function my\_theme\_styles() {

    wp\_enqueue\_style('my-theme-style', get\_stylesheet\_uri());

}

add\_action('wp\_enqueue\_scripts', 'my\_theme\_styles');

function wp2023\_enqueue\_scripts() {

    if (is\_page('khung\_de\_ban') || is\_singular('product')) {

        wp\_enqueue\_script('jquery');

        // Enqueue Cropper CSS

        wp\_enqueue\_style('cropper-css', 'https://cdnjs.cloudflare.com/ajax/libs/cropperjs/1.5.11/cropper.min.css', array(), '1.5.11');

        // Enqueue Cropper JS

        wp\_enqueue\_script('cropper-js', 'https://cdnjs.cloudflare.com/ajax/libs/cropperjs/1.5.11/cropper.min.js', array('jquery'), '1.5.11', true);

        wp\_enqueue\_script('custom-script-handle', 'path-to-your-script.js', array('jquery'), null, true);

        // Assume you have the $post\_id available in this context:

        $post\_id = get\_the\_ID();

        $max\_images = get\_post\_meta($post\_id, 'max\_images', true);

        wp\_localize\_script('custom-script-handle', 'image', array(

            'maxUploads' => $max\_images ? $max\_images : 10 // Default to 10 if not set

        ));

        // Get parameter display image in frame

        wp\_enqueue\_script('my\_custom\_script', 'path\_to\_your\_script.js');

        // Get product category slugs

        $product\_cats = wp\_get\_post\_terms(get\_the\_ID(), 'product\_cat', array('fields' => 'slugs'));

        // Determine parameters based on category slugs

        $parameters = in\_array('khung-de-ban', $product\_cats) ? 1 : (in\_array('khung-khac-laser', $product\_cats) ? 2 : 0);

        // Localize script with parameters

        wp\_localize\_script('my\_custom\_script', 'myScriptParams', array('parameters' => $parameters));

        // Css main

        wp\_enqueue\_style('wp2023-css', get\_template\_directory\_uri() . './../../plugins/wp-ecommerce/public/css/main.css', array(), '1.0.0');

    }

}

add\_action('wp\_enqueue\_scripts', 'wp2023\_enqueue\_scripts');

function add\_custom\_upload\_button() {

    global $product;

    $productId = $product->get\_id();

    // Check if the product is in the 'KHUNG ĐỂ BÀN' category

    if ( has\_term( 'khung-de-ban', 'product\_cat', $product->get\_id() ) ) {

        // Use do\_shortcode to display the content of your custom shortcode

        echo do\_shortcode('[tabletop\_frame\_template]');

    }

    else if (has\_term( 'khung-treo-tuong', 'product\_cat', $product->get\_id() )) {

        echo do\_shortcode('[tabletop\_frame\_template]');

    }

    else if (has\_term( 'khung-khac-laser', 'product\_cat', $product->get\_id() )) {

        echo do\_shortcode('[tabletop\_frame\_template]');

    }

    else if (has\_term( 'khung-bang-khen', 'product\_cat', $product->get\_id() )) {

        echo do\_shortcode('[tabletop\_frame\_template]');

    } else {

        echo do\_shortcode('[tabletop\_frame\_template]');

    }

}

add\_action('woocommerce\_single\_product\_summary', 'add\_custom\_upload\_button', 25);

// Hook to add the custom field to the product general options

add\_action('woocommerce\_product\_options\_general\_product\_data', 'wp2023\_add\_link\_shopee\_field');

// Function to display the custom field

function wp2023\_add\_link\_shopee\_field() {

    woocommerce\_wp\_text\_input(

        array(

            'id' => 'link\_shopee',

            'label' => \_\_('Shopee Link', 'wp-photo'),

            'placeholder' => 'https://shopee...',

            'desc\_tip' => 'true',

            'description' => \_\_('Enter the Shopee link for this product.', 'wp-photo'),

        )

    );

}

add\_action('woocommerce\_process\_product\_meta', 'wp2023\_save\_link\_shopee\_field');

function wp2023\_save\_link\_shopee\_field($post\_id) {

    // Check if `link\_shopee` is set and not empty

    if (isset($\_POST['link\_shopee']) && !empty($\_POST['link\_shopee'])) {

        // Sanitize and update the post meta

        update\_post\_meta($post\_id, 'link\_shopee', sanitize\_text\_field($\_POST['link\_shopee']));

    } else {

        // Delete the post meta if the custom field is not set

        delete\_post\_meta($post\_id, 'link\_shopee');

    }

}

function wp2023\_tabletop\_frame\_template() {

    global $wpdb;

    ob\_start();

global $product;

