Plugin files :
total 76K
-rwxr-xr-x 1 www-data www-data 19 Sep 30 05:09 README.md
drwxr-xr-x 4 www-data www-data 4.0K Sep 30 05:09 assets
        drwxr-xr-x 2 www-data www-data 4.0K Sep 30 05:09 assets/css
        drwxr-xr-x 2 www-data www-data 4.0K Sep 30 05:09 assets/imgs
        -rwxr-xr-x 1 www-data www-data 30K Sep 30 05:09 assets/logo.png
-rwxr-xr-x 1 www-data www-data 1.8K Sep 30 05:09 elementor-custom-widget.php
-rwxr-xr-x 1 www-data www-data 2.5K Sep 30 05:09 elementor-widget.php
-rwxr-xr-x 1 www-data www-data 5.2K Sep 30 05:09 gutenberg-button.js
-rwxr-xr-x 1 www-data www-data 985 Sep 30 05:09 gutenberg-button.php
-rwxr-xr-x 1 www-data www-data 2.0K Sep 30 05:09 mce-button.js
-rwxr-xr-x 1 www-data www-data 4.5K Sep 30 05:09 mce-button.php
-rwxr-xr-x 1 www-data www-data 31K Sep 30 05:09 openai-dalle-plugin.php
-rwxr-xr-x 1 www-data www-data 214 Sep 30 05:09 wdiget.js

Elementor-custom-widget.php

```php
<?php
namespace YourNamespace;
use Elementor\Widget_Base;
use Elementor\Controls_Manager;

if ( ! defined( 'ABSPATH' ) ) {
exit; // Exit if accessed directly.
}


class Your_Custom_Widget
{

public function get_name() {
return 'your-custom-widget';
}

public function get_title() {
return __( 'Your Custom Widget', 'your-text-domain' );
}

public function get_icon() {
return 'eicon-button'; // Use Elementor icons or your custom icons
}
```

```php
public function get_categories() {
return [ 'basic' ]; // Assign the widget to a category
}

protected function _register_controls() {
$this->start_controls_section(
'section_options',
[
'label' => __( 'Options', 'your-text-domain' ),
]
);

$this->add_control(
'text_input',
[
'label' => __( 'Text Input', 'your-text-domain' ),
'type' => Controls_Manager::TEXT,
'default' => '',
]
);

$this->add_control(
'action',
[
'label' => __( 'Action', 'your-text-domain' ),
'type' => Controls_Manager::SELECT,
'options' => [
'rewrite' => __( 'Rewrite', 'your-text-domain' ),
'replace' => __( 'Replace', 'your-text-domain' ),
'correct' => __( 'Correct', 'your-text-domain' ),
],
'default' => 'rewrite',
]
);

$this->end_controls_section();
}

protected function render() {
$settings = $this->get_settings_for_display();
```

```
?>
<div class="your-custom-widget">
<button class="open-modal"><?php _e( 'Open Modal', 'your-text-domain' );
?></button>
<div class="modal" style="display:none;">
<div class="modal-content">
<span class="close">&times;</span>
<p><?php echo $settings['text_input']; ?></p>
<p><?php echo $settings['action']; ?></p>
</div>
</div>
</div>
<?php
}

}
```

elementor-widget.php

```
<?php
class OpenAI_DALLE_Elementor_Widget extends \Elementor\Widget_Base {
public function get_name() {
return 'openai-dalle-widget';
}

public function get_title() {
return 'Muse Elementor';
}

public function get_icon() {
return 'fa fa-magic';
}

public function get_categories() {
return [ 'general' ];
}
```

```
protected function render() {
$prompt = '';
$image_prompt = '';

// Call functions to generate content using OpenAI and DALL-E APIs
$generated_text = generate_openai_content($prompt);
$generated_image = generate_dalle_image($image_prompt);

// Format the generated content
$text_parts = explode("\n", $generated_text, 2);
$content = $text_parts[0];
if (isset($generated_image)) {
$content .= '<br><img src="' . esc_url($generated_image) . '"
alt="Generated Image" style="display: block; margin-left: auto;
margin-right: auto; width: 50%;">';
}
$content .= $text_parts[1];

echo $content;
}
}
function openai_dalle_plugin_elementor_button() {
if( ! current_user_can('edit_posts') && ! current_user_can('edit_pages') )
{
return;
}

if( get_user_option('rich_editing') !== 'true' ) {
return;
}

add_action( 'elementor/editor/after_enqueue_styles', function() {
wp_enqueue_style( 'openai-dalle-elementor-button-style',
plugins_url('css/elementor-button.css', __FILE__) );
} );

add_action( 'elementor/editor/footer', function() {
?>
<script>
```

```
elementor.hooks.addAction( 'panel/openai-dalle', function( panel, model,
view ) {
var button = jQuery( '<a>', {
class: 'elementor-button elementor-button-success',
text: 'Muse Elementor',
click: function() {
window.location.href = '<?php echo
admin_url('admin.php?page=openai-dalle-plugin'); ?>';
},
} );
view.$el.find( '.elementor-panel-footer-buttons' ).append( button );
} );
</script>
<?php
} );
}
add_action( 'elementor/loaded', 'openai_dalle_plugin_elementor_button' );

function openai_dalle_elementor_widgets() {
\Elementor\Plugin::instance()->widgets_manager->register_widget_type(new
