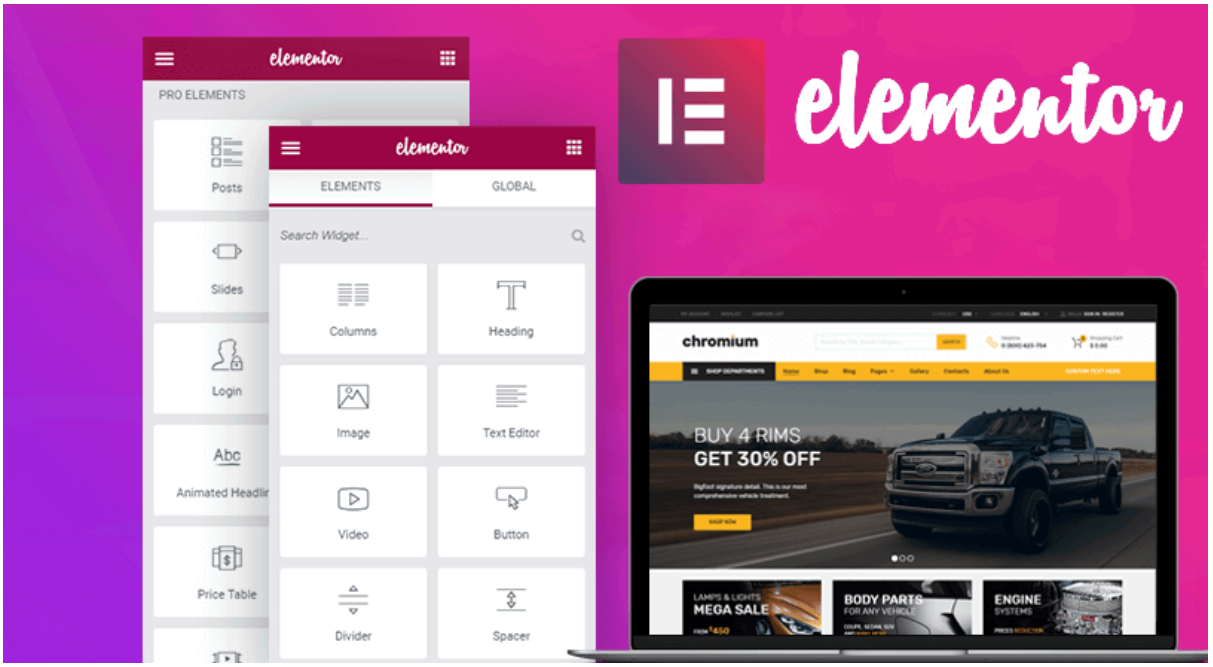
Columbia Documentation

---

1. What is Elementor?



Elementor is a **drag-and-drop page builder** for WordPress. It allows you to create beautiful websites without needing to know any coding. You can design pages by simply dragging widgets like text, images, and buttons into place. It's easy to use and very powerful!

How to Use Elementor:

1. **Install Elementor:**
    - Go to **Plugins > Add New** in your WordPress dashboard.
    - Search for "Elementor," click **Install**, and then **Activate**.
2. **Create a Page:**
    - Go to **Pages > Add New**.
    - Click **Edit with Elementor** to start designing your page.
3. **Using the Editor:**
    - On the left side, you'll see widgets (like text, images, and buttons).
    - Drag and drop widgets into the page on the right.
    - Customize each widget by clicking on it.
4. **Publish Your Page:**
    - When you're happy with the design, click **Publish** to make your page live.

Helpful Resources:

- **Elementor Blog**: How to use elementor in WordPress?
- **YouTube Channels**: Build a blog in elementor in WordPress.

---

2. custom-header.php

File Purpose:
This file sets up the custom header feature for the theme. It allows users to add and customize the header image and text from the WordPress Customizer.

Custom Header Setup:

- **Function:** `colombia_custom_header_setup()`
- **Purpose:**
  - Enables custom header support in the theme.
  - Sets default values like image, text color, and size.
  - Allows flexible header height.
- **Key Feature:** Uses `add_theme_support( 'custom-header', $args )` to enable the feature.
- **Hook:**
  - `add_action( 'after_setup_theme', 'colombia_custom_header_setup' )` – Runs the setup after the theme loads.

Header Styling:

- **Function:** `colombia_header_style()`
- **Purpose:**
  - Applies custom CSS to the header text and image.
  - Hides header text if the user chooses to do so in the Customizer.
- **Key Conditions:**
  - If the default text color is used, no additional CSS is applied.
  - If custom text color is set, the header text is styled with that color.
- **Output:**
  - Hides `.site-title` and `.site-description` if text is disabled.
  - Applies custom text color using inline CSS.

Functions and Features:

- **`add_theme_support( 'custom-header' )`**

    - **Purpose:** Adds support for custom headers in the theme.
    - **Parameters:**
        - `default-image` – Sets a default header image (empty by default).
        - `default-text-color` – Default header text color (black: `000000`).
        - `width` – Header image width (1000px).
        - `height` – Header image height (250px).
        - `flex-height` – Allows users to adjust header height.
- **`get_header_textcolor()`**
    - **Purpose:** Retrieves the current header text color.
- **`display_header_text()`**
    - **Purpose:** Checks if the header text should be displayed.
    - **Returns:** `true` if text should be visible, `false` if hidden.
- **`esc_attr()`**
    - **Purpose:** Escapes and outputs the header text color safely to prevent security issues.
- **`apply_filters()`**
    - **Purpose:** Allows other developers to modify the default custom header arguments through the `colombia_custom_header_args` filter.

How It Works:

1. When the theme is activated, the custom header feature is enabled.
2. Users can upload a header image and set text color through the WordPress Customizer.
3. The `colombia_header_style()` function applies CSS to the header based on user settings.

---

3. customizer.php

File Purpose

The file defines customization options for various sections of the website, such as the site title, tagline, testimonials, blog section, client logos, and newsletter. These settings are registered using the `WP_Customize_Manager` class, allowing users to customize their site's look and feel directly from the WordPress admin panel.

Customizer Setup:

- The `colombia_customize_register` function hooks into the WordPress Customizer to modify the site's title, description, and header color with live preview capabilities (`postMessage`).
- Selective refresh is enabled for the site title and description for smoother updates without a full page reload.

Theme Options Panel:

- A new Customizer panel (`colombia_panel`) is created to house all the theme-specific customization options. This panel is named "Bizwork Theme Options" and serves as the primary container for other sections.

Top Header Section:

- This section allows enabling or disabling the top header, top menu, social menu, and search icon.
- Each feature (e.g., top menu, social menu) is controlled by a checkbox in the `colombia_header_section`.

Footer Section:

- A separate section for customizing the footer, allowing users to enable/disable the social menu and modify the footer copyright text.
- There's also an option to enable the "Go to Top" feature, which adds a button for smooth scrolling to the top of the page.

Homepage Banner Section:

- This section manages the homepage banner, including options to enable/disable it, customize the heading, subheading, and upload images for banner sliders.
- Users can upload custom images for the banner sliders and provide text for the slider headings and subheadings.

Service Section:

- The service section on the homepage can be enabled/disabled via the Customizer.
- Users can customize the heading for the service section and upload a main image for the services.
- Each service box (like "Fast & Easy Service") can have its own heading text customized.