// // Get the Shopee link from the product's custom field

// $shopee\_link = get\_post\_meta($product->get\_id(), 'link\_shopee', true);

if ( is\_a( $product, 'WC\_Product' ) ) {

    $productId = $product->get\_id();

    $isKhungKhacLaser = has\_term('khung-khac-laser', 'product\_cat', $productId);

        // Print a small script to set the JavaScript variable

        echo "<script type='text/javascript'>\n";

        echo "var isKhungKhacLaser = " . json\_encode($isKhungKhacLaser) . ";\n";

        echo "</script>\n";

    if (has\_term('khung-de-ban', 'product\_cat', $productId)) {

        // Pass the category ID to the included file

        $term = get\_term\_by('slug', 'khung-de-ban', 'product\_cat');

        $category\_id = $term ? $term->term\_id : null;

        set\_query\_var('category\_id', $category\_id);

        include(get\_template\_directory() . './../../plugins/wp-ecommerce/includes/templates/tabletop-frame.php');

    }

    elseif (has\_term('khung-khac-laser', 'product\_cat', $productId)) {

        // Pass the category ID to the included file

        $term = get\_term\_by('slug', 'khung-khac-laser', 'product\_cat');

        $category\_id = $term ? $term->term\_id : null;

        set\_query\_var('category\_id', $category\_id);

        include(get\_template\_directory() . './../../plugins/wp-ecommerce/includes/templates/laser-frame.php');

    }

    elseif (has\_term('khung-bang-khen', 'product\_cat', $productId)) {

        // Pass the category ID to the included file

        $term = get\_term\_by('slug', 'khung-bang-khen', 'product\_cat');

        $category\_id = $term ? $term->term\_id : null;

        set\_query\_var('category\_id', $category\_id);

        include(get\_template\_directory() . './../../plugins/wp-ecommerce/includes/templates/merit-frame.php');

        echo '<div id="forminator-form-container" style="display: none;">';

        echo do\_shortcode('[forminator\_form id="372"]');

        echo '</div>';

    } else {

        include(get\_template\_directory() . './../../plugins/wp-ecommerce/includes/templates/default-frame.php');

    }

} else {

    $productId = 0; // Fallback if not on a product page

}

    ?>

<style>

.modal {

    display: none;

    /\* Hidden by default \*/

    position: fixed;

    /\* Stay in place \*/

    z-index: 100;

    /\* Sit on top \*/

    left: 50%;

    top: 50%;

    transform: translate(-50%, -50%);

    /\* Centers the modal \*/

    width: 100%;

    /\* Full width \*/

    height: 100%;

    /\* Full height \*/

    overflow-y: auto;

    /\* Enable scroll if needed \*/

    background-color: rgb(0, 0, 0);

    /\* Fallback color \*/

    background-color: rgba(0, 0, 0, 0.4);

    /\* Black w/ opacity \*/

}

/\* Modal Content Box \*/

.modal-content {

    position: relative;

    background-color: #fefefe;

    margin: 2% auto 0;

    /\* 15% from the top and centered \*/

    /\* padding: 30px; \*/

    border: 1px solid #888;

    width: 80%;

    overflow-y: hidden;

    /\* Could be more or less, depending on screen size \*/

}

/\* The Close Button \*/

.close {

    position: absolute;

    top: -12px;

    right: 2px;

    color: #aaa;

    float: right;

    font-size: 28px;

    font-weight: bold;

}

.close:hover,

.close:focus {

    color: black;

    text-decoration: none;

    cursor: pointer;

}

.wrap-size {

    display: flex;

    flex-wrap: wrap;

    justify-content: space-between;

    align-items: center;

}

.size {

    padding: 5px 10px;

    border: 1px solid #f6941c;

    border-radius: 4px;

    cursor: pointer;

    color: #222;

}

.size:hover {

    background-color: #f6941c;

    /\* or any highlight color \*/

    color: white;

}

.size.selected {

    background-color: #f6941c;

    /\* or any highlight color \*/

    color: white;

}

.orientation {

    display: flex;

    flex-wrap: wrap;

    justify-content: flex-start;

    align-items: center;

}

.orientation .orientation-button:first-child {

    margin: 0;

}

.orientation .orientation-button {

    padding: 5px 10px;

    border: 1px solid #f6941c;

    border-radius: 4px;

    cursor: pointer;

    color: #222;

    margin-left: 20px;

}

.orientation .orientation-button:hover {

    background-color: #f6941c;

    /\* or any highlight color \*/

    color: white;

}

.orientation .orientation-button.selected {

    background-color: #f6941c;