OpenAI_DALLE_Elementor_Widget());
}
add_action('elementor/widgets/widgets_registered',
'openai_dalle_elementor_widgets');
```

Gutenberg-button.js

```
const { registerPlugin } = wp.plugins;
const { PluginSidebar, PluginSidebarMoreMenuItem } = wp.editPost;
const { PanelBody, Button, TextareaControl,Panel,ToolbarGroup,
ToolbarButton } = wp.components;
const { useState,Fragment } = wp.element;
const { dispatch, select } = wp.data;

const AiMuseSidebar = () => {
const [responseText, setResponseText] = useState('');

const options = [
{ label: 'Elaborate', value: 'develop' },
```

```
{ label: 'Resume', value: 'summarize' },
{ label: 'Complete', value: 'complete' },
];
const imgs_options = [
{ label: 'Replace', value: 'replace' },
{ label: 'Generate with text', value: 'generate' },
];
const handleMenuClick = (selectedOption) => {
console.log('Option clicked:', selectedOption);

// const selectedText = select('core/editor').getSelectedBlock();
const selectedText = window.getSelection().toString();

if (!selectedText) {
console.log('No text selected!');
alert('Please select some text first.');
return;
}

console.log(`Selected text: ${selectedText} ${selectedOption.label}`);
callOpenAI(selectedOption.label, selectedText)
.then((generatedText) => {
setResponseText(generatedText);
console.log(generatedText);
})
.catch((error) => {
console.error('Error:', error);
});
// Perform the desired action using the OpenAI API
// You need to modify the following line to call the appropriate function
callOpenAI(selectedOption.value, selectedText).then((generatedText) => {
setResponseText(generatedText);
});

// const processedText = `This is a sample processed text for
demonstration purposes. ${selectedOption.label}`;

};
async function callOpenAI(prompt, text) {
```

```
// fetchOptionValue('api_key').then((newApiKey) => {
// setApiKey(newApiKey)
// });
const apiKey = openaiData.api_key
// console.log('OpenAI API Key:', apiKey);
const apiUrl = 'https://api.openai.com/v1/completions';
const promptText = `${prompt} this ${text}`;
const requestBody = {
prompt: promptText,
model: 'text-davinci-003',
max_tokens: 4000,
n: 1,
stop: null,
temperature: 0.7,
top_p: 1,
frequency_penalty: 0.0,
presence_penalty: 0.6,
};
const response = await fetch(apiUrl, {
method: 'POST',
headers: {
'Content-Type': 'application/json',
'Authorization': `Bearer ${apiKey}`,
},
body: JSON.stringify(requestBody),
});
if (response.ok) {
const responseBody = await response.json();
const generatedText = responseBody.choices[0].text;
return generatedText;
} else {
throw new Error(`API request failed: ${response.status}`);
}
}
const handleApply = () => {
const selectedText = window.getSelection().toString();
if (!selectedText) {
console.log('No text selected!');
alert('Please select some text first.');
return;
```

```
}
document.execCommand('insertText', false, responseText);
};

const buttons = options.map((option) =>
React.createElement(Button, {
key: option.value,
isSecondary: true,
onClick: () => handleMenuClick(option),
}, option.label)
);
const img_buttons = imgs_options.map((option) =>
React.createElement(Button, {
key: option.value,
isSecondary: true,
onClick: () => handleMenuClick(option),
}, option.label)
);
return React.createElement(
React.Fragment,
null,
React.createElement(PluginSidebarMoreMenuItem, { target: 'ai-muse-sidebar'
}, 'AI Muse'),
React.createElement(
PluginSidebar,
{ name: 'ai-muse-sidebar', title: 'AI Muse' },
React.createElement(
Panel,
{ className: 'my-dropdown-menu' },
React.createElement(PanelBody, {
title: 'Content',
initialOpen: false,
},
React.createElement(PanelBody, { className: 'editor-post-excerpt' },
buttons),
React.createElement(TextareaControl, {
className: 'components-textarea-control__input css-1cxopg7 e1w5nnrk0',
label: 'API Response',
value: responseText,
onChange: (value) => setResponseText(value),
```

```
}),
React.createElement(Button, { className: 'components-button is-primary',
onClick: handleApply }, 'Apply')
),
React.createElement(PanelBody, {
title: 'Images',
initialOpen: false,
},
React.createElement(PanelBody, { className: 'editor-post-excerpt' },
img_buttons),
React.createElement(TextareaControl, {
className: 'components-textarea-control__input css-1cxopg7 e1w5nnrk0',
label: 'API Response',
value: responseText,
onChange: (value) => setResponseText(value),
}),
React.createElement(Button, { className: 'components-button is-primary',
onClick: handleApply }, 'Apply')
)
)
)

);
};



registerPlugin('ai-muse-button', {
render: () => {