Sanitization & Transport:

- Each setting uses `sanitize_callback` to ensure data is safe and clean. For example, `sanitize_text_field` is used to sanitize text inputs.
- The transport method, typically set to `refresh`, controls how changes are previewed in the live Customizer. For quick updates, `postMessage` is used where applicable.

Image Uploads:

The `WP_Customize_Image_Control` class is used for allowing users to upload images for various sections like the banner and service section.

Service Box 1 Customizer Options

- **Description Setting**: Allows the user to edit the description text for the first service box. The text can be customized through the Customizer UI.
- **Image Setting**: Adds the ability to upload an image (icon) for the first service box.

Service Box 2 Customizer Options

- **Heading Setting**: Adds an option to customize the heading text for the second service box.
- **Description Setting**: Similar to Service Box 1, this setting allows editing the description text for Service Box 2.
- **Image Setting**: Allows the user to upload an image (icon) for the second service box.

Service Box 3 Customizer Options

- **Heading Setting**: Allows the user to customize the heading for the third service box.
- **Description Setting**: This option lets users edit the description text for the third service box.
- **Image Setting**: Provides an option to upload an image for the third service box.

Testimonial Section Customizer Options

- **Testimonial Section Enable/Disable**: Users can toggle the visibility of the testimonial section.
- **Testimonial Section Heading**: Allows users to set the main heading for the testimonial section.
- **Testimonial 1 Settings**:
  - **Name and Post**: Users can set the name and post (position) of the first testimonial.
  - **Image**: Option to upload an image for the first testimonial.
  - **Content**: Allows users to edit the testimonial text. HTML tags can be used here for linking or formatting.
- **Testimonial 2 Settings**:
  - **Name and Post**: Users can set the name and post of the second testimonial.
  - **Image**: Option to upload an image for the second testimonial.
  - **Content**: Allows users to edit the testimonial content.

Latest Blog Section Customizer Options

- **Enable/Disable Latest Blog Section**: Adds a checkbox to show or hide the blog section.
- **Featured Image Toggle**: Provides a setting to enable or disable the display of featured images in the blog section.

Logo Section Customizer Options

- **Enable/Disable Logo Section**: This option allows the user to toggle the visibility of the logo section.
- **Logo Upload Options**: Users can upload multiple client logos (Logo 1, Logo 2, etc.), which will be displayed in the section. Each logo can be customized with specific height and width attributes.

How This Code Works:

Each section (e.g., header, footer, banner, service boxes, testimonials, blog, and logo sections) is registered using `add_section`.

For each element, settings are defined using `add_setting`, which specifies details like the default value, capability (who can edit), and sanitization callbacks.

Controls are added via `add_control` or custom control classes (like `WP_Customize_Image_Control` for images) to define the user interface in the Customizer, allowing end users to update content.

`sanitize_callback` ensures the input data is safe and properly sanitized before being saved.

`esc_url_raw` is used to ensure URLs (like images) are stored securely, and `wp_kses_post` allows safe HTML content for text areas.

4. functions.php (Enhancements Section)

File Purpose:
This section of the `functions.php` file enhances the theme by adding extra features and modifying how WordPress handles certain elements.

Main Components:

1. Body Classes:

- **Function:** `colombia_body_classes( $classes )`
- **Purpose:**
  - Adds custom CSS classes to the `<body>` tag to help with styling different pages.
  - Dynamically applies classes based on page type and sidebar presence.
- **Key Features:**
  - Adds `hfeed` class to non-single pages (like archives).
  - Adds `no-sidebar` class if the primary sidebar (`sidebar-1`) is inactive.
  - Always adds `ct-sticky-sidebar` for sticky sidebar support.
- **Hook:**
  - `add_filter( 'body_class', 'colombia_body_classes' )` – Filters the list of body classes to include custom ones.

---

2. Pingback URL in Header:

- **Function:** `colombia_pingback_header()`
- **Purpose:**
  - Adds a pingback URL to the site's header, enabling auto-discovery for pingbacks.
  - This helps notify other sites when you link to them.
- **Condition:**
  - Only adds the pingback link if:
    - Viewing a single post, page, or attachment.
    - Pingbacks are enabled (`pings_open()`).
- **Output:**
  - Prints a `<link rel="pingback" href="URL">` tag in the `<head>`.
- **Hook:**
  - `add_action( 'wp_head', 'colombia_pingback_header' )` – Adds the pingback link to the head section of the HTML.

Functions and Features:

- **`body_class` Filter**
  - **Purpose:** Modifies the default CSS classes for the `<body>` element.
  - **Returns:** Array of classes including default and custom ones.
- **`is_singular()`**
  - **Purpose:** Checks if the current view is for a single post, page, or attachment.
  - **Returns:** `true` for single views, `false` for archives or homepage.

- `is_active_sidebar( 'sidebar-1' )`
    - **Purpose:** Checks if the sidebar with ID `sidebar-1` has widgets.
    - **Returns:** `true` if the sidebar is active, `false` if empty.
- `pings_open()`
    - **Purpose:** Checks if pingbacks are enabled for the current post/page.
    - **Returns:** `true` if pingbacks are allowed, `false` otherwise.
- `get_bloginfo( 'pingback_url' )`
    - **Purpose:** Retrieves the site's pingback URL.
- `esc_url()`
    - **Purpose:** Escapes and sanitizes URLs to ensure they are safe for output.

How It Works:

1. When a page loads, the body class filter runs, adding custom classes based on the page type and sidebar status.
2. For single posts or pages with pingbacks enabled, a pingback URL is added to the header automatically.

---

5. Custom-template-tags.php

File Purpose:
This file contains template tags that add important information to blog posts, such as the author, post date, categories, and tags. These functions enhance the theme by displaying metadata in a structured way.

Main Components:

1. Post Date/Time Display:

- **Function:** `colombia_posted_on()`
- **Purpose:**
    - Displays the post's published and updated dates.
    - Adds a clickable link to the post using the date.
- **Output:**
    - `<time>` HTML tags with the post date.
- **Conditional:**
    - If the post has been modified, both the original and updated dates are shown.

---

2. Post Author Display:

- **Function:** `colombia_posted_by()`
- **Purpose:**
  - Displays the name of the post's author.
  - Adds a link to the author's archive page.
- **Output:**
  - Shows the author’s name and links it to their posts.
  - Wrapped in a `byline` class for styling.

---

3. Post Footer Information (Categories, Tags, Comments):

- **Function:** `colombia_entry_footer()`
- **Purpose:**
  - Displays categories, tags, and comment links at the bottom of posts.
- **Key Features:**
  - **Categories:** Shows categories with a "Posted in" label.
  - **Tags:** Lists tags under "Tagged" text.
  - **Comments:** Adds a comment link if comments are open.
  - **Edit Link:** Adds an "Edit" button for logged-in users.

---

4. Post Thumbnail Display:

- **Function:** `colombia_post_thumbnail()`
- **Purpose:**
  - Displays the post’s featured image (thumbnail).
- **Conditional:**
  - If the post is protected by a password or doesn’t have a thumbnail, it skips displaying the image.
  - On single posts, the image appears as a `div`.
  - On archive pages, the thumbnail links to the full post.
- **Output:**
  - `<div>` or `<a>` with the post thumbnail inside.