    /\* or any highlight color \*/

    color: white;

}

.thumbnails-templates .thumbnail-image {

    cursor: pointer;

}

#wp2023-image-thumbnails-container {

    margin: 5px 0;

}

#review-button {

    width: 50;

    padding: 5px 10px;

    border: 1px solid #f6941c;

    background-color: #fff;

}

#review-button:hover {

    background-color: #f6941c;

}

</style>

<!-- JavaScript for handling image upload and cropping -->

<script>

jQuery(document).ready(function($) {

    var lastEditType = '';

    var lastParameter = '';

    let currentCropper = null;

    let editedImages = [];

    let imageIdToEdit;

    var productId = <?php echo json\_encode( $productId ); ?>;

    var currentImageIndex = null;

    var validFiles = [];

    var selectedSize = '';

    var TemplateId = '';

    var selectedTemplateIndex = '';

    var imageFiles = [];

    let selectedImageSrc;

    var croppedImages = [];

    function updateCropperAspectRatio(aspectRatio) {

        if (!aspectRatio) {

            // Calculate aspect ratio for non-horizontal images

            selectedSize = $('.wrap-size div.selected').data('value').split('x').map(Number);

            aspectRatio = selectedSize[0] / selectedSize[1]; // Height / Width for vertical

        }

        console.log('Aspect Ratio:', aspectRatio);

        if (currentCropper) {

            currentCropper.setAspectRatio(aspectRatio);

            currentCropper.replace(currentCropper.canvas.toDataURL());

        }

    }

    function calculateAspectRatioHorizontal() {

        selectedSize = $('.wrap-size div.selected').data('value').split('x').map(Number);

        var aspectRatio = selectedSize[1] / selectedSize[0]; // Width / Height for horizontal

        console.log('Calculated Horizontal Aspect Ratio:', aspectRatio);

        return aspectRatio; // Return the calculated aspect ratio

    }

    $('.wrap-size div:first-child').addClass('selected');

    updateCropperAspectRatio();

    // If there's a Cropper instance, set the default aspect ratio

    // Size selection event

    $('.wrap-size div').on('click', function() {

        $('.wrap-size div').removeClass('selected');

        $(this).addClass('selected');

        updateCropperAspectRatio();

    });

    $('.thumbnails-templates .thumbnail-image').each(function() {

        if ($(this).data('is-horizontal') === false) {

            $(this).addClass('selected').css('border', '2px solid #f6941c');

            // If you need to trigger any other function when this class is added, do it here.

            selectedImageSrc = $(this).attr('src');

            TemplateId = $(this).data('template-id');

            selectedTemplateIndex = $(this).data('index');

            return false; // This will break the loop after adding the class to the first element.

        }

    });

    $('.thumbnails-templates').on('click', '.thumbnail-image', function() {

        // console.log('Data attribute (is-horizontal):', $(this).data('is-horizontal')); // Debug

        $('.thumbnail-image').removeClass('selected').css('border', '');

        $(this).addClass('selected').css('border', '2px solid #f6941c');

        selectedImageSrc = $(this).attr('src');

        TemplateId = $(this).data('template-id');

        selectedTemplateIndex = $(this).data('index');

        var aspectRatio;

        // Update the aspect ratio if the image is a horizontal template

        if ($(this).data('is-horizontal')) {

            aspectRatio = calculateAspectRatioHorizontal();

            // console.log('Thumbnail Click - Horizontal Aspect Ratio:', aspectRatio);

        } else {

            selectedSize = $('.wrap-size div.selected').data('value').split('x').map(Number);

            aspectRatio = selectedSize[0] / selectedSize[1];

            // console.log('Thumbnail Click - Vertical Aspect Ratio:', aspectRatio);

        }

        updateCropperAspectRatio(aspectRatio);

    });

    $('.orientation .orientation-button:first-child').addClass('selected');

    $('.orientation-button').on('click', function() {

        var orientation = $(this).data('orientation'); // Get the data-orientation value

        // Remove 'selected' class from all buttons and then add to the clicked one

        $('.orientation-button').removeClass('selected');

        $(this).addClass('selected');

        // Calculate and update the aspect ratio based on the orientation

        selectedSize = $('.wrap-size div.selected').data('value').split('x').map(Number);

        if (currentCropper && $('.thumbnail-image.selected').length > 0) {

            selectedSize = $('.wrap-size div.selected').data('value').split('x').map(Number);

            var aspectRatio = orientation === 'horizontal' ? selectedSize[1] / selectedSize[0] :

                selectedSize[0] / selectedSize[1];

            currentCropper.setAspectRatio(aspectRatio);

