# WordPress Code Workshop

# - Child Theme

## Objective

In this assignment, students will work on creating a child theme, adding a new feature, and developing A11y.

## Exercise 1: Create a Child Theme

### Step 1: Install the Hello Elementor Theme

1. Log in to your WordPress admin dashboard.
2. Navigate to **Appearance > Themes > Add New**.
3. In the search bar, type "Hello Elementor" and press Enter.
4. Locate the Hello Elementor theme, click **Install**, and wait for the installation to complete.
   * Note: You don’t need to activate it yet, as we’ll activate the child theme later, but the parent theme must be installed.

### Step 2: Download the Hello Elementor Child Theme

1. Visit the official Elementor GitHub repository or website to obtain the child theme.
   * A reliable source is the Elementor GitHub page: Elementor GitHub.
2. Download the child theme as a zip file (e.g., hello-elementor-child.zip).

### Step 3: Install the Child Theme

1. Return to your WordPress admin dashboard.
2. Go to **Appearance > Themes > Add New > Upload Theme**.
3. Click **Choose File**, select the hello-elementor-child.zip file you downloaded, and click **Install Now**.
4. Wait for the installation to finish.

### Discussion

* Why do we need a child theme?

## Exercise 2: Editing the Child Theme’s CSS for Accessibility

Below are practical CSS examples you can add to the style.css file to make your site more accessible. These focus on common accessibility principles like visibility, usability, and scalability.

### Improve Focus Styles

Ensure that keyboard users can easily see which element is focused (e.g., links, buttons, or form fields) by adding a clear focus outline.

a:focus, button:focus, input:focus, textarea:focus {

outline: 2px solid #007CBA; /\* A high-contrast blue \*/

outline-offset: 2px; /\* Adds space around the outline \*/

}

## Exercise 3: Upload SVG - functions.php

### Task

Students will create a new feature in the code map to upload SVG file type.

### Steps

1. Log into your hosting control panel (e.g., cPanel, Plesk).
2. Open File Manager.
3. Go to: public\_html/wp-content/themes/your-theme-name/.
4. Locate functions.php and edit it.

function allow\_svg\_upload\_for\_admins($mimes) {

if (current\_user\_can('administrator')) {

$mimes['svg'] = 'image/svg+xml';

}

return $mimes;

}

add\_filter('upload\_mimes', 'allow\_svg\_upload\_for\_admins');

### Discussion

* Why do we use the allow\_svg\_upload\_for\_admins function? What will happen to uploads by non-admins?
* What is the purpose of the functions.php file in WordPress?
* What types of functionality can you add or modify using functions.php?

## WP Plugin Development

### Exercise 4: Make Your Own Plugins

#### Task

Create a new plugin folder and file.

### Step 1: Create a New Plugin Folder

1. Go to your WordPress installation directory.
2. Navigate to the wp-content/plugins folder.
3. Create a new folder for your plugin. For example, name it footer-text-plugin.

### Step 2: Create the Plugin File

1. Inside your plugin folder (footer-text-plugin), create a PHP file, e.g., footer-text-plugin.php.
2. Open the PHP file in a text editor (e.g., VSCode, Sublime Text).

### Step 3: Add the PHP Code

<?php

/\*

Plugin Name: Footer Text Plugin

Author: xbi

Author URI: www.example.com

Description: Adds text at bottom of posts.

Version: 1.0

\*/

// Function to add custom text to post content

function add\_footer\_text($content) {

return $content . '<p>Custom footer text by xbi.</p>';

}

// Hook function to 'the\_content'

add\_filter('the\_content', 'add\_footer\_text');

?>

### Step 4: Activate the Plugin and See Results

### Discussion

* Why is it important to develop your own WordPress plugins from a sustainability and security perspective?
* Consider the long-term impact of using third-party plugins versus custom-built solutions.
* How can developing your own plugins contribute to the overall performance, security, and maintainability of a WordPress site over time?