Creating a WordPress Plugin for EmaModal

This guide will walk you through the process of converting the `EmaModal` React component into a self-contained, distributable WordPress plugin. This approach allows any WordPress user to install, activate, and configure the modal through a simple admin interface.

Core Concepts of the Plugin

1. **Self-Contained**: All necessary code (PHP, JavaScript, CSS) will be included within the plugin's folder.
2. **Admin Settings Page**: The plugin will create an options page under the WordPress "Settings" menu where the user can enter their unique `emaUrl` and customize other modal properties.
3. **React-Powered Frontend**: We will use a modern JavaScript build process (Vite) to compile our React code into static assets that WordPress can load.
4. **Automatic Injection**: The plugin will automatically add the modal to the site's footer on all public-facing pages.

Step 1: Plugin File Structure

First, create a new folder for your plugin. The standard location is `wp-content/plugins/`. Let's name our folder `ema-modal-plugin`.

```
/ema-modal-plugin
|
|-- ema-modal-plugin.php      # The main plugin file (PHP)
|
|-- /react-src/                # Our React source code
|   |-- /src/
|   |   |-- components/
|   |   |   |-- EmaModal.tsx   # Your original component
|   |   |   `-- (other-react-components...)
|   |   |-- index.tsx       # The entry point for the React app
|   |   `-- index.css          # Styles for the React app
|   |-- package.json
|   |-- pnpm-lock.yaml
|   `-- vite.config.js
|
`-- /dist/              # The compiled, static JS/CSS assets (generated)
    |-- ema-modal.js
    `-- ema-modal.css
```

Step 2: The Main Plugin File (`ema-modal-plugin.php`)

This is the heart of your WordPress plugin. Create `ema-modal-plugin.php` and add the following code. This single file will handle:

- Plugin identification (the header comment).
- Creating the admin settings page.
- Saving the user's settings.
- Loading the compiled React scripts and styles on the frontend.
- Passing the saved settings from PHP to our JavaScript.

```php
<?php
/**
 * Plugin Name:       EMA Modal Plugin
 * Description:       Integrates the interactive EMA Modal into any WordPress si
 * Version:           1.0.0
 * Author:            Your Name
 * Author URI:        https://yourwebsite.com
 * License:           GPL v2 or later
 * License URI:       https://www.gnu.org/licenses/gpl-2.0.html
 * Text Domain:       ema-modal
 */

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

// 1. Create the Admin Menu item
function ema_modal_options_page_html() {
    if ( ! current_user_can( 'manage_options' ) ) {
        return;
    }
    ?>
    <div class="wrap">
        <h1><?php echo esc_html( get_admin_page_title() ); ?></h1>
        <form action="options.php" method="post">
            <?php
            settings_fields( 'ema_modal_options' );
            do_settings_sections( 'ema_modal' );
            submit_button( 'Save Settings' );
            ?>
        </form>
    </div>
    <?php
}

function ema_modal_options_page() {
    add_options_page(
        'EMA Modal Settings',
        'EMA Modal',
        'manage_options',
        'ema_modal',
        'ema_modal_options_page_html'
    );
}
add_action( 'admin_menu', 'ema_modal_options_page' );

// 2. Register Settings, Sections, and Fields
function ema_modal_register_settings() {
    register_setting( 'ema_modal_options', 'ema_modal_settings' );

    add_settings_section(
        'ema_modal_section_main',
        'Main Configuration',
        null,
        'ema_modal'
    );

    add_settings_field(
        'ema_modal_field_url',
        'EMA URL',
        'ema_modal_field_url_cb',
        'ema_modal',
        'ema_modal_section_main'
    );
     add_settings_field(
        'ema_modal_field_themecolor',
        'Theme Color',
        'ema_modal_field_color_cb',
        'ema_modal',
        'ema_modal_section_main'
    );
}
add_action( 'admin_init', 'ema_modal_register_settings' );

function ema_modal_field_url_cb() {
    $options = get_option( 'ema_modal_settings' );
    $url = isset( $options['ema_url'] ) ? esc_attr( $options['ema_url'] ) : '';
    echo '<input type="url" name="ema_modal_settings[ema_url]" value="' . $url .
}
```

```php
function ema_modal_field_color_cb() {
    $options = get_option( 'ema_modal_settings' );
    $color = isset( $options['theme_color'] ) ? esc_attr( $options['theme_color'
    echo '<input type="color" name="ema_modal_settings[theme_color]" value="' .
}


// 3. Enqueue Scripts and Styles for the Frontend
function ema_modal_enqueue_assets() {
    // Only enqueue on the public-facing site, not in the admin area
    if ( is_admin() ) {
        return;
    }

    $asset_path = plugin_dir_path( __FILE__ ) . 'dist/';
    $asset_uri = plugin_dir_url( __FILE__ ) . 'dist/';

    // Enqueue CSS
    wp_enqueue_style(
        'ema-modal-style',
        $asset_uri . 'ema-modal.css',
        [],
        filemtime( $asset_path . 'ema-modal.css' )
    );

    // Enqueue JS
    wp_enqueue_script(
        'ema-modal-script',
        $asset_uri . 'ema-modal.js',
        [],
        filemtime( $asset_path . 'ema-modal.js' ),
        true // Load in footer
    );

    // Get saved settings and pass them to our script
    $settings = get_option( 'ema_modal_settings' );
    $props = [
        'emaUrl'     => isset( $settings['ema_url'] ) ? $settings['ema_url'] : '
        'themeColor' => isset( $settings['theme_color'] ) ? $settings['theme_col
        // Add defaults for other props here
    ];
    wp_localize_script( 'ema-modal-script', 'emaModalProps', $props );
}
add_action( 'wp_enqueue_scripts', 'ema_modal_enqueue_assets' );

// 4. Add the root element for React to mount onto
```

```
function ema_modal_add_root_element() {
    // Again, only on the public-facing site
     if ( is_admin() ) {
        return;
    }
    echo '<div id="ema-modal-root"></div>';
}
add_action( 'wp_footer', 'ema_modal_add_root_element' );
```