console.log('In render!'); // Verify that this function is called
return React.createElement(AiMuseSidebar);
},
});
```

Gutenberg-button.php

```
<?php

// gutenberg
function ai_muse_enqueue_gutenberg_scripts() {
```

```
// Get the option value
$api_key = get_option('openai_api_key');
// Pass the option value to the JavaScript file
wp_enqueue_script(
'ai-muse-gutenberg-button',
plugin_dir_url(__FILE__) . 'gutenberg-button.js',
array('wp-plugins', 'wp-edit-post', 'wp-element', 'wp-data'),
'1.0',
true
);
// Pass the API key to the script
wp_localize_script('ai-muse-gutenberg-button', 'openaiData', array(
'api_key' => get_option('openai_api_key'),
));
}

add_action('admin_enqueue_scripts', 'ai_muse_enqueue_gutenberg_scripts');

// Include external CSS
function ai_muse_enqueue_styles() {
wp_enqueue_style(
'ai-muse-button-styles',
plugin_dir_url(__FILE__) . 'assets/css/gutenberg-styles.css',
array(),
'1.0',
'all'
);
}
add_action('enqueue_block_editor_assets', 'ai_muse_enqueue_styles');
```

Mce-button.js

```
document.addEventListener("DOMContentLoaded", function() {
console.log("inside mce-button");

let submitButton = document.getElementById("submit-btn");
// Change background color and text color on hover
submitButton.addEventListener("mouseover", function() {
this.style.backgroundColor = "#4a09ca";
this.style.color = "#fff";
```

```
this.style.borderRadius = "30%";
});
// Reset background color and text color on mouseout
submitButton.addEventListener("mouseout", function() {
this.style.backgroundColor = "";
this.style.color = "";
this.style.borderRadius = "";
});

// Hide button on click
submitButton.addEventListener("click", function() {
this.style.display = "none";
});
});

document.addEventListener("DOMContentLoaded", function() {
console.log("inside TinyMCE plugin");

// Get the switch and settings-card elements
var switchElement = document.getElementById("switch");
var settingsCard = document.querySelector(".settings-card");

// Add event listener for the switch
switchElement.addEventListener("change", function() {
if (switchElement.checked) {
// If the switch is checked, set the opacity to 1
settingsCard.style.opacity = 1;
} else {
// If the switch is not checked, set the opacity to 0.2
settingsCard.style.opacity = 0.2;
}
});
});

function ai_muse_ajax_action(action) {
// ... (rest of the ai_muse_ajax_action function)
}

(function() {
tinymce.PluginManager.add('ai_muse_mce_button', function(editor, url) {
```

```
// ... (rest of the tinymce.PluginManager.add code)
});
})();

/* Create a new intersection observer instance */
const observer = new IntersectionObserver(entries => {
// ... (rest of the intersection observer code)
});
/* Observe the section */
observer.observe(document.querySelector('.separator'));
```

Mce-button.php

```
<?php
// // AI MUSE BUTTON TINYMCE
function ai_muse_enqueue_plugin_scripts($plugin_array) {
// Enqueue the JavaScript file
wp_enqueue_script('ai_muse_mce_button', plugin_dir_url(__FILE__) .
'mce-button.js', array('jquery'), '1.0', true);

// Add the script to the TinyMCE plugins array
$plugin_array['ai_muse_mce_button'] = plugin_dir_url(__FILE__) .
'mce-button.js';

return $plugin_array;
}

function ai_muse_ajax_callback() {
// Define ajaxData object
$ajax_data = array(
'ajax_url' => admin_url( 'admin-ajax.php' ),
'ajax_nonce' => wp_create_nonce( 'ai_muse_ajax_nonce' ),
);

// Send ajaxData as JSON response
wp_send_json( $ajax_data );
}
add_action( 'wp_ajax_ai_muse_ajax_callback', 'ai_muse_ajax_callback' );

function ai_muse_localize_scripts() {
```

```
// Define ajaxData object
$ajax_data = array(
'ajax_url' => admin_url( 'admin-ajax.php' ),
'ajax_nonce' => wp_create_nonce( 'ai_muse_ajax_nonce' ),
);

// Localize the ai_muse_mce_button script with ajaxData object
wp_localize_script( 'ai_muse_mce_button', 'ajaxData', $ajax_data );

// Add inline JavaScript code with localized data
$inline_script = 'var ajaxData = ' . json_encode( $ajax_data ) . ';';
wp_add_inline_script( 'ai_muse_mce_button', $inline_script );
}
add_action( 'admin_enqueue_scripts', 'ai_muse_localize_scripts' );

function ai_muse_register_mce_button($buttons) {
array_push($buttons, 'ai_muse_mce_button');
return $buttons;
}
function ai_muse_add_mce_button() {
if (current_user_can('edit_posts') && current_user_can('edit_pages')) {
add_filter('mce_external_plugins', 'ai_muse_enqueue_plugin_scripts');
add_filter('mce_buttons', 'ai_muse_register_mce_button');
}
}
function ai_muse_process_text() {
check_ajax_referer('ai_muse_ajax_nonce', 'security');