---

5. `wp_body_open()` Fallback (For Older WordPress Versions):

- **Function:** `wp_body_open()`
- **Purpose:**
  - Provides backward compatibility for themes using WordPress versions before 5.2.

    - Ensures that the `wp_body_open` action runs properly.
- **Output:**
    - Triggers the `wp_body_open` action.

Functions and Features:

- `get_the_date()` / `get_the_modified_date()`
    - **Purpose:** Retrieves the published or modified date of a post.
    - **Returns:** Date in different formats.
- `get_the_author()`
    - **Purpose:** Returns the author's display name.
- `get_author_posts_url()`
    - **Purpose:** Retrieves the URL of the author's archive page.
- `get_the_category_list()`
    - **Purpose:** Retrieves a list of categories for the current post.
- `get_the_tag_list()`
    - **Purpose:** Returns tags associated with the post.
- `comments_popup_link()`
    - **Purpose:** Displays a link to leave comments or view existing ones.
- `the_post_thumbnail()`
    - **Purpose:** Displays the featured image of the post.
- `is_singular()`
    - **Purpose:** Checks if the current page is a single post, page, or attachment.

How It Works:

1. When posts are displayed, these functions add metadata like the date, author, and categories.
2. The thumbnail is displayed dynamically based on whether the post is single or part of an archive.
3. Footer sections with comments and edit links are generated automatically for each post.

6. wptt-webfont-loader.php

The file `wptt-webfont-loader.php` manages the loading and integration of web fonts into a WordPress site. It could handle enqueueing web font stylesheets, optimizing font loading, or conditionally loading fonts based on certain criteria.

7. content-none.php

File Purpose: Displays messages and provides options to users when there are no posts or content available based on the current query.

Code Breakdown:

- **Section Wrapper**:
  - The section is given the classes `no-results` and `not-found`, indicating that no content was found.
- **Header**:
  - Displays a page title indicating that nothing was found.
- **Conditional Content**:
  - **For Home Page with Publishing Permissions**:
    - A message inviting users to publish their first post, with a link to the WordPress admin page for creating a new post.
  - **For Search Results**:
    - A message informing users that no matches were found for their search terms and offers the search form to try different keywords.
  - **For General No Results**:
    - A message suggesting that users try searching again and includes the search form.
- **Search Form**:
  - Included to allow users to start a new search.

Functions and Features:

- `is_home()`:
  - **Purpose**: Checks if the current page is the blog posts index (i.e., the home page of the blog).
  - **Returns**: `true` if it is the home page, `false` otherwise.
- `current_user_can( 'publish_posts' )`:
  - **Purpose**: Determines if the current user has the capability to publish posts.
  - **Parameters**:
    - `'publish_posts'` – A capability check to see if the user can publish posts.
  - **Returns**: `true` if the user has the capability, `false` otherwise.
- `esc_html_e()`:
  - **Purpose**: Translates and safely outputs HTML-escaped text.
  - **Parameters**:
    - String to translate and display.
    - Optional: Text domain for localization.
- `wp_kses()`:
  - **Purpose**: Allows specific HTML tags while stripping out all other tags.
  - **Parameters**:
    - String containing HTML.
    - Array of allowed HTML tags and attributes.

- `esc_url()`:
  - **Purpose**: Escapes a URL for safe use in HTML attributes.
  - **Parameters**:
    - URL to be escaped.
- `admin_url( 'post-new.php' )`:
  - **Purpose**: Generates the URL to the WordPress admin page for creating a new post.
  - **Parameters**:
    - `'post-new.php'` – The specific admin page.
- `get_search_form()`:
  - **Purpose**: Outputs the search form.
  - **Returns**: HTML code for the search form.

8. content-page.php

File Purpose: Renders the content of a page, including the page title, featured image, content, and an edit link if applicable.

- **Featured Image**:
  - Calls a custom function to display the featured image of the page.
- **Content**:
  - Displays the main content of the page.
  - Includes pagination if the content is split across multiple pages.
- **Footer**:
  - If an edit link is available, it shows a link to edit the page with the page title.

Functions and Features:

- `the_ID()`:
  - **Purpose**: Outputs the ID of the current post or page.
  - **Returns**: The post or page ID.
- `post_class()`:
  - **Purpose**: Outputs a list of CSS classes for the post or page.
  - **Parameters**:
    - None (classes are generated based on the post type and status).
- `the_title( '<h1 class="entry-title">', '</h1>' )`:
  - **Purpose**: Displays the title of the post or page within `<h1>` tags.
  - **Parameters**:
    - Opening HTML tag.
    - Closing HTML tag.
- `colombia_post_thumbnail()`:
  - **Purpose**: Calls a custom function to display the post thumbnail (featured image).
  - **Returns**: HTML markup for the post thumbnail.
- `the_content()`:
  - **Purpose**: Outputs the content of the post or page.
  - **Returns**: The content of the post or page.

- `wp_link_pages()`:
  - **Purpose**: Outputs pagination links for split content.
  - **Parameters**:
    - `before`: HTML to display before the pagination links.
    - `after`: HTML to display after the pagination links.
- `get_edit_post_link()`:
  - **Purpose**: Checks if an edit link is available for the current post or page.
  - **Returns**: Edit link URL if available, otherwise `false`.
- `edit_post_link()`:
  - **Purpose**: Displays a link to edit the post or page.
  - **Parameters**:
    - Link text (formatted with screen-reader text for accessibility).
    - Opening HTML tag for the link.
    - Closing HTML tag for the link.
- `wp_kses()`:
  - **Purpose**: Allows specific HTML tags while stripping out all other tags.
  - **Parameters**:
    - String containing HTML.
    - Array of allowed HTML tags and attributes.
- `wp_kses_post()`:
  - **Purpose**: Allows specific HTML tags suitable for post content.
  - **Parameters**:
    - The content to be filtered.
- `esc_html__()`:
  - **Purpose**: Translates and escapes HTML text.
  - **Parameters**:
    - String to translate and escape.
    - Optional: Text domain for localization.

9. content-search.php

File Purpose: This template part is used for displaying search results. It structures the layout for each search result item, including the featured image, title, excerpt, and categories.

- **Article Wrapper**:
  - The article is identified with the post ID and specific classes for styling. It wraps the content of each search result.
- **Card Item**:
  - Contains the main structure of the search result item, including media and body sections.
- **Card Media**:
  - Displays the post thumbnail if available.
- **Card Body**:
  - Contains the header, content, and footer of the search result item.
- **Entry Header**:

    - **Meta Information**: Displays meta information for posts, including date and author, depending on whether it is a singular view or not.
    - **Title**: Displays the title with appropriate HTML tags and link if not in a singular view.
- **Entry Content**:
    - Displays the content or excerpt of the post.
    - **Singular View**: Shows full content.
    - **Non-Singular View**: Shows an excerpt and a "Read More" button.
- **Entry Footer**:
    - Displays additional meta information for the post, if in a singular view.