        }

    });

    // Whenever a size or template is selected, call the updateCropperAspectRatio function

    $('.wrap-size div, .thumbnails-templates img').on('click', function() {

        // Set the selected state for sizes and templates

        if ($(this).hasClass('size')) {

            $('.wrap-size div').removeClass('selected');

        } else {

            $('.thumbnail-image').removeClass('selected');

        }

        $(this).addClass('selected');

        // Update the aspect ratio of the cropper

        updateCropperAspectRatio();

    });

    $('#wp2023-custom-image').on('change', function(event) {

        // Reset cropped images array

        croppedImages = [];

        imageFiles = event.target.files;

        var maxUploads = image.maxUploads || 10;

        clearThumbnails();

        if (imageFiles.length > maxUploads) {

            displayError(`You can only upload up to ${maxUploads} images for the selected frame type.`);

            return;

        }

        if (imageFiles.length > maxUploads) {

            displayError(`You can only upload up to ${maxUploads} images.`);

            return;

        }

        validateFiles(imageFiles); // Call the validation function

        // Check if there are valid files after validation

        if (validFiles.length > 0) {

            // Automatically start editing the first valid image

            currentImageIndex = 0;

            startEditingImage(currentImageIndex);

        }

    });

    function clearThumbnails() {

        var container = document.getElementById('wp2023-image-thumbnails-container');

        while (container.firstChild) {

            container.removeChild(container.firstChild);

        }

    }

    // Function to start editing the image

    function startEditingImage(index) {

        if (index >= imageFiles.length) {

            // All images have been processed

            return; // Exit the function

        }

        var file = imageFiles[index];

        if (!file) return;

        var reader = **new** FileReader();

        reader.onload = function(e) {

            $('#image-to-edit').attr('src', e.target.result);

            $('#image-edit-modal').show();

            // Determine aspect ratio based on the selected template and size

            var isHorizontalTemplate = $('.thumbnail-image.selected').data('is-horizontal');

            var isHorizontalButton = $('.orientation-button.selected').data('orientation') === 'horizontal';

            var isHorizontal = isHorizontalTemplate || isHorizontalButton;

            selectedSize = $('.wrap-size div.selected').data('value');

            var aspectRatio = NaN;

            if (selectedSize) {

                var sizes = selectedSize.split('x').map(Number);

                aspectRatio = isHorizontal ? sizes[1] / sizes[0] : sizes[0] / sizes[1];

            }

            if (currentCropper) {

                currentCropper.destroy();

            }

            currentCropper = **new** Cropper($('#image-to-edit')[0], {

                aspectRatio: aspectRatio,

                zoomable: true,

                scrolable: true,

                rotatable: true,

                viewMode: 2,

                dragMode: 'none', // Prevent moving the crop box

                cropBoxResizable: false, // Prevent resizing the crop box

                cropBoxMovable: true, // Prevent moving the crop box

            });

        };

        reader.readAsDataURL(file);

    }

    function validateFiles(files) {

        // Reset validFiles only when new files are chosen

        validFiles = [];

        // $('#wp2023-image-thumbnails-container').empty();

        $('#error-message-container').hide();

        const maxFileSize = 5 \* 1024 \* 1024; // 5MB

        Array.from(files).forEach(file => {

            if (file.size > maxFileSize) {

                console.log('File too large:', file.name);

                displayError('File too large. Maximum size is 5MB.');

                return;

            }

            // If the file passed both checks, add it to the valid files array

            validFiles.push(file);

            // readAndPreviewImage(file, validFiles.length - 1);

        });

        if (validFiles.length === 0) {

            displayError('No valid files to upload. Please check your files and try again.');

        }

    }

    $('#modal-zoom-in').on('click', function() {

        if (currentCropper) {

            currentCropper.zoom(0.1);

            updateEdit(currentImageIndex, 'zoom', 'in');

        }

    });

    $('#modal-zoom-out').on('click', function() {

        if (currentCropper) {

            currentCropper.zoom(-0.1);

            updateEdit(currentImageIndex, 'zoom', 'out');

        }

    });

    $('#modal-rotate-left').on('click', function() {

        if (currentCropper) {

            currentCropper.rotate(-45);

            updateEdit(currentImageIndex, 'rotate', '-45');

        }

    });

    $('#modal-rotate-right').on('click', function() {

        if (currentCropper) {

            currentCropper.rotate(45);

            updateEdit(currentImageIndex, 'rotate', '45');

        }

    });

    $(window).resize(function() {

        if (currentCropper) {

            // Adjust cropper size here if necessary

            currentCropper.resize();