$ai_action = sanitize_text_field($_POST['ai_action']);
$text = sanitize_textarea_field($_POST['text']);

// Perform the desired action using the OpenAI API
$processed_text = ai_muse_call_openai_api($ai_action, $text);

echo $processed_text;
wp_die();
}

add_action('wp_ajax_ai_muse_process_text', 'ai_muse_process_text');
add_action('admin_init', 'ai_muse_add_mce_button');
```

```php
function ai_muse_call_openai_api($ai_action, $text) {
// Call the OpenAI API here based on the $ai_action and $text

$api_key = get_option('openai_api_key');
$api_url = 'https://api.openai.com/v1/completions';

$headers = array(
'Content-Type' => 'application/json',
'Authorization' => 'Bearer ' . $api_key
);
$prompt = '';

switch ($ai_action) {
case 'summarize':
$prompt = "Summarize the following text: {$text}";
break;
case 'write_paragraphs':
$prompt = "Write complete paragraphs for the following text: {$text}";
break;
case 'paraphrase':
$prompt = "Paraphrase the following text: {$text}";
break;
case 'change_tone':
$prompt = "Rewrite the following text in a more formal tone: {$text}";
break;
}

$data = array(
'engine' => 'davinci-codex',
'prompt' => $prompt,
'max_tokens' => 4000,
'temperature' => 0.8
);

$headers = array(
'Content-Type: application/json',
'Authorization: Bearer ' . $api_key
);
```

```
$response = wp_remote_post($api_url, array(
'headers' => $headers,
'body' => json_encode($body),
'timeout' => 120,
));
// Check if the request was successful
if (is_wp_error($response)) {
// Handle the error
// echo "Error: " . $response->get_error_message();
} else {
// If the request was successful, print the response body
$response_body = wp_remote_retrieve_body($response);
// echo $response_body;
}

$response_data = json_decode(wp_remote_retrieve_body($response), true);

if (isset($response_data['choices']) &&
isset($response_data['choices'][0]) &&
isset($response_data['choices'][0]['text'])) {
return $response_data['choices'][0]['text'];
}

return 'Error: Unable to process the text.';
}

// Include external CSS
function ai_muse_enqueue_mce_styles() {
wp_enqueue_style(
'ai-muse-button-styles',
plugin_dir_url(__FILE__) . 'assets/css/mce-styles.css',
array(),
'1.0',
'all'
);
}
add_action('admin_enqueue_scripts', 'ai_muse_enqueue_mce_styles');
```

Openai-dalle-plugin.php

```php
<?php
/**
* Plugin Name: AI Muse Content Generator
* Plugin URI: https://facile.codes/ai-muse
* Description: A plugin that generates content using OpenAI API and images
using DALL-E API,.
* Version: 1.1
* Author: facile.codes
* Author URI: https://facile.codes
* License: GPL2
*/

include_once plugin_dir_path( __FILE__ ) . 'mce-button.php';
include_once plugin_dir_path( __FILE__ ) . 'gutenberg-button.php';



// Check if Elementor is installed and active
if (did_action('elementor/loaded')) {
add_action('elementor/widgets/widgets_registered', function
($widgets_manager) {
$widgets_manager->register_widget_type(new
YourNamespace\Your_Custom_Widget());
});

// Enqueue the custom widget JavaScript file
add_action('elementor/frontend/after_enqueue_scripts',
'enqueue_your_custom_widget_scripts');
}

function enqueue_your_custom_widget_scripts()
{
wp_enqueue_script(
'your-custom-widget-js',
plugin_dir_url(__FILE__) . 'widget.js',
['jquery'],
'1.0.0',
true
);
```

```
}

function openai_dalle_plugin_menu() {
add_menu_page(
'AI Muse Content Generator',
'AI Muse',
'manage_options',
'openai-dalle-plugin',
'openai_dalle_plugin_page'
);

add_submenu_page(
'openai-dalle-plugin',
'AI Muse Chat',
'Chat',
'manage_options',
'openai-dalle-chat',
'openai_dalle_plugin_chat_page'
);
add_submenu_page(
'openai-dalle-plugin',
'AI Muse Documenation',
'Documentation',
'manage_options',
'openai-dalle-documentation',
'openai_dalle_plugin_documentation_page'
);
add_submenu_page(
'openai-dalle-plugin',
'AI Muse Settings',
'Settings',
'manage_options',
'openai-dalle-settings',
'openai_dalle_plugin_settings_page'
);
}
add_action('admin_menu', 'openai_dalle_plugin_menu');

// The callback function for the REST API route
function my_get_option(WP_REST_Request $request) {
```

```
// Retrieve the 'option_name' parameter from the request
$option_name = $request->get_param('option_name');
// Get the option value using the 'option_name'
$option_value = get_option($option_name);
// Return the option value in a REST response
return new WP_REST_Response($option_value, 200);
}
/// SHORTCODE
function openai_dalle_shortcode($atts) {
$prompt = isset($atts['prompt']) ? sanitize_text_field($atts['prompt']) :
'';
$image_prompt = isset($atts['image_prompt']) ?
