# Product Gallery Slider - Documentation

**Version:** 1.5.0

Thank you for choosing Product Gallery Slider! This guide will walk you through setting up and customizing the plugin for your WordPress site.

# 1. User Guide

This section covers the basic installation and usage of the plugin for non-developers.

## 1.1. Installation

1. Download the `product-gallery-slider.zip` file.
2. In your WordPress dashboard, navigate to **Plugins > Add New**.
3. Click the **Upload Plugin** button at the top of the page.
4. Choose the `product-gallery-slider.zip` file and click **Install Now**.
5. Once installed, click **Activate Plugin**.

## 1.2. Standard Gallery (Shortcode & Block)

You have two ways to add a standard gallery to any page or post.

#### Using the Gutenberg Block (Recommended)

1. In the page editor, click the "+" button to add a new block.
2. Search for "Product Gallery Slider" and add it.
3. Click the "Select Images" button to open the Media Library.
4. Choose your images and click "Create a new gallery", then "Insert gallery".

#### Using the Shortcode

To create a gallery with three images (with Media Library IDs: 123, 45, 678), you would use:

```
[product_gallery ids="123, 45, 678"]
```

## 1.3. WooCommerce Integration

The plugin automatically integrates with WooCommerce. You have two options for displaying the gallery on your single product pages.

#### Option A: Automatic Replacement (Default)

By default, the plugin will automatically replace the standard WooCommerce product gallery. This works for most standard themes. No action is required.

#### Option B: Manual Placement with Shortcode (For Page Builders)

If you use a page builder (like Elementor, Divi, etc.), use the `[wc_product_gallery]` shortcode for full control.

1. In your page builder layout, remove the default "Product Images" widget.
2. Add a "Shortcode" or "HTML" widget in its place.
3. Insert the shortcode: `[wc_product_gallery]`

**Note:** If your layout has separate sections for mobile and desktop, you can safely use this shortcode in both. The plugin will handle it correctly.

# 2. Developer Guide

This section provides information on how to extend and customize the plugin using WordPress filters and JavaScript events. Add the following code snippets to your theme's `functions.php` file or a custom plugin.

## 2.1. PHP Filters

### `pgs_auto_replace_wc_gallery`

Disable the automatic replacement of the WooCommerce gallery. This is useful if you *only* want to use the `[wc_product_gallery]` shortcode for placement.

- **Type:** `boolean`
- **Default:** `true`

```
add_filter( 'pgs_auto_replace_wc_gallery', '__return_false' );
```

### `pgs_variation_gallery_images`

Modify the gallery images for a specific product variation.

- **Parameters:**
  - `$images` (array): The array of image data for the variation.
  - `$variation_obj` (WC_Product_Variation): The variation product object.
  - `$product` (WC_Product): The main parent product object.

**Example: Add a specific lifestyle image to a "Blue T-Shirt" variation.**

```
add_filter( 'pgs_variation_gallery_images', 'add_lifestyle_image_to_blue_variation', 10, 3 );

function add_lifestyle_image_to_blue_variation( $images, $variation_obj, $product ) {

    // Target a specific product by its slug
    if ( $product->get_slug() !== 't-shirt' ) {
        return $images;
    }

    // Target the 'Blue' variation by its attribute
    if ( $variation_obj->get_attribute( 'color' ) === 'Blue' ) {

        $lifestyle_image_id = 456; // The ID of your lifestyle image
```

```
        $image_url = wp_get_attachment_image_url( $lifestyle_image_id, 'large' );

        if ( $image_url ) {
            // Add the new image to the end of the gallery
            $images[] = [
                'url' => $image_url,
                'alt' => 'Model wearing the blue t-shirt',
            ];
        }
    }

    return $images;
}
```

## 2.2. JavaScript Custom Event

The plugin fires a custom event on the `document` object whenever the active image in a gallery changes.

- **Event Name:** `pgs_after_image_change`
- **Event Detail:** The event's `detail` object contains:
  - `gallery` (object): The `ProductGallery` class instance.
  - `newIndex` (integer): The index of the newly active image.

**Example: Log the new image index to the console whenever the gallery changes.**

```
document.addEventListener('pgs_after_image_change', function(e) {
    const galleryInstance = e.detail.gallery;
    const newImageIndex = e.detail.newIndex;

    console.log('Gallery image changed!');
    console.log('Gallery ID:', galleryInstance.galleryWrapper.id);
    console.log('New active image index:', newImageIndex);
});
```