Functions and Features:

- `has_post_thumbnail()`:
    - **Purpose**: Checks if the current post or page has a featured image.
    - **Returns**: `true` if a featured image exists, otherwise `false`.
- `colombia_post_thumbnail()`:
    - **Purpose**: Calls a custom function to display the post thumbnail (featured image).
    - **Returns**: HTML markup for the post thumbnail.
- `the_title( '<h1 class="entry-title">', '</h1>' )`:
    - **Purpose**: Displays the title of the post or page within `<h1>` tags for singular views.
    - **Parameters**:
        - Opening HTML tag.
        - Closing HTML tag.
- `the_title( '<h3 class="entry-title"><a href="' . esc_url( get_permalink() ) . '" rel="bookmark">', '</a></h3>' )`:
    - **Purpose**: Displays the title of the post or page with a link to the full post for non-singular views.
    - **Parameters**:
        - Opening HTML tag with URL to the post.
        - Closing HTML tag.
- `colombia_posted_on()`:
    - **Purpose**: Calls a custom function to display meta information for the post date.
    - **Returns**: HTML markup for the post date.
- `colombia_posted_by()`:
    - **Purpose**: Calls a custom function to display meta information for the post author.
    - **Returns**: HTML markup for the post author.
- `get_the_category_list()`:
    - **Purpose**: Retrieves a list of categories for the post.
    - **Parameters**:
        - Separator between categories.
- `the_content()`:

    - **Purpose**: Displays the full content of the post.
    - **Returns**: The content of the post.
- `wp_trim_words()`:
    - **Purpose**: Trims the post excerpt to a specified number of words.
    - **Parameters**:
        - The text to trim.
        - The number of words to limit to.
- `get_theme_mod()`:
    - **Purpose**: Retrieves a theme modification value.
    - **Parameters**:
        - The modification key.
        - Optional: Default value if the modification is not set.
- `wp_link_pages()`:
    - **Purpose**: Outputs pagination links for split content.
    - **Parameters**:
        - `before`: HTML to display before the pagination links.
        - `after`: HTML to display after the pagination links.
- `get_edit_post_link()`:
    - **Purpose**: Checks if an edit link is available for the current post.
    - **Returns**: Edit link URL if available, otherwise `false`.
- `edit_post_link()`:
    - **Purpose**: Displays a link to edit the post.
    - **Parameters**:
        - Link text (formatted with screen-reader text for accessibility).
        - Opening HTML tag for the link.
        - Closing HTML tag for the link.

10. content.php

File Purpose: Displays individual posts with optional featured images, post titles, metadata (such as post date and author), content, and category information.

Code Breakdown:

- **Article Wrapper**:
    - `<article id="post-<?php the_ID(); ?>" <?php post_class(); ?>>`: Container for an individual post. Uses the post ID and class names.
- **Card Item**:
    - `<div class="card-item card-blog-post">`: Wraps the post content in a styled card element.
- **Post Thumbnail**:
    - `if (has_post_thumbnail())`: Checks if the post has a featured image.
        - `<div class="card-media">`: Container for the post's featured image.

        - `<?php colombia_post_thumbnail(); ?>`: Calls a custom function to display the post's featured image.
- **Card Body**:
    - `<div class="card-body">`: Contains the main content of the post.
- **Entry Header**:
    - `<header class="entry-header">`: Contains the post's title and metadata.
    - `if ( 'post' === get_post_type() )`: Checks if the current post type is 'post'.
        - **Singular vs. Non-Singular**:
            - `if ( !is_singular())`: Displays post metadata if not viewing a single post.
                - `<div class="entry-meta">`: Container for post metadata (date and author).
                - `<?php colombia_posted_on(); ?>`: Calls a custom function to display the post's date.
                - `<?php colombia_posted_by(); ?>`: Calls a custom function to display the post's author.
            - `if ( is_singular() )`: Displays post title and metadata when viewing a single post.
                - `the_title()`: Displays the post title with appropriate HTML tags based on the view (single or not).
                - `$categories_list`: Retrieves and displays the list of categories for the post.
                - `if ( $categories_list )`: Displays categories if available.
        - `if ( !is_singular())`: Displays the categories list for posts on archive or index views.
- **Entry Content**:
    - `<div class="entry-content">`: Contains the post content.
    - `if ( is_singular() )`: Displays the full content of the post if viewing a single post.
        - `the_content()`: Displays the full content of the post.
    - `else`: Displays a trimmed excerpt and a "Read More" link if not viewing a single post.
        - `wp_trim_words()`: Trims the excerpt to a specified number of words.
        - `<a href="<?php the_permalink(); ?>" class="btn read-btn text-uppercase">`: Provides a link to the full post with customizable "Read More" text.
- **Pagination**:
    - `wp_link_pages()`: Displays pagination links for multi-page posts.

    - `array( 'before' => '<div class="page-links">' . esc_html__( 'Pages:', 'colombia' ), 'after' => '</div>' )`: Customizes the pagination output.
- **Closing Tags**:
    - `</div><!-- .card-body -->`: Closes the card body container.
    - `</div><!-- .card-item -->`: Closes the card item container.
    - `</article><!-- #post-<?php the_ID(); ?> -->`: Closes the article container.

Functions and Features:

- `has_post_thumbnail()`:
    - **Purpose**: Checks if the post has a featured image.
    - **Returns**: Boolean value (true or false).
- `colombia_post_thumbnail()`:
    - **Purpose**: Displays the post's featured image.
    - **Returns**: HTML markup for the featured image.
- `the_title()`:
    - **Purpose**: Displays the post title.
    - **Parameters**:
        - Before and after HTML tags.
- `colombia_posted_on()`:
    - **Purpose**: Displays the post's publication date.
    - **Returns**: HTML markup for the post date.
- `colombia_posted_by()`:
    - **Purpose**: Displays the post's author.
    - **Returns**: HTML markup for the post author.
- `get_the_category_list()`:
    - **Purpose**: Retrieves a list of categories associated with the post.
    - **Parameters**:
        - Separator between categories.
- `the_content()`:
    - **Purpose**: Displays the full content of the post.
    - **Returns**: HTML markup for the post content.
- `wp_trim_words()`:
    - **Purpose**: Trims a string to a specified number of words.
    - **Parameters**:
        - Text to trim.
        - Number of words.
- `get_theme_mod()`:
    - **Purpose**: Retrieves a value from the theme customizer settings.
    - **Parameters**:
        - Setting name.
        - Default value if not set.
- `wp_link_pages()`:
    - **Purpose**: Displays pagination links for multi-page posts.

    - **Parameters**:
        - Arguments for customizing pagination output.

11. 404.php

File Purpose: Displays a 404 error page when a requested page or resource cannot be found. Provides users with options to search for content, view recent posts, browse popular categories, and check archives.

Code Breakdown:

- **Header**:
    - `get_header()`: Includes the header template for the theme.
- **Container**:
    - `<div class="container">`: Wraps the main content of the page for styling purposes.
- **Site Wrapper**:
    - `<div class="site-wrapper">`: Additional wrapper for the main content area.
- **Main Content Area**:
    - `<main id="primary" class="site-main">`: Contains the main content of the page.
- **Error Section**:
    - `<section class="error-404 not-found">`: Section specifically for 404 error content.
    - **Header**:
        - `<header class="page-header">`: Contains the title of the 404 error page.
        - `esc_html_e()`: Escapes and displays a translatable string for the error message.
    - **Page Content**:
        - `<div class="page-content">`: Contains the main content for the 404 error page.
        - `<p>`: Provides a message indicating that the page was not found and suggests trying a search or other links.
        - `get_search_form()`: Includes the search form widget for users to search the site.
        - `the_widget( 'WP_Widget_Recent_Posts' )`: Displays a widget showing recent posts.
- **Categories Widget**:
    - `<div class="widget widget_categories">`: Container for the categories widget.
    - `<h2 class="widget-title">`: Widget title for "Most Used Categories".
    - `wp_list_categories()`: Lists categories ordered by count in descending order, displaying up to 10 categories.