sanitize_text_field($atts['image_prompt']) : '';

// Call functions to generate content using AI Muse APIs
$generated_text = generate_openai_content($prompt);
$generated_image = generate_dalle_image($image_prompt);

// Format the generated content
$text_parts = explode("\n", $generated_text, 2);
$content = $text_parts[0];
if (isset($generated_image)) {
$content .= '<br><img src="' . esc_url($generated_image) . '"
alt="Generated Image" style="display: block; margin-left: auto;
margin-right: auto; width: 50%;">';
}
$content .= $text_parts[1];

return $content;
}
add_shortcode('openai_dalle', 'openai_dalle_shortcode');
// The button on the top of the editor
function add_openai_dalle_button() {
$screen = get_current_screen();
if ($screen->id != 'edit-post') {
return;
}
?>
<style>
.openai-dalle-button {
```

```
background: #0073aa;
border-color: #0073aa;
color: #fff;
padding: 4px 10px;
text-align: center;
display: inline-block;
margin: 0 5px;
vertical-align: top;
position: relative;
text-decoration: none;
border-radius: 2px;
cursor: pointer;
line-height: 24px;
}
.openai-dalle-button:hover {
background: #008ec2;
border-color: #008ec2;
}
</style>
<script>
document.addEventListener('DOMContentLoaded', function() {
var addButton = document.createElement('a');
addButton.className = 'openai-dalle-button';
addButton.innerHTML = 'Generate with AI Muse';
addButton.onclick = function() {
window.location.href = '<?php echo
admin_url('admin.php?page=openai-dalle-plugin'); ?>';
};

var wpBodyContent = document.getElementById('wpbody-content');
if (wpBodyContent) {
var subsubsub = wpBodyContent.querySelector('.subsubsub');
if (subsubsub) {
subsubsub.insertAdjacentElement('afterend', addButton);
}
}
});

</script>
<?php
```

```
}
add_action('admin_footer-edit.php', 'add_openai_dalle_button');


function openai_dalle_plugin_page() {
$post_published = false;

if (isset($_POST['submit'])) {
$prompt = sanitize_text_field($_POST['prompt']);
$post_type = sanitize_text_field($_POST['post_type']);
$numberOfWords = sanitize_text_field($_POST['number_of_words']);

$image_prompt = sanitize_text_field($_POST['image_prompt']);
$type_of_image = sanitize_text_field($_POST['type_of_image']);
$numberOfImages = sanitize_text_field($_POST['number_of_images']);

// Call functions to generate content using AI Muse APIs
$generated_text =
generate_openai_content($prompt,$post_type,$numberOfWords);
$generated_images =
generate_dalle_image($image_prompt,$type_of_image,$numberOfImages);
}

if (isset($_POST['publish_post']) || isset($_POST['save_draft'])) {
$generated_content = stripslashes_deep($_POST['generated_content']);
$post_status = isset($_POST['publish_post']) ? 'publish' : 'draft';
// getting title
$post_title = wp_strip_all_tags($generated_content);
$max_length = 25;
if (strlen($post_title) > $max_length) {
$post_title = substr($post_title, 0, $max_length);
}
// Create the new post
$new_post = array(
'post_title' => $post_title,
'post_content' => $generated_content,
'post_status' => $post_status,
'post_author' => get_current_user_id(),
'post_type' => 'post',
);
```

```
// Insert the post into the database
wp_insert_post($new_post);

// Set the flag to true if the post is published
if ($post_status === 'publish') {
$post_published = true;
}
}

?>
<div class="container">
<!-- Header Navigation -->
<div class="openai-dalle-header-nav">
<?php
$image_folder_url = plugins_url('assets/', __FILE__);
// Replace 'example.jpg' with the name of the image you want to display
$image_url = $image_folder_url . 'logo.png';
?>
<a href="<?php echo home_url(); ?>"><img src="<?php echo
esc_url($image_url) ?>" alt="Logo"></a>
<ul class="openai-dalle-nav">
<li class="openai-dalle-nav-item">
<a href="<?php echo admin_url('admin.php?page=openai-dalle-plugin'); ?>"
class="openai-dalle-nav-link"><?php _e('Chat', 'openai-dalle'); ?></a>
</li>
<li class="openai-dalle-nav-item">
<a href="https://facile.codes/ai-muse"
class="openai-dalle-nav-link">Documentation</a>
</li>
<li class="openai-dalle-nav-item">
<a href="<?php echo admin_url('admin.php?page=openai-dalle-settings'); ?>"
class="openai-dalle-nav-link"><?php _e('Settings', 'openai-dalle'); ?></a>
</li>
<!-- Add more navigation items here -->
</ul>
</div>

<section class="nav">
<h1>AI MUSE</h1>
```

```
<br>
<h3 class="span loader"><span class="m">U</span><span
class="m">n</span><span class="m">l</span><span class="m">o</span><span
class="m">c</span><span class="m">k</span><span
class="m"> </span><span class="m">t</span><span
class="m">h</span><span class="m">e</span><span
class="m"> </span><span class="m">P</span><span
class="m">o</span><span class="m">w</span><span class="m">e</span><span
class="m">r</span><span class="m"> </span><span
class="m">o</span><span class="m">f</span><span
class="m"> </span><span class="m">A</span><span
class="m">I</span></h3>
<a href="#start">
<span></span>
<span></span>
<span></span>
<span></span>
let's go
</a>
</section>
<section class="separator">
<?php
$image3_url = $image_folder_url . 'imgs/revo3.png';
$image4_url = $image_folder_url . 'imgs/revo4.png';
$image5_url = $image_folder_url . 'imgs/revo5.png';
?>
<img src="<?php echo esc_url($image3_url) ?>">
<img src="<?php echo esc_url($image4_url) ?>">
<img src="<?php echo esc_url($image5_url) ?>">
</section>
<main class="main" id="start">
<section class="slider" id="tab-svelte">
<!-- Content -->
<div class="wrap">

<?php if ($post_published): ?>
<div id="message" class="updated notice notice-success is-dismissible">
<p><?php _e('Post published successfully.', 'openai-dalle'); ?></p>
</div>
<?php endif; ?>
```

```
<h2>AI Muse - Content Generator</h2>
<p>Generate content using OpenAI API and images using DALL-E API.</p>

<form method="post" action="">
<div class="settings-container">
<div class="settings-card">
<div class="card-title-container">
<h3>Text Settings</h3>
<input type="checkbox" id="switch" class="switch" checked/>
<label for="switch" class="switchtoggle">Toggle</label>
</div>

<table class="form-table">
<tr valign="top">
<th scope="row"><label>Idea:</label></th>
<td><input type="text" name="prompt" value="<?php echo
isset($_POST['prompt']) ? esc_attr($_POST['prompt']) : 'ferrari'; ?>"
required /></td>
</tr>
<tr valign="top">
<th scope="row"><label>Number of Words:</label></th>
<td><input type="number" name="number_of_words" min="1" max="5000"
value="<?php echo isset($_POST['number_of_words']) ?
