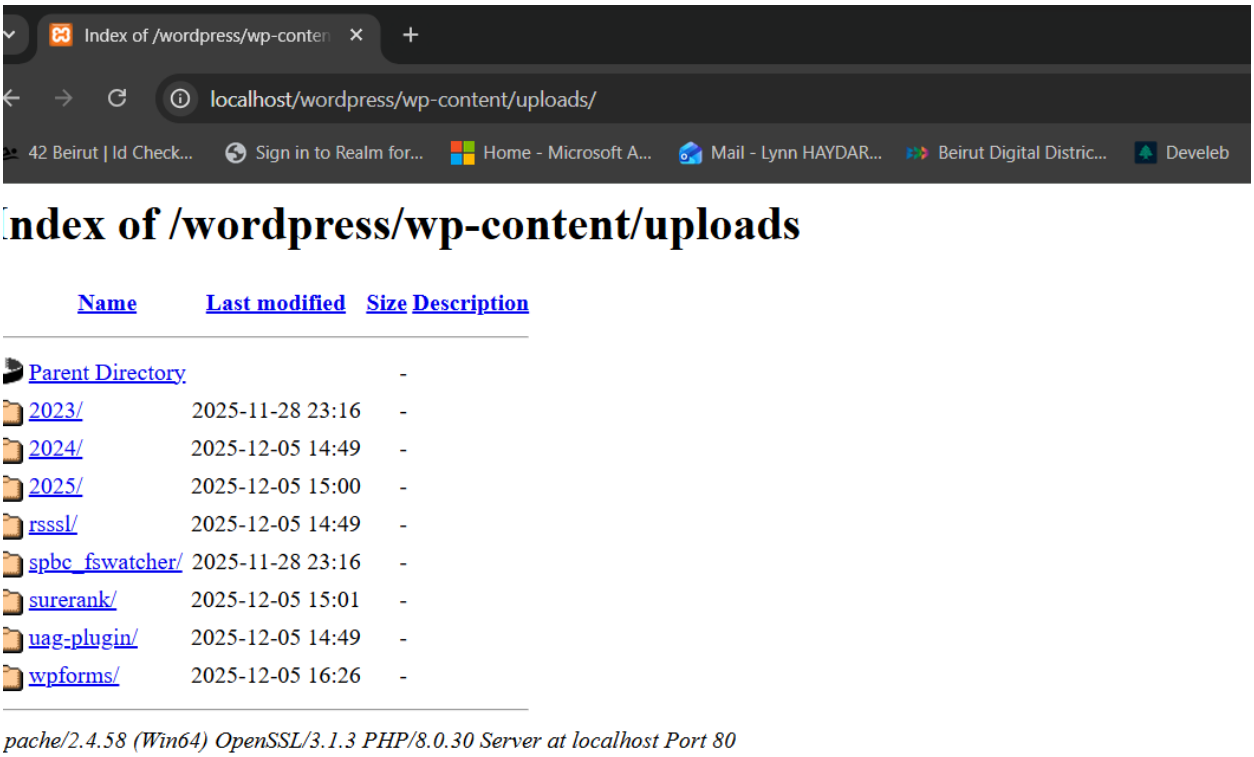
But actually:

✓ WordPress validates the form submission
✓ SureForms uses WordPress nonces
✓ CSRF attacks cannot successfully submit the form

This is why it's called a **false positive**.

2- Directory browsing means someone can visit:

http://localhost/wordpress/wp-content/uploads/



3-
Figure 60_files

and see all files.
This is a security risk.

Open:

`xampp/htdocs/wordpress/.htaccess`

Add this line **at the very top or bottom**:

```
Options -Indexes
```

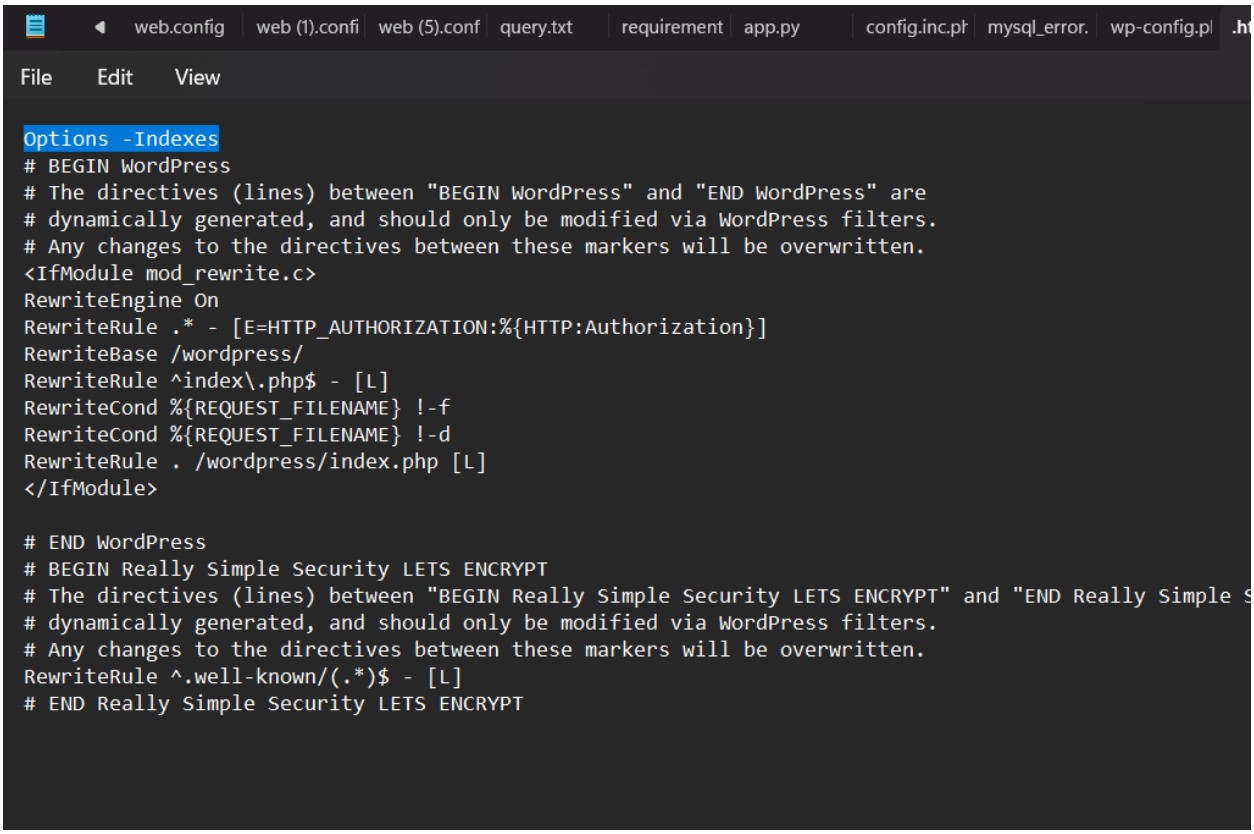


Figure 61_adding

It tells Apache to **deny listing all directories**, even if a folder has no index.php.

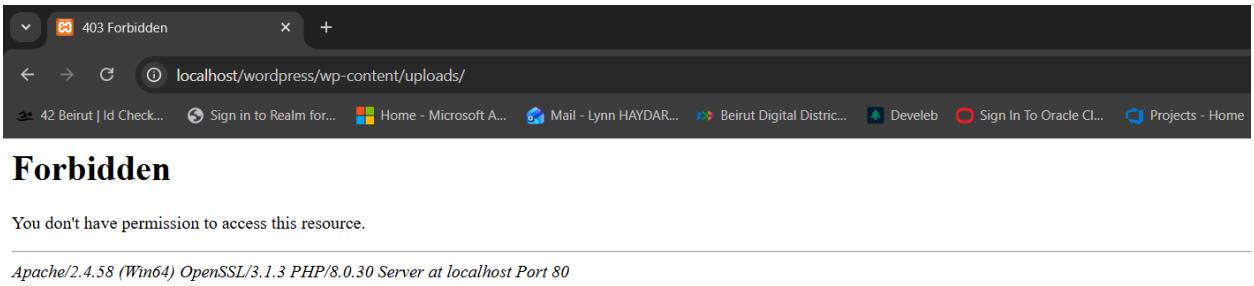


Figure 62_it worked

But it created another vulnerability

Table of Figures