- **Archives Widget**:
  - `$colombia_archive_content`: Defines content for the archives widget, including a smiley face.
  - `the_widget( 'WP_Widget_Archives', 'dropdown=1', "after_title=</h2>$colombia_archive_content" )`: Displays a dropdown archives widget with custom content after the title.
- **Tag Cloud Widget**:
  - `the_widget( 'WP_Widget_Tag_Cloud' )`: Displays a tag cloud widget.
- **Footer**:
  - `get_footer()`: Includes the footer template for the theme.

Functions and Features:

- `get_header()`:
  - **Purpose**: Includes the header template file.
  - **Returns**: No direct output; includes header markup.
- `get_search_form()`:
  - **Purpose**: Displays the search form.
  - **Returns**: HTML markup for the search form.
- `the_widget()`:
  - **Purpose**: Displays a widget.
  - **Parameters**:
    - Widget class.
    - Widget arguments.
    - Widget settings.
- `wp_list_categories()`:
  - **Purpose**: Lists categories with specified arguments.
  - **Parameters**:
    - `orderby`: Criteria to order categories (e.g., 'count').
    - `order`: Order direction (e.g., 'DESC').
    - `show_count`: Whether to display the number of posts in each category.
    - `title_li`: HTML before the list of categories.
    - `number`: Number of categories to display.
- `convert_smilies()`:
  - **Purpose**: Converts smiley text to graphical emoticons.
  - **Parameters**:
    - Text containing smileys.
- `esc_html_e()`:
  - **Purpose**: Escapes and displays a translatable string.
  - **Parameters**:
    - String to be displayed.
    - Text domain for translation.
- `the_widget()`:
  - **Purpose**: Displays a specified widget.

    - **Parameters**:
        - Widget class.
        - Widget arguments.
        - Widget settings.
- `get_footer()`:
    - **Purpose**: Includes the footer template file.
    - **Returns**: No direct output; includes footer markup.

12. archive.php

File Purpose: Displays a list of posts on archive pages, including category, tag, author, and date archives. It features the archive title, description, and post listing.

Code Breakdown:

- **Header**:
    - `get_header();`: Includes the header template file (`header.php`).
- **Container and Main Wrapper**:
    - `<div class="container">`: Main container for the page content.
    - `<div class="main-wrapper">`: Wrapper for the main content and sidebar.
    - `<main id="primary" class="site-main ct-post-wrapper">`: Main content area where posts will be displayed.
- **Check for Posts**:
    - `if ( have_posts() ) :`: Checks if there are posts to display.
- **Archive Header**:
    - `<header class="page-header">`: Container for the archive title and description.
    - `the_archive_title( '<h1 class="page-title">', '</h1>' );`: Displays the title of the archive page (e.g., category or tag name).
    - `the_archive_description( '<div class="archive-description">', '</div>' );`: Displays the description of the archive page, if available.
- **The Loop**:
    - `while ( have_posts() ) :`: Starts the loop to go through each post.
    - `the_post();`: Sets up the post data for the current post in the loop.
    - `get_template_part( 'template-parts/content', get_post_type() );`: Includes a template part for displaying the post content based on the post type (e.g., `content-post.php` for standard posts). It allows for post type-specific customization.
    - `endwhile;`: Ends the loop.
- **Post Navigation**:

    - `the_posts_navigation();`: Displays navigation links to previous and next sets of posts, if available.
- **No Posts Found**:
    - `else :`: Runs if no posts are found.
    - `get_template_part( 'template-parts/content', 'none' );`: Includes a template part for displaying a message when no posts are found (e.g., `content-none.php`).
- **Sidebar**:
    - `get_sidebar();`: Includes the sidebar template file (`sidebar.php`).
- **Footer**:
    - `get_footer();`: Includes the footer template file (`footer.php`).

Functions and Features:

- `get_header()`:
    - **Purpose**: Includes the header section of the site.
    - **Parameters**: None.
- `have_posts()`:
    - **Purpose**: Checks if there are any posts to display.
    - **Returns**: Boolean value (true if there are posts).
- `the_archive_title()`:
    - **Purpose**: Displays the title of the current archive page.
    - **Parameters**:
        - HTML before and after the title.
- `the_archive_description()`:
    - **Purpose**: Displays the description of the current archive page.
    - **Parameters**:
        - HTML before and after the description.
- `the_post()`:
    - **Purpose**: Sets up the post data for the current post in the loop.
    - **Parameters**: None.
- `get_template_part()`:
    - **Purpose**: Includes a template part based on the post type or name.
    - **Parameters**:
        - File name to include (`template-parts/content`).
        - Optional post type or name (`get_post_type()`).
- `the_posts_navigation()`:
    - **Purpose**: Displays navigation links for paginated posts.
    - **Parameters**: None.
- `get_template_part()` **(No Posts)**:
    - **Purpose**: Includes a template part for displaying a message when no posts are found.
    - **Parameters**:
        - File name to include (`template-parts/content`).
        - Optional name (`'none'`).

- `get_sidebar()`:
  - **Purpose**: Includes the sidebar section of the site.
  - **Parameters**: None.
- `get_footer()`:
  - **Purpose**: Includes the footer section of the site.
  - **Parameters**: None.

13. comments.php

File Purpose: Displays the comments area on a page or post, including the comment list, comment form, and navigation for comments.

Code Breakdown:

- **Password Protected Posts**:
  - `if ( post_password_required() ) { return; }`:
    - **Purpose**: Checks if the post is protected by a password and if the visitor has not entered the password. If true, it prevents loading the comments.
- **Comments Area**:
  - `<div id="comments" class="comments-area">`:
    - **Purpose**: Container for the comments section.
- **Comments Title**:
  - `if ( have_comments() ) :`:
    - **Purpose**: Checks if there are comments to display.
  - `<h2 class="comments-title">`:
    - **Purpose**: Displays the title of the comments section.
  - `$colombia_comment_count = get_comments_number();`:
    - **Purpose**: Gets the number of comments on the post.
  - `if ( '1' === $colombia_comment_count )`:
    - **Purpose**: Checks if there is only one comment.
    - `printf( esc_html__( 'One thought on &ldquo;%1$s&rdquo;', 'colombia' ), '<span>' . wp_kses_post( get_the_title() ) . '</span>' );`:
      - **Purpose**: Displays a message for a single comment.
  - `else`:
    - **Purpose**: If there is more than one comment.
    - `printf( esc_html( _nx( '%1$s thought on &ldquo;%2$s&rdquo;', '%1$s thoughts on &ldquo;%2$s&rdquo;', $colombia_comment_count, 'comments title', 'colombia' ) ), number_format_i18n( $colombia_comment_count ),`

`'<span>' . wp_kses_post( get_the_title() ) . '</span>' );`:

- **Purpose**: Displays a message for multiple comments.