        }

    });

    var imageEdits = {};

    var lastParameterValue = null;

    function updateEdit(imageIndex, editType, parameter) {

        // You may want to store these values to be used when applying the crop

        if (!imageEdits[imageIndex]) {

            // Initialize default values if not already set

            imageEdits[imageIndex] = {

                zoom: 0,

                rotate: 0,

                editType: null,

                parameters: null

            };

        }

        if (editType === 'zoom') {

            // Increment or decrement the zoom level

            imageEdits[imageIndex].zoom += (parameter === 'in' ? 0.1 : -0.1);

        } else if (editType === 'rotate') {

            // Increment or decrement the rotation angle

            imageEdits[imageIndex].rotate += (parameter === '45' ? 45 : -45);

        }

        imageEdits[imageIndex].editType = editType;

        imageEdits[imageIndex].parameters = parameter;

        // Update the last edit type and parameter

        lastEditType = editType;

        lastParameterValue = imageEdits[imageIndex][editType];

    }

    $('#wp2023-apply-crop').on('click', function() {

        if (currentCropper && typeof currentImageIndex !== 'undefined') {

            var croppedCanvas = currentCropper.getCroppedCanvas();

            var marginPixels = 1.5 \* 37.7952755906; // Convert cm to pixels

            // Calculate new dimensions for scaling the image down

            var scaledWidth = croppedCanvas.width - (2 \* marginPixels);

            var scaledHeight = croppedCanvas.height - (2 \* marginPixels);

            // Create a canvas for the scaled-down image

            var scaledCanvas = document.createElement('canvas');

            scaledCanvas.width = scaledWidth;

            scaledCanvas.height = scaledHeight;

            var scaledCtx = scaledCanvas.getContext('2d');

            // Draw the scaled-down image (shrink to add margin)

            scaledCtx.drawImage(croppedCanvas, 0, 0, croppedCanvas.width, croppedCanvas.height, 0, 0,

                scaledWidth, scaledHeight);

            // Create a canvas with the original dimensions (to include margin)

            var canvasWithMargin = document.createElement('canvas');

            canvasWithMargin.width = croppedCanvas.width;

            canvasWithMargin.height = croppedCanvas.height;

            var ctx = canvasWithMargin.getContext('2d');

            ctx.fillStyle = "#ffffff"; // Margin color (white)

            ctx.fillRect(0, 0, canvasWithMargin.width, canvasWithMargin.height); // Fill the background

            // Draw the scaled-down image onto the center of the canvas with margins

            ctx.drawImage(scaledCanvas, marginPixels, marginPixels);

            var currentEdits = imageEdits[currentImageIndex] || {};

            // Capture the current index before incrementing

            let capturedIndex = currentImageIndex;

            // Convert canvas to Blob and update the validFiles array

            canvasWithMargin.toBlob(function(blob) {

                croppedImages.push({

                    blob: blob,

                    index: capturedIndex

                });

                // Optionally, update the thumbnail preview

                updateThumbnailPreview(canvasWithMargin, capturedIndex);

            }, 'image/png'); // Adjust the MIME type as needed

            if (capturedIndex < imageFiles.length - 1) {

                currentImageIndex++;

                startEditingImage(currentImageIndex); // Start editing the next image

            } else {

                // All images have been processed, now you can close the modal and clean up

                $('#image-edit-modal').hide();

                if (currentCropper) {

                    currentCropper.destroy();

                    currentCropper = null;

                }

                currentImageIndex = null; // Reset the index or set it to the initial state

                // console.log('All images have been edited.');

            }

        }

    });

    function updateThumbnailPreview(croppedCanvas, index) {

        // Convert the cropped canvas to a data URL

        var dataUrl = croppedCanvas.toDataURL();

        // Create an image element

        var img = **new** Image();

        img.src = dataUrl;

        img.style = 'max-width: 160px; margin-right: 10px;'; // Set any required styles

        img.classList.add('cropped-thumbnail');

        img.dataset.index = index; // Set a data attribute for the index

        // Append the new image to the thumbnails container

        var container = document.getElementById('wp2023-image-thumbnails-container');

        container.appendChild(img);

    }

    // Close button to hide the modal and destroy the cropper instance

    $('.close').on('click', function() {

        var modal = $('#image-edit-modal');

        modal.hide();

        if (currentCropper) {

            currentCropper.destroy();

            currentCropper = null;

        }

    });

    $('.plus-btn').click(function(e) {

        e.preventDefault();

        var quantity = parseInt($('#quantity').val());

        $('#quantity').val(quantity + 1);

    });