intval($_POST['number_of_words']) : '666'; ?>" required /></td>
</tr>
<tr valign="top">
<th scope="row"><label>Post Type:</label></th>
<td>
<select name="post_type" required>
<option value="">--Select an option--</option>
<?php
$post_types = array(
'post' => 'Post',
'code' => 'Code',
'write_paragraphs' => 'Write Paragraph',
'change_tone' => 'Change Tone',
'summarize' => 'Summarize',
);
```

```
$selected_post_type = isset($_POST['post_type']) ? $_POST['post_type'] :
'';

foreach ($post_types as $key => $value) {
$selected = $selected_post_type === $key ? ' selected' : '';
echo "<option value='{$key}'{$selected}>{$value}</option>";
}
?>
</select>
</td>
</tr>
</table>
</div>
<div class="settings-card">
<div class="card-title-container">
<h3>Image Settings</h3>
<input type="checkbox" id="switch_2" class="switch" checked/>
<label for="switch_2" class="switchtoggle">Toggle</label>
</div>
<table class="form-table">
<tr valign="top">
<th scope="row"><label>Idea:</label></th>
<td><input type="text" name="image_prompt" value="<?php echo
isset($_POST['image_prompt']) ? esc_attr($_POST['image_prompt']) :
'ferrari'; ?>" required /></td>
</tr>
<tr valign="top">
<th scope="row"><label>Number of Images:</label></th>
<td><input type="number" name="number_of_images" min="1" max="4"
value="<?php echo isset($_POST['number_of_images']) ?
intval($_POST['number_of_images']) : '4'; ?>" required /></td>
</tr>
<tr valign="top">
<th scope="row"><label>Type of Image:</label></th>
<td>
<select name="type_of_image" required>
<option value="">--Select an option--</option>
<?php
$image_types = array(
'photograph' => 'Photograph',
```

```
'illustration' => 'Illustration',
'painting' => 'Painting',
'drawing' => 'Drawing',
'collage' => 'Collage',
'digital_art' => 'Digital Art',
);

$selected_image_type = isset($_POST['type_of_image']) ?
$_POST['type_of_image'] : '';

foreach ($image_types as $key => $value) {
$selected = $selected_image_type === $key ? ' selected' : '';
echo "<option value='{$key}'{$selected}>{$value}</option>";
}
?>
</select>
</td>
</tr>
<tr valign="top">
<th scope="row" colspan="2">
<!-- Loader -->

<svg version="1.1" id="loader" style="display:none;"
xmlns="http://www.w3.org/2000/svg"
xmlns:xlink="http://www.w3.org/1999/xlink" x="0px" y="0px"
viewBox="0 0 100 100" enable-background="new 0 0 100 100"
xml:space="preserve">
<rect fill="none" stroke="#4a09ca" stroke-width="4" x="25" y="25"
width="50" height="50">
<animateTransform
attributeName="transform"
dur="0.5s"
from="0 50 50"
to="180 50 50"
type="rotate"
id="strokeBox"
attributeType="XML"
begin="rectBox.end"/>
</rect>
<rect x="27" y="27" fill="#4a09ca" width="46" height="50">
```

```
<animate
attributeName="height"
dur="1.3s"
attributeType="XML"
from="50"
to="0"
id="rectBox"
fill="freeze"
begin="0s;strokeBox.end"/>
</rect>
</svg>
</th>
</tr>
</table>
</div>
</div>
<?php
// // Customizing the submit button
// $args = array(
// 'text' => "Let's go", // Change the text of the button
// 'class' => 'custom-button-class', // Add custom CSS class to the button
// 'type' => 'submit', // Set the type of the button, default is 'submit'
// );

// // Render the submit button with custom arguments
// submit_button( );


// ?>
<input type="submit" name="submit" id="submit" class="glory-btn"
value="Generate">
</form>
<br>
<div class="row">


<?php
if (isset($generated_text) || !empty($generated_images)) {
$content = '';
?>
```

```
<input type="text" name="generated_title" value="<?php echo
esc_attr($title); ?>" style="width:100%">
<?php
$paragraphs = array_filter(explode("\n", $generated_text), 'strlen');
$image_index = 0;
$num_images = count($generated_images);
$content = '';
foreach ($paragraphs as $index => $paragraph) {
$content .= $paragraph . '<br>';
if ($image_index < $num_images) {
$content .= '<img src="' . esc_url($generated_images[$image_index]) . '"
alt="Generated Image" style="display: block; margin-left: auto;
margin-right: auto; width: 50%;"><br>';
$image_index++;
}
}
wp_editor($content, 'generated_content_editor', array('textarea_name' =>
'generated_content', 'media_buttons' => false, 'textarea_rows' => 10));
?>
<form method="post" action="">
<input type="hidden" name="generated_content" value="<?php echo
esc_attr($content); ?>">
<input type="submit" name="publish_post" class="button button-primary"
value="Publish">
<input type="submit" name="save_draft" class="button button-secondary"
value="Save Draft">