- **Comment Navigation**:
  - `the_comments_navigation();`:
    - **Purpose**: Displays navigation links for comments (previous and next).
- **Comment List**:
  - `<ol class="comment-list">`:
    - **Purpose**: Ordered list container for the comments.
  - `wp_list_comments( array( 'style' => 'ol', 'short_ping' => true ) );`:
    - **Purpose**: Displays a list of comments. The `style` argument specifies the list type, and `short_ping` enables short pings for quick replies.
- **Comment Navigation (After List)**:
  - `the_comments_navigation();`:
    - **Purpose**: Displays navigation links for comments after the list.
- **Comments Closed**:
  - `if ( ! comments_open() ) :`:
    - **Purpose**: Checks if comments are closed for the post.
  - `<p class="no-comments"><?php esc_html_e( 'Comments are closed.', 'colombia' ); ?></p>`:
    - **Purpose**: Displays a message indicating that comments are closed if comments are not open.
- **Comment Form**:
  - `comment_form();`:
    - **Purpose**: Displays the comment form for users to submit new comments.

Functions and Features:

- `post_password_required()`:
  - **Purpose**: Checks if the current post is protected by a password.
  - **Returns**: Boolean value (true if password protected).
- `have_comments()`:
  - **Purpose**: Checks if there are comments to display.
  - **Returns**: Boolean value (true if comments exist).
- `get_comments_number()`:
  - **Purpose**: Retrieves the number of comments for the current post.
  - **Returns**: Integer (number of comments).
- `the_comments_navigation()`:
  - **Purpose**: Displays navigation links for comments.
  - **Parameters**: None.
- `wp_list_comments()`:
  - **Purpose**: Displays a list of comments.
  - **Parameters**:

        - `style`: Defines the style of the list (e.g., `ol` for ordered list).
        - `short_ping`: Enables short pings for quick replies.
- `comments_open()`:
    - **Purpose**: Checks if comments are open for the current post.
    - **Returns**: Boolean value (true if comments are open).
- `comment_form()`:
    - **Purpose**: Displays the comment form.
    - **Parameters**: None.

---

14. elementor-template.php

File Purpose:
This file is a custom template for displaying pages created with Elementor. It defines a clean layout without sidebars or extra elements, focusing entirely on the content built in Elementor.

Main Components:

1. Template Name Declaration:

- **Code:** `Template Name: Elementor Template`
- **Purpose:**
    - This line tells WordPress that this is a custom template called "Elementor Template."
    - Users can select this template when creating or editing a page in the WordPress admin.

---

2. Header Inclusion:

- **Function:** `get_header()`
- **Purpose:**
    - Loads the site's header (usually `header.php`).
    - Ensures that the page has the same header as the rest of the site.

---

3. Main Content Section:

- **HTML Tag:** `<main class="elementor-content">`
- **Function:** `the_content()`
- **Purpose:**
    - Displays the main content of the page.
    - Pulls the content created in the WordPress editor or Elementor.

- **Class:**
    - `elementor-content` – Can be styled with CSS to match the Elementor design.

---

4. Footer Inclusion:

- **Function:** `get_footer()`
- **Purpose:**
    - Loads the site's footer (usually `footer.php`).
    - Ensures consistency by using the same footer across pages.

Functions and Features:

- **`get_header()`**
    - **Purpose:** Loads the header template part.
    - **Returns:** HTML for the header section.
- **`the_content()`**
    - **Purpose:** Displays the content of the post or page.
    - **Returns:** The main body text or Elementor layout.
- **`get_footer()`**
    - **Purpose:** Loads the footer template part.
    - **Returns:** HTML for the footer section.

How It Works:

1. When a page using this template is loaded, WordPress calls the header and footer.
2. The content created in Elementor or the WordPress editor is displayed in the middle section (`main`).
3. This simple layout is ideal for full-width Elementor designs, ensuring that no unnecessary elements interfere with the page layout.

---

15. footer.php

File Purpose:
This template displays the footer section of the website. It closes the main content area and adds footer widgets, social links, copyright information, and a "Go to Top" button.

Main Components:

1. Footer Wrapper:

- **HTML Tag:** `<footer id="colophon" class="site-footer">`
- **Purpose:**
    - This wraps the entire footer section and uses the ID `colophon` for styling and accessibility.
    - The class `site-footer` is added for CSS styling.

---

2. Footer Widgets Section:

- **Condition:**
    - `if (is_active_sidebar('footer-1') || is_active_sidebar('footer-2') || is_active_sidebar('footer-3'))`
    - **Purpose:**
        - Checks if any of the three footer widget areas (`footer-1`, `footer-2`, `footer-3`) are active.
        - If at least one widget is active, the section is displayed.
- **Class:**
    - `footer-top` – Contains widget areas styled in columns using `flex-row` for responsiveness.

Widgets Displayed:

- `dynamic_sidebar('footer-1')` – Displays widgets added to the first footer area.
- `dynamic_sidebar('footer-2')` – Displays widgets for the second area.
- `dynamic_sidebar('footer-3')` – Displays widgets for the third area.

---

3. Footer Bottom Section:

- **Class:** `footer-bottom`
- **Purpose:**
    - Contains site information, including copyright, social links, and theme attribution.

---

4. Copyright Text:

- **Function:**
  - `get_theme_mod('colombia_copyright_option', 'Copyright All Rights Reserved © 2024')`
  - **Purpose:**
    - Retrieves custom copyright text from the theme customizer.
    - Defaults to "Copyright All Rights Reserved © 2024" if no custom text is set.

---

5. Social Links Section:

- **Condition:**
  - `if($footer_social == 1)`
  - **Purpose:**
    - Checks if the social menu is enabled through the customizer (`colombia_footer_social_menu`).
    - Calls the `colombia_social_menu()` function to display social icons.

---

6. Theme Credit:

- **Function:**
  - `printf( esc_html__( 'Theme: %1$s by %2$s.', 'colombia' ), 'Colombia', '<a href="https://cemreworks.com/">Cemre Works</a>' )`
  - **Purpose:**
    - Displays the theme name and author with a link to the author's website.

---

7. Go to Top Button:

- **Condition:**
  - `if($footer_go_to_top == 1)`
  - **Purpose:**
    - Adds a "Go to Top" button if enabled in the theme customizer (`colombia_enable_go_to_top_option`).
    - Clicking the button scrolls the user back to the top of the page.

Functions and Features:

- `is_active_sidebar()`
    - **Purpose:** Checks if a widget area has active widgets.
    - **Returns:** True if widgets are present, false if empty.
- `dynamic_sidebar()`
    - **Purpose:** Displays the widgets from a specified sidebar area.
- `get_theme_mod()`
    - **Purpose:** Retrieves custom values from the theme customizer.
- `wp_footer()`
    - **Purpose:** Ensures all scripts and footer hooks are loaded before closing the page.

How It Works:

1. The footer displays widgets if they are active.
2. Copyright text and social links are pulled from the customizer.
3. A "Go to Top" button is conditionally displayed.
4. Finally, `wp_footer()` ensures the footer is properly loaded and the HTML closes with the `</html>` tag.

16. front-page.php

File Purpose:

This is the **Front Page Template** for the **Colombia** WordPress theme. It displays dynamic content sections like banners, about us, testimonials, blogs, client logos, and newsletters, all customizable through the WordPress Customizer.

Key Sections and Features:

1. **Home Banner**
    - Displays a slider with two banners.
    - Customizable images, headings, and subheadings via Customizer.
2. **About Us Section**
    - Includes title, main image, description, and a call-to-action button.