    $('.minus-btn').click(function(e) {

        e.preventDefault();

        var quantity = parseInt($('#quantity').val());

        if (quantity > 1) {

            $('#quantity').val(quantity - 1);

        }

    });

    var isUploading = false;

    $('#image-upload-form').on('submit', function(e) {

        e.preventDefault();

        e.stopPropagation();

        var userPhoneNum = $('#userPhoneNumber').val().trim();

        var engravedContent = '';

        var quantity = $('#quantity').val();

        if (isKhungKhacLaser) {

            engravedContent = $('#engraved-content').val().trim();

            if (!engravedContent) {

                $('#engravedError').text('Please enter your engraved content.').show();

                return;

            } else {

                $('#engravedError').hide();

            }

        }

        // Check if userPhoneNum is empty

        if (!userPhoneNum) {

            $('#phoneError').text('Please enter your phone number.').show(); // Show error message

            return; // Prevent form submission

        } else {

            $('#phoneError').hide();

        }

        if (croppedImages.length === 0) {

            // Notify the user that they need to select and crop an image

            $('#error-message-container').text('Please select and crop an image before submitting.')

                .show();

            return; // Prevent form submission

        } else {

            $('#error-message-container').hide();

        }

        if (isUploading) {

            // console.log('Upload is already in progress.');

            return; // Exit if an upload is in progress

        }

        var validFiles = $('#wp2023-custom-image')[0].files; // Assuming this is your file input

        if (validFiles.length === 0) {

            $('#error-message-container').text(

                    'o valid files to upload. Please check your files and try again.')

                .show();

            return; // No files to upload

        } else {

            $('#error-message-container').hide();

        }

        isUploading = true; // Set the flag to indicate upload is in progress

        var formData = **new** FormData(this);

        formData.append('action', 'handle\_image\_upload');

        formData.append('phoneNumber', jQuery('#userPhoneNumber').val());

        if (isKhungKhacLaser) {

            formData.append('engravedContent', jQuery('#engraved-content').val());

        }

        formData.append('size', selectedSize);

        formData.append('quantity', quantity);

        formData.append('product\_id', productId);

        formData.append('templateId', TemplateId);

        formData.append('selectedTemplateIndex', selectedTemplateIndex);

        // Check if the image has been cropped

        if (imageEdits[currentImageIndex]) {

            formData.append('editType', imageEdits[currentImageIndex].editType || '');

            formData.append('parameters', lastParameterValue || '{}');

        }

        croppedImages.forEach(function(croppedImage, index) {

            formData.append('croppedImage\_' + index, croppedImage.blob, 'cropped-image-' +

                index + '.png');

            formData.append('croppedImageIndex\_' + index, croppedImage.index);

        });

        submitForm(formData, function() {

            isUploading = false; // Reset the flag after upload

        });

        $(this).find(':submit').prop('disabled', true);

    });

    function submitForm(formData) {

        $('#loadingSpinner').show();

        $.ajax({

            url: '<?php echo admin\_url('admin-ajax.php'); ?>',

            type: 'POST',

            data: formData,

            contentType: false,

            processData: false,

            success: function(response) {

                updateDatabaseWithEditedImagePath(response.editedImagePath);

                if (response && response.data && response.data.shopeeLink) {

                    $('#loadingSpinner').hide();

                    window.location.href = response.data.shopeeLink;

                } else {

                    $('#loadingSpinner').hide();

                    showToast('Cảm ơn bạn đã mua hàng');

                    resetForm();

                }

            },

            error: function(xhr, status, error) {

                $('#loadingSpinner').hide();

                // Show error toast notification

                showToast('Bạn mua hàng không thành công: ');

                resetForm();

            }

        });

    }

    function showToast(message, duration = 5000) {

        // Create the toast message element

        var toastHTML = '<div class="toast-message">' + message + '</div>';

        var $toast = $(toastHTML);

        // Append toast message to the container

        $('#toast-container').append($toast);

        // Show the toast message with a fade-in effect

        $toast.fadeIn(400, function() {

            // After 'duration' milliseconds, start fade-out effect

            setTimeout(function() {

                $toast.fadeOut(400, function() {

                    // Remove the toast message element after fade-out

                    $toast.remove();

                });

            }, duration);

        });

    }

    function resetForm() {

        $('#image-upload-form')[0].reset();

        $('#image-upload-form').find(':submit').prop('disabled', false);

        isUploading = false;

        var thumbnailContainer = document.getElementById('wp2023-image-thumbnails-container');

        while (thumbnailContainer.firstChild) {

            thumbnailContainer.removeChild(thumbnailContainer.firstChild);

        }

        // Clear the croppedImages array

        croppedImages = []; // Reset the array to empty

    }

    function updateDatabaseWithEditedImagePath(editedImagePath) {

        var data = {

            action: 'save\_edited\_image\_path',

            editedImagePath: editedImagePath

            // Add other necessary data like ImageId, CustomerId etc.