<input type="button" name="preview_post" class="button button-secondary"
value="Preview"
onclick="window.open('/?p=preview','mywindow','width=900,height=600');">
</form>
<?php
}
?>
</div>
</form>


</section>
</main>
</div>
```

```php
<?php
}

function generate_openai_content($prompt,$type,$numberOfWords) {
$api_key = get_option('openai_api_key');
$api_url = 'https://api.openai.com/v1/completions';

$headers = array(
'Content-Type' => 'application/json',
'Authorization' => 'Bearer ' . $api_key
);
if($type=="post"){
$proprt = 'Write a blog ' . $type . ' in ' . $numberOfWords . ' word about
' . $prompt;
}else{
$proprt = $type . ' in ' . $numberOfWords . ' this ' . $prompt;
}
// echo $proprt;
$body = array(
'prompt' => $proprt,
'model' => 'text-davinci-003',
'max_tokens' => 4000, // Increase max tokens for longer articles
'n' => 1,
'stop' => null,
'temperature' => 0.7,
'top_p' => 1,
'frequency_penalty' => 0.0,
'presence_penalty' => 0.6,
// 'stop' => [" Human:", " AI:"],
);

$response = wp_remote_post($api_url, array(
'headers' => $headers,
'body' => json_encode($body),
'timeout' => 120,
));
// Check if the request was successful
if (is_wp_error($response)) {
// Handle the error
```

```
echo "Error: " . $response->get_error_message();
} else {
// If the request was successful, print the response body
$response_body = wp_remote_retrieve_body($response);
echo $response_body;
}

// if (is_wp_error($response)) {
// return false;
// }
$response_body = json_decode(wp_remote_retrieve_body($response), true);
// echo $response_body;
// echo '$response_body';

$generated_text = $response_body['choices'][0]['text'];
return $generated_text;
}
function generate_dalle_image($image_prompt,$type,$numberOfImages) {
$api_key = get_option('openai_api_key');
$api_url = 'https://api.openai.com/v1/images/generations';

$headers = array(
'Content-Type' => 'application/json',
'Authorization' => 'Bearer ' . $api_key
);
$proprt = 'generate a ' . $type . ' about ' . $image_prompt;

$body = array(
'prompt' => $proprt,
'num_images' => (int) $numberOfImages, // Generate 3 images
);

$response = wp_remote_post($api_url, array(
'headers' => $headers,
'body' => json_encode($body),
'timeout' => 120,
));
if (is_wp_error($response)) {
// Handle the error
// echo "Error: " . $response->get_error_message();
```

```
} else {
// If the request was successful, print the response body
$response_body = wp_remote_retrieve_body($response);
// echo $response_body;
}

$response_body = json_decode(wp_remote_retrieve_body($response), true);
$generated_image_urls = array();
foreach ($response_body['data'] as $image) {
// echo $image['url'];
// echo "looool";
// echo "looool";
// echo "looool";
$generated_image_urls[] = $image['url'];
}

return $generated_image_urls;
}

// Start Settings
/////////////////

function openai_dalle_plugin_settings_init() {
register_setting('openai_dalle_plugin_settings_group',
'openai_dalle_plugin_settings');

add_settings_section(
'openai_dalle_plugin_settings_section',
'API Settings',
'openai_dalle_plugin_settings_section_callback',
'openai_dalle_plugin_settings'
);
}


function openai_dalle_plugin_settings_section_callback() {
echo 'Enter your OpenAI API key below:';
}
function openai_dalle_plugin_settings_page() {
?>
```

```
<!-- Header Navigation -->
<div class="container">
<!-- Header Navigation -->
<div class="openai-dalle-header-nav">
<?php
$image_folder_url = plugins_url('assets/', __FILE__);
// Replace 'example.jpg' with the name of the image you want to display
$image_url = $image_folder_url . 'logo.png';
?>
<a href="<?php echo home_url(); ?>"><img src="<?php echo
esc_url($image_url) ?>" alt="Logo"></a>
<ul class="openai-dalle-nav">
<li class="openai-dalle-nav-item">
<a href="<?php echo admin_url('admin.php?page=openai-dalle-plugin'); ?>"
class="openai-dalle-nav-link"><?php _e('Chat', 'openai-dalle'); ?></a>
</li>
<li class="openai-dalle-nav-item">
<a href="https://facile.codes/ai-muse"
class="openai-dalle-nav-link">Documentation</a>
</li>
<li class="openai-dalle-nav-item">
<a href="<?php echo admin_url('admin.php?page=openai-dalle-settings'); ?>"
class="openai-dalle-nav-link"><?php _e('Settings', 'openai-dalle'); ?></a>
</li>
<!-- Add more navigation items here -->
</ul>
</div>
<main class="main" id="start">
<section class="nav">
<h1>AI Muse - Settings</h1>
<p>Configure and customize AI Muse, use <a href="">documentation</a> to
discover more.</p>
</section>
<section class="slider" id="tab-svelte">
<div class="wrap">