    - Displayed if `colombia_home_about_section_enable` is enabled in the Customizer.
3. **Testimonials**
    - Displays 3 customer testimonials with name, position, content, and images.
    - Customizable through Customizer.
4. **Blog Section**
    - Displays 6 recent blog posts with excerpts, categories, and optional featured images.
    - Fetched via a custom `WP_Query` based on settings in the Customizer.
5. **Client Logos**
    - Shows client logos (5 images) if enabled.
    - Customizable via Customizer.
6. **Newsletter Section**
    - Includes a newsletter subscription prompt with a customizable button text and link.

Key Logic:

- **Conditional Display**: Each section is shown based on a corresponding `get_theme_mod` setting in the Customizer (e.g., `colombia_home_banner_section_enable`).
- **Dynamic Content**: Text, images, and links are dynamically fetched from the Customizer for easy customization by the user.
- **WP_Query for Blogs**: Displays the latest 6 blog posts with options for featured images and categories.

Customization:

- All sections can be customized via the WordPress Customizer (e.g., images, text, links).
- Sections are enabled or disabled through Customizer options.

This template makes it easy for users to customize the front page content without coding, while ensuring flexibility and dynamic content display.

17. functions.php

File Purpose:

This file defines essential functions and features for the **Colombia** WordPress theme, including theme setup, widget registration, custom logo, custom header, navigation menus, script and style enqueuing, and various custom functions for handling posts, widgets, and the Customizer.

Key Features & Functions:

1. **Theme Setup (`colombia_setup`)**
    - Translations, RSS feed links, title tag management, post thumbnail support.
    - Registers three navigation menus: Primary, Top Menu, and Social Menu.
    - Adds support for HTML5 elements and custom background.
    - Adds support for custom logo.
2. **Content Width (`colombia_content_width`)**
    - Sets the content width globally for proper design layout.
3. **Widget Areas (`colombia_widgets_init`)**
    - Registers widget areas: Sidebar, Footer 1, Footer 2, Footer 3.
4. **Social Menu (`colombia_social_menu`)**
    - Displays a social media menu if one is defined in the WordPress menu system.
5. **Script and Style Enqueueing (`colombia_scripts`)**
    - Loads fonts, Font Awesome, Slick Carousel, and custom JS scripts.
    - Adds support for RTL styles and enqueues comment-reply script for threaded comments.
6. **Customizer Support**
    - Includes additional functionality to support the WordPress Customizer.
7. **Custom Header & Template Tags**
    - Implements custom header functionality and includes template files for various custom functions.
8. **Jetpack Compatibility**
    - Checks if Jetpack is active and includes relevant compatibility files.
9. **Sanitization Functions:**
    - `colombia_sanitize_checkbox`: Sanitizes checkbox input as a boolean.

    - `colombia_sanitize_script`: Sanitizes HTML input for allowed tags.
    - `colombia_sanitize_multiple_category`: Sanitizes multiple categories input.
10. **Related Posts (`colombia_related_post`)**
    - Displays related posts from the same category under single post pages based on category matching.
11. **Excerpt Length**
    - Filters the default excerpt length to a custom length of 22 words.

Key Logic:

- **Theme Setup**: Registers default features and supports key WordPress features, like featured images and custom menus.
- **Widget Areas**: Enables customization of the sidebar and footer by adding widget areas.
- **Social Menu**: Dynamically displays social media links through the `wp_nav_menu` function.
- **Scripts and Styles**: Efficiently loads necessary styles and scripts for a responsive, functional theme experience.
- **Sanitization**: Ensures that all user input in the Customizer and other areas is properly sanitized for security and reliability.
- **Related Posts**: Dynamically fetches and displays posts related to the current post's category.

This file serves as the core of the **Colombia** theme's functionality, enabling customization, layout control, and essential WordPress feature support like widgets, navigation menus, and post handling.

18. header.php

File Purpose:
This file generates the header section of your WordPress theme, which includes the `<head>` section and everything up to the content area of the site.

HTML Structure:

- `<!doctype html>`: Declares the document type and version of HTML.
- `<html>`: The root element of the document.
- `<head>`: Contains meta-information about the document, such as character encoding and viewport settings.

Meta Tags:

- `<meta charset="<?php bloginfo( 'charset' ); ?>">`: Sets the character encoding for the website, using the charset specified in WordPress settings.
- `<meta name="viewport" content="width=device-width, initial-scale=1">`: Ensures the site is responsive and scales correctly on different devices.

WordPress Functions:

- `<?php wp_head(); ?>`: Inserts necessary scripts, styles, and meta tags required by plugins and WordPress itself.

Body and Page Container:

- `<body <?php body_class(); ?>>`: Adds WordPress-specific classes to the `<body>` tag to help with styling and scripting.
- `<?php wp_body_open(); ?>`: A hook for inserting content right after the opening `<body>` tag.

Skip Link:

- `<a class="skip-link screen-reader-text" href="#primary">`: Provides a link that allows users to skip directly to the main content, improving accessibility.

Header Section:

- `<header id="masthead" class="site-header">`: Marks the start of the header section.

Top Header Conditional Display:

- Checks if the top header section is enabled through theme settings (`colombia_enable_top_header`).
- Displays the top header section with a menu and social links if enabled.

Top Header Menu:

- `<?php wp_nav_menu(); ?>`: Displays the top menu if it is enabled in theme settings and if a menu is assigned to the 'top-menu' location.

Social Links and Search Icon:

- Displays social links and search functionality in the top header if enabled through theme settings.

Main Header Section:

- `<div class="main-header-wrap">`: Wraps the main header elements and optionally includes a background image.

Site Branding:

- Displays the site's custom logo and title. The title is displayed as an `<h1>` on the homepage and as a `<p>` elsewhere.
- Shows the site description if it is set or if the site is in customizer preview mode.

Main Navigation Menu:

- `<nav id="site-navigation" class="main-navigation">`: Contains the primary navigation menu.
- `<?php wp_nav_menu(); ?>`: Outputs the main menu assigned to the 'menu-1' location.

Search Functionality:

- Provides a search box that can be toggled visible or hidden, depending on the theme settings.

Closing Tags:

- Closes the header section and main wrapper with appropriate closing tags.

19. Index.php

File Purpose: This is a generic template file used in a WordPress theme. It serves as a fallback to display content when no more specific template file is available.

It is one of the two essential files required for a WordPress theme, the other being `style.css`.

Header Inclusion:

- `get_header();`: Includes the header template file (`header.php`) at the top of the page.

Container and Main Wrapper:

- `<div class="container">`: A wrapper for the main content area to provide layout and styling.
- `<div class="main-wrapper">`: Contains the main content and sidebar.

Main Content Area:

- `<main id="primary" class="site-main ct-post-wrapper">`: Marks the main content section of the page.

  Loop for Displaying Posts:

- `if ( have_posts() ) :`: Checks if there are posts available to display.
- `if ( is_home() && ! is_front_page() ) :`: Checks if the current page is the home page but not the front page, and if true, displays the page title.
- `while ( have_posts() ) :`: Starts the loop to iterate over each post.
- `the_post();`: Sets up post data for the current post in the loop.
- `get_template_part( 'template-parts/content', get_post_type() );`: Includes the template part specific to the post type. For example, if the post type is 'post', it will include `template-parts/content-post.php`.