        };

        $.post('<?php echo admin\_url('admin-ajax.php'); ?>', data, function(response) {});

    }

    function displayError(errorMessage) {

        var errorContainer = $('#error-message-container');

        if (errorContainer.length) {

            errorContainer.text(errorMessage).show();

        } else {

            // If the container doesn't exist, create it or log the error to the console

            console.error("Error container not found on the page.");

        }

    }

    function showLoadingIndicator() {

        // Show loading indicator

        $('#loading-indicator').show();

    }

    function hideLoadingIndicator() {

        // Hide loading indicator

        $('#loading-indicator').hide();

    }

});

</script>

<?php

    return ob\_get\_clean();

}

add\_shortcode('tabletop\_frame\_template', 'wp2023\_tabletop\_frame\_template');

function handle\_image\_upload() {

    global $wpdb;

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

    $shopee\_link = get\_post\_meta($productId, 'link\_shopee', true);

    $phoneNumber = isset($\_POST['phoneNumber']) ? sanitize\_text\_field($\_POST['phoneNumber']) : '';

    $engravedContent = isset($\_POST['engravedContent']) ? sanitize\_text\_field($\_POST['engravedContent']) : '';

     // Retrieve the size from the POST data

    $sizeImage = isset($\_POST['size']) ? sanitize\_text\_field($\_POST['size']) : '';

    $quantity = isset($\_POST['quantity']) ? sanitize\_text\_field($\_POST['quantity']) : '';

    $customerId = 0;

    $templateId = isset($\_POST['templateId']) ? intval($\_POST['templateId']) : 0;

    $selectedTemplateIndex = isset($\_POST['selectedTemplateIndex']) ? intval($\_POST['selectedTemplateIndex']) : null;

    // Retrieve the full array of image templates

    $image\_templates = get\_post\_meta($productId, 'image\_templates\_urls', true);

    // If there is a valid selected template index, get the corresponding URL

    $selectedTemplateURL = null;

    if ($selectedTemplateIndex !== null && is\_array($image\_templates) && isset($image\_templates[$selectedTemplateIndex])) {

        $selectedTemplateURL = $image\_templates[$selectedTemplateIndex];

    }

    if (empty($sizeImage)) {

        wp\_send\_json\_error(['message' => 'Size parameter is missing.']);

        wp\_die();

    }

 // Save full name to customers table and get the customer ID

    if (!empty($phoneNumber)) {

        $inserted = $wpdb->insert(

            "{$wpdb->prefix}wp2023\_customers",

            array('PhoneNumber' => $phoneNumber),

            array('%s')

        );

        if ($inserted) {

            $customerId = $wpdb->insert\_id;

        } else {

            // Log the error and send a JSON error response

            error\_log('Failed to insert customer: ' . $wpdb->last\_error);

            wp\_send\_json\_error(['message' => 'Failed to save customer information']);

            wp\_die();

        }

    }

    $response\_messages = [];

    $upload\_overrides = array('test\_form' => false);

    // Handle original images

    $originalImageFilePaths = []; // Store file paths of the original images

    if (isset($\_FILES['wp2023-custom-image']) && is\_array($\_FILES['wp2023-custom-image']['name'])) {

        $file\_count = count($\_FILES['wp2023-custom-image']['name']);

        for ($i = 0; $i < $file\_count; $i++) {

            $file = [

                'name'     => $\_FILES['wp2023-custom-image']['name'][$i],

                'type'     => $\_FILES['wp2023-custom-image']['type'][$i],

                'tmp\_name' => $\_FILES['wp2023-custom-image']['tmp\_name'][$i],

                'error'    => $\_FILES['wp2023-custom-image']['error'][$i],

                'size'     => $\_FILES['wp2023-custom-image']['size'][$i],

            ];

            // Use the WordPress function to handle the upload

            $moveOriginalFile = wp\_handle\_upload($file, $upload\_overrides);

            if ($moveOriginalFile && !isset($moveOriginalFile['error'])) {

                $originalImagePath = $moveOriginalFile['url']; // URL of the uploaded file

                $originalImageFilePaths[] = $moveOriginalFile['file']; // Save the file path for later deletion