<!-- <h1>AI Muse Settings</h1> -->
<form method="post" action="options.php">
<?php
settings_fields('openai_dalle_plugin_settings_group');
do_settings_sections('openai_dalle_plugin_settings');
```

```
// Display settings fields with current value
$openai_api_key = get_option('openai_api_key');
$temperature = get_option('temperature');
$max_tokens = get_option('max_tokens');
$show_ai_muse = get_option('show_ai_muse');
$language = get_option('ai_muse_language');
?>
<div class="settings-container">
<div class="settings-card">
<h3>API Settings</h3>

<table class="form-table">
<tr valign="top">
<th scope="row">OpenAI API Key:</th>
<td><input type="text" name="openai_api_key" value="<?php echo
esc_attr($openai_api_key); ?>" /></td>
</tr>
<tr valign="top">
<th scope="row">Temperature:</th>
<td><input type="number" name="temperature" value="<?php echo
esc_attr($temperature); ?>" /></td>
</tr>
<tr valign="top">
<th scope="row">Max Tokens:</th>
<td><input type="number" name="max_tokens" value="<?php echo
esc_attr($max_tokens); ?>" /></td>
</tr>
</table>
</div>
<div class="settings-card">
<h3>Add-Ons</h3>
<table class="form-table">
<tr valign="top">
<th scope="row">Enable on Editor:</th>
<td>
<input type="checkbox" name="show_ai_muse" <?php if ($show_ai_muse) echo
'checked'; ?> />
</td>
</tr>
```

```
<tr valign="top">
<th scope="row">Enable on Elementor:</th>
<td>
<input type="checkbox" name="show_ai_muse_elementor" alt="SOON" disabled
/>
</td>
</tr>
<tr valign="top">
<th scope="row">Enable on WooCommerce:</th>
<td>
<input type="checkbox" name="show_ai_muse_elementor" alt="SOON" disabled
/>
</td>
</tr>
</table>
</div>
<div class="settings-card">
<h3>Conntent settings</h3>
<table class="form-table">
<tr valign="top">
<th scope="row">Language:</th>
<td>
<select name="language">
<option value="en" <?php if ($language === 'en') echo 'selected';
?>>English</option>
<option value="fr" <?php if ($language === 'fr') echo 'selected';
?>>French</option>
<option value="ar" <?php if ($language === 'ar') echo 'selected';
?>>Arabic</option>
<option value="de" <?php if ($language === 'de') echo 'selected';
?>>German</option>
<option value="es" <?php if ($language === 'es') echo 'selected';
?>>Spanish</option>
<option value="it" <?php if ($language === 'it') echo 'selected';
?>>Italian</option>
<option value="ja" <?php if ($language === 'ja') echo 'selected';
?>>Japanese</option>
<option value="ko" <?php if ($language === 'ko') echo 'selected';
?>>Korean</option>
```

```
<option value="zh" <?php if ($language === 'zh') echo 'selected';
?>>Chinese</option>
<!-- Add more language options as needed -->
</select>
</td>
</tr>
</table>
</div>
<div class="settings-card">
<p>Contact us for support and inquiries.</p>
</div>
</div>
<?php

submit_button();
?>
</form>
<?php
// Check if settings have been saved
if ( isset( $_GET['settings-updated'] ) && $_GET['settings-updated'] ==
'true' ) {
echo '<div id="message" class="updated notice
is-dismissible"><p><strong>Settings saved.</strong></p><button
type="button" class="notice-dismiss"><span
class="screen-reader-text">Dismiss this notice.</span></button></div>';
}
?>
</div>
</section>
</main>
</div>
<?php
}

function openai_dalle_plugin_settings() {
add_settings_section(
'openai_dalle_plugin_settings_section',
'',
null,
'openai_dalle_plugin_settings'
```

```
);

register_setting(
'openai_dalle_plugin_settings_group',
'openai_api_key'
);

register_setting(
'openai_dalle_plugin_settings_group',
'dalle_api_key'
);
}
add_action('admin_init', 'openai_dalle_plugin_settings');
function openai_api_key_callback() {
echo '<input type="text" name="api_key" value="' .
esc_attr(get_option('api_key')) . '" />';
}

function max_tokens_callback() {
echo '<input type="number" name="max_tokens" value="' .
esc_attr(get_option('max_tokens')) . '" />';
}

function temperature_callback() {
echo '<input type="number" step="0.01" name="temperature" value="' .
esc_attr(get_option('temperature')) . '" />';
}

function remove_footer_text_admin() {
wp_enqueue_style(
'custom-admin-css',
false
);
$custom_css = '
#wpfooter {
display: none;
}
';
wp_add_inline_style('custom-admin-css', $custom_css);
}
```

```
add_action('admin_enqueue_scripts', 'remove_footer_text_admin');


?>
```

Wdiget.js

```
jQuery(document).ready(function($) {
$('.open-modal').on('click', function() {
$(this).siblings('.modal').show();
});

$('.close').on('click', function() {
$(this).parent().parent('.modal').hide();
});
});
```