  Pagination:

- `the_posts_navigation();`: Displays navigation links to move between pages of posts, if applicable.

  No Posts Scenario:

- `else :`: Handles the case when no posts are available.
- `get_template_part( 'template-parts/content', 'none' );`: Includes a template part for displaying a message or content when no posts are found (e.g., 'No posts found').

  Sidebar:

- `get_sidebar();`: Includes the sidebar template file (`sidebar.php`) to display widgets or additional content.

  Footer Inclusion:

- `get_footer();`: Includes the footer template file (`footer.php`) at the bottom of the page.

20. page.php

File Purpose: This template is used to display all pages in WordPress by default. It handles the layout and presentation of static pages.
Header Inclusion:

- `get_header();`: Includes the header template file (`header.php`) at the top of the page.

  Container and Wrapper:

- `<div class="container">`: Wraps the main content area for layout and styling purposes.
- `<div class="site-wrapper">`: Contains the main content section.

Main Content Area:

- `<main id="primary" class="site-main">`: Marks the main content area of the page.

Loop for Displaying Page Content:

- `while ( have_posts() ) :`: Starts the loop to iterate over posts. For pages, this will typically handle a single page's content.
- `the_post();`: Sets up post data for the current page.
- `get_template_part( 'template-parts/content', 'page' );`: Includes a template part specific to displaying page content. This will typically load `template-parts/content-page.php`.

Comments Section:

- `if ( comments_open() || get_comments_number() ) :`: Checks if comments are enabled or if there are existing comments for the page.
- `comments_template();`: Includes the comments template file (`comments.php`) to display the comments section if comments are enabled or present.

Closing Tags:

- Closes the `main` and `container` divs.

Footer Inclusion:

- `get_footer();`: Includes the footer template file (`footer.php`) at the bottom of the page.

21. readme.txt

A `readme.txt` file is crucial for providing essential information about your WordPress plugin or theme.

Plugin/Theme Name and Description

- **Name:** Clearly state the name of your plugin or theme.
- **Description:** Provide a brief overview of what it does, its features, and benefits.

Installation Instructions

- **Basic Installation:** Explain how to install the plugin or theme, typically through the WordPress admin interface or via FTP.

- **Requirements:** Mention any necessary prerequisites like PHP version, WordPress version, or any other dependencies.

Changelog

- **Version History:** List changes, improvements, and fixes in each version update. This helps users see what's new or changed.

Credits and Acknowledgments

- **Contributors:** Acknowledge anyone who contributed to the development of the plugin or theme.
- **Libraries and Resources:** Mention any external libraries, tools, or resources used in the development.

License Information

- **License Type:** Specify the license under which the plugin or theme is distributed (e.g., GPL, MIT).
- **License Details:** Include any relevant details or links to the full license text.

Author Information

Contact Details: Provide contact information for the author or development team, including websites or social media profiles.

22. search.php

File Purpose: This template is used to display search results pages in WordPress. It shows the results of a search query.
Header Inclusion:

- `get_header();`: Includes the header template file (`header.php`) at the top of the page.

Container and Main Wrapper:

- `<div class="container">`: A wrapper for layout and styling.
- `<div class="main-wrapper">`: Contains the main content area and sidebar.

Main Content Area:

- `<main id="primary" class="site-main ct-post-wrapper">`: Marks the primary section where search results will be displayed.

Search Results Header:

- `if ( have_posts() ) :`: Checks if there are posts matching the search query.
- `<header class="page-header">`: Contains the header for the search results page.
- `<h1 class="page-title">`: Displays the title of the search results page.
- `printf( esc_html__( 'Search Results for: %s', 'colombia' ), '<span>' . get_search_query() . '</span>' );`: Outputs a message showing the search query used, with translation support.

Loop for Displaying Search Results:

- `while ( have_posts() ) :`: Starts the loop to iterate over search results.
- `the_post();`: Sets up post data for each search result.
- `get_template_part( 'template-parts/content', 'search' );`: Includes a template part specific to displaying each search result. If customized, this could be `content-search.php`.

Pagination:

- `the_posts_navigation();`: Displays navigation links to move between pages of search results, if applicable.

No Results Scenario:

- `else :`: Handles the case when no search results are found.
- `get_template_part( 'template-parts/content', 'none' );`: Includes a template part for displaying a message when no results are found (e.g., 'No results found').

Sidebar:

- `get_sidebar();`: Includes the sidebar template file (`sidebar.php`) to display widgets or additional content.

Closing Tags:

- Closes the `main` and `container` divs.

Footer Inclusion:

- `get_footer();`: Includes the footer template file (`footer.php`) at the bottom of the page.

23. sidebar.php

File Purpose:

- This template is used to display the sidebar containing widgets. It is a common part of WordPress themes for adding additional content or functionality.

  Check for Active Sidebar:

- `if ( ! is_active_sidebar( 'sidebar-1' ) ) { return; }`: Checks if the sidebar with the ID `'sidebar-1'` has active widgets. If no widgets are active, the sidebar is not displayed, and the code execution stops.

  Sidebar Container:

- `<aside id="secondary" class="widget-area">`: Defines the `<aside>` element with an ID of `'secondary'` and a class of `'widget-area'`. This contains the widget area.

  Display Widgets:

- `<?php dynamic_sidebar( 'sidebar-1' ); ?>`: Outputs the widgets assigned to the `'sidebar-1'` widget area. The `dynamic_sidebar()` function fetches and displays widgets that have been added to this sidebar in the WordPress admin.

24. single.php

File Purpose: This template is used to display individual single posts. It handles the layout and presentation for a single post page.
Header Inclusion:

- `get_header();`: Includes the header template file (`header.php`) at the top of the page.

Container and Main Wrapper:

- `<div class="container">`: A wrapper for layout and styling.
- `<div class="main-wrapper">`: Contains the main content area and sidebar.

Main Content Area:

- `<main id="primary" class="site-main">`: Marks the main content section where the single post will be displayed.

Loop for Displaying Single Post:

- `while ( have_posts() ) :`: Starts the loop to iterate over posts. For single posts, this loop will typically handle one post.
- `the_post();`: Sets up post data for the current post.

Post Content:

- `get_template_part( 'template-parts/content', get_post_type() );`: Includes a template part specific to displaying post content. It loads a file named `content-{post-type}.php`, where `{post-type}` is replaced with the post type (e.g., `content-post.php`).

Post Navigation:

- `the_post_navigation();`: Displays navigation links to the previous and next posts.
- `array( 'prev_text' => ... , 'next_text' => ... )`: Customizes the text and HTML structure for the navigation links, including labels for previous and next posts.

Related Posts:

- `do_action('colombia_related_posts');`: Executes a custom action hook for displaying related posts. This allows additional code or plugins to add related posts functionality.

Comments Section:

- `if ( comments_open() || get_comments_number() ) :`: Checks if comments are enabled or if there are existing comments for the post.
- `comments_template();`: Includes the comments template file (`comments.php`) to display the comments section if comments are enabled or present.

Sidebar:

- `get_sidebar();`: Includes the sidebar template file (`sidebar.php`) to display widgets or additional content alongside the post.

Closing Tags:

- Closes the `main` and `container` divs.

Footer Inclusion:

- `get_footer();`: Includes the footer template file (`footer.php`) at the bottom of the page.