                $result = $wpdb->insert(

                    "{$wpdb->prefix}wp2023\_images",

                    array(

                        'OriginalImagePath' => $originalImagePath,

                        'UploadDate' => current\_time('mysql'),

                        'Status' => 'uploaded',

                        'SizeImage' => $sizeImage,

                        'Quantity' => $quantity,

                        'PdfInformation' => $engravedContent,

                        'SelectedTemplate' => $selectedTemplateURL,

                        'CustomerId' => $customerId,

                        'ProductId' =>  $productId,

                        'TemplateId' => $templateId

                        // Set Status, EditType, Parameters as needed, or use defaults

                    ),

                    array(

                        '%s',

                        '%s',

                        '%s',

                        '%s',

                        '%d',

                        '%s',

                        '%s',

                        '%d',

                        '%d',

                        '%d',

                        // ... other formats as necessary ...

                    )

                );

                if (false === $result) {

                    // Handle error, insert failed

                    $response\_messages[] = 'Failed to insert original image URL into the database.';

                } else {

                    $response\_messages[] = 'Original image URL inserted into the database successfully.';

                }

                $response\_messages[] = 'Original file uploaded successfully.';

            } else {

                // Handle errors

                $response\_messages[] = 'Error uploading original file: ' . $moveOriginalFile['error'];

            }

        }

    }

    foreach ($\_FILES as $key => $file) {

        if (strpos($key, 'croppedImage\_') === 0) {

            $imageIndex = intval($\_POST['imageIndex\_' . substr($key, strlen('croppedImage\_'))]);

            $editType = isset($\_POST['editType\_' . $imageIndex]) ? $\_POST['editType\_' . $imageIndex] : null;

            $parameters = isset($\_POST['parameters\_' . $imageIndex]) ? $\_POST['parameters\_' . $imageIndex] : null;

            // Use the WordPress function to handle the upload

            $upload\_overrides = array('test\_form' => false);

            $moveCroppedFile = wp\_handle\_upload($file, $upload\_overrides);

            if ($moveCroppedFile && !isset($moveCroppedFile['error'])) {

                $editedImagePath = $moveCroppedFile['url']; // URL of the uploaded file

                // Insert into database

                $result = $wpdb->insert(

                    "{$wpdb->prefix}wp2023\_images",

                    array(

                        'EditedImagePath' => $editedImagePath,

                        'UploadDate' => current\_time('mysql'),

                        'Status' => 'edited',

                        'EditType' => $editType,

                        'Parameters' => $parameters,

                        'SizeImage' => $sizeImage,

                        'Quantity' => $quantity,

                        'PdfInformation' => $engravedContent,

                        'SelectedTemplate' => $selectedTemplateURL,

                        'CustomerId' => $customerId,

                        'ProductId' => $productId,

                        'TemplateId' => $templateId,

                        // Set other fields as needed

                    ),

                    array(

                        '%s',

                        '%s',

                        '%s',

                        '%s', // EditType

                        '%s', // Parameters

                        '%s',

                        '%d',

                        '%s',

                        '%s',

                        '%d',

                        '%d',

                        '%d',

                        // ... other formats ...

                    )

                );

                if (false === $result) {

                    $response\_messages[] = 'Failed to insert edited image URL into the database.';

                } else {

                    $response\_messages[] = 'Edited image URL inserted into the database successfully.';

                }

            } else {

                $response\_messages[] = 'Error uploading edited file: ' . $moveCroppedFile['error'];

            }

        }

    }

    // Check if the image upload and database insertion are successful

    if (!in\_array('Failed to insert original image URL into the database.', $response\_messages) && !in\_array('Failed to insert edited image URL into the database.', $response\_messages)) {

        // Compose the email

        $to = get\_option('admin\_email'); // Gets the admin email from WordPress settings

        $subject = 'New Image Upload Notification';

        $body = 'A new image has been uploaded. Phone number: ' . $phoneNumber;

        // Attempt to send the email and log the result

        if (wp\_mail($to, $subject, $body)) {

            error\_log('Email sent successfully to ' . $to);

        } else {

            error\_log('Failed to send email to ' . $to);

        }

    }

     // Return response messages or handle as needed

    wp\_send\_json\_success(['messages' => $response\_messages, 'shopeeLink' => $shopee\_link]);

    wp\_die();

}

add\_action('wp\_ajax\_handle\_image\_upload', 'handle\_image\_upload');

add\_action('wp\_ajax\_nopriv\_handle\_image\_upload', 'handle\_image\_upload');

remove\_action('woocommerce\_after\_shop\_loop\_item', 'woocommerce\_template\_loop\_add\_to\_cart', 10);

remove\_action('woocommerce\_single\_product\_summary', 'woocommerce\_template\_single\_add\_to\_cart', 30);