Step 3: Set Up the React App

This is nearly identical to the process in the previous guide, but it's now encapsulated inside your plugin folder.

1. **Initialize the Project**:
   - Navigate into the plugin directory: `cd ema-modal-plugin`
   - Create and move into the React source directory: `mkdir react-src && cd react-src`
   - Initialize the project: `pnpm init`
   - Install dependencies: `pnpm add react react-dom` and `pnpm add -D vite @vitejs/plugin-react`
2. **Configure Vite (`vite.config.js`)**: Create this file inside `react-src`. It tells Vite where to put the compiled files.

```
// /react-src/vite.config.js
import react from '@vitejs/plugin-react';

import { defineConfig } from 'vite';

export default defineConfig({
    plugins: [react()],
    build: {
        // Output to the /dist folder in the plugin's root
        outDir: '../dist',
        assetsDir: '',
        rollupOptions: {
            output: {
                entryFileNames: 'ema-modal.js',
                assetFileNames: 'ema-modal.css'
            }
        }
    }
});
```

3. **Create the Entry Point (`index.tsx`)**: This file, located in `react-src/src/`, is what boots up your React application. It looks for the `#ema-modal-root` div and the `emaModalProps` object that our PHP script provides.

   ```
   // /react-src/src/index.tsx
   import React from 'react';

   import EmaModal from './components/EmaModal';
   import './index.css';
   import ReactDOM from 'react-dom/client';

   // Add your component styles here

   const rootElement = document.getElementById('ema-modal-root');

   if (rootElement) {
       // These props are passed from WordPress via wp_localize_script
       const props = (window as any).emaModalProps || {};

       // Only render if the essential emaUrl is provided
       if (props.emaUrl) {
           const root = ReactDOM.createRoot(rootElement);
           root.render(
               <React.StrictMode>
                   <EmaModal {...props} />
               </React.StrictMode>
           );
       }
   }
   ```

4. **Add Your Component**: Place your `EmaModal.tsx` and all its related UI components and hooks into `react-src/src/components/`. Remember to adapt any Next.js-specific code (like `<Image>`) to standard web APIs (`<img>`).

Step 4: Build and Distribute

1. **Build Your React App**:
   - From inside the `/react-src/` directory, run the build command: `pnpm build`.
   - Vite will compile everything and place the final `ema-modal.js` and `ema-modal.css` files into the `/dist/` folder at the root of your plugin.

2. **Prepare for Distribution**:
    - Your plugin is now fully functional. To share it, you just need to zip the entire `ema-modal-plugin` folder.
    - **IMPORTANT**: Before zipping, delete the `node_modules` folder from `react-src` to dramatically reduce the file size. The `dist` folder contains everything needed for the plugin to run.
    - The resulting `.zip` file can be uploaded directly to any WordPress site via the **Plugins** > **Add New** > **Upload Plugin** screen.

Once installed and activated, the user can navigate to **Settings** > **EMA Modal** in their WordPress dashboard, enter their URL, and the modal will instantly appear on their site.