**Step-by-Step Documentation for Creating the "Free-Range-Code-Challenge" Plugin**

**Step 1: Plugin Setup**

* **Organize the Structure:** Inside the project folder named Free-Range-Code-Challenge, create the main plugin file named Free-Range-Code-Challenge.php. This serves as the entry point for the plugin.
* **Plugin Header and Core Functions:** Add the plugin header for identification in WordPress and include all necessary files for functionality, such as the custom post type (CPT) and block registration files.
* **Custom Post Type and Block Registration:** Use register\_post\_type to define a custom post type for "Books" and acf\_register\_block\_type to register the Gutenberg block for the Book Listing feature.
* **Assets Management:** Ensure proper enqueuing of CSS and JavaScript files to manage styling and functionality for the block.

We needed a modular and maintainable structure, so the main plugin file only serves as an orchestrator, while the actual logic is compartmentalized into included files.

**Step 2: Register Custom Post Type**

* **File Structure for CPT:** Create a file named cpt-books.php under an includes folder to handle the registration of the "Books" post type.
* **Post Type and Taxonomy:** Register a post type to store book data and a custom taxonomy to categorize books, ensuring flexibility for filtering.

**Step 3: Create the Gutenberg Block**

* **Block Registration:** Include the code for block registration after the custom post type file is included in the main plugin file. Use Advanced Custom Fields (ACF) for defining block content fields.
* **Render Callback:** Define a render\_book\_listing\_block function to generate the block's frontend output dynamically using a template.

Gutenberg blocks provide a user-friendly way for the client to display and manage the "Book Listing" section across different pages. The ACF integration simplifies the creation and management of block-specific fields.

**Step 4: Add Advanced Custom Fields (ACF)**

* **Activate ACF Plugin**: Ensure the ACF plugin is active to manage custom fields.
* **Create Field Group:** Add a block-specific field group for the custom block. Include fields for title and description to allow content customization.
* **Set Display Rules:** Configure the field group to appear only when the "Book Listing" block is used.

ACF streamlines the process of adding and managing metadata. The custom fields make the block dynamic and reusable across pages with unique titles and descriptions for marketing purposes.

**Step 5: Create the Block Template**

* **Template File:** Create a templates folder in the plugin directory and add a block-book-listing.php file for the block's HTML structure.
* **Dynamic Content:** Use this template to loop through book posts and display their titles, descriptions, and categories.

Separating the template from logic ensures the code remains clean, modular, and easier to maintain. It also allows customization without affecting the plugin's core functionality.

**Step 6: Add Styling**

* **CSS Styling:** Create a CSS file under an assets/css folder and write styles to enhance the visual appearance of the block.
* **Enqueue Styles:** Link the CSS file to the plugin to ensure proper loading.

Good aesthetics improve user experience. Separating styles into an assets folder makes the plugin more maintainable and adheres to best practices.

**Step 7: Add Filtering & Search Functionality**

* **JavaScript for Interaction:** Create a JavaScript file in the assets/js folder to handle filtering and search logic dynamically.
* **Filtering Logic:** Use a dropdown for filtering books based on categories by matching a data attribute assigned to each book card.
* **Search Logic:** Implement an input field to search books by title or description by checking for matching text.

Dynamic interaction enhances usability. Using data attributes for filtering ensures a seamless client-side experience, while the search functionality provides quick navigation without reloading the page.