A **Content Management System (CMS)** is a software application that helps users create, manage, and modify digital content on a website without needing to write code directly. It simplifies the process of building and maintaining websites, making it accessible to both developers and non-technical users.

Let’s break it down in detail:

**1. Core Concept of a CMS**

A CMS acts as a middle layer between users and the website's backend. It provides a graphical interface that allows users to manage content without touching the underlying code or database.

* **Content:** This includes text, images, videos, documents, and other digital assets.
* **Management:** Provides tools to organize, edit, publish, and control access to content.
* **System:** The structured environment that handles everything behind the scenes, like storing content in databases and delivering it to the website frontend.

**Popular CMS Platforms**

* **WordPress:** Powers over 40% of websites globally. Great for blogs, business sites, and even e-commerce.
* **Joomla:** Flexible for complex sites but has a steeper learning curve.
* **Drupal:** Best for highly customized, large-scale projects.
* **Shopify/WooCommerce:** Specialized in e-commerce.

2. How to use child theme in wordpress?

Create a new folder in the themes directory and name it anything you like. For our example, let’s name it “child-theme”. Inside this folder create a style.css and a functions.php file. Open the stylesheet and use the following to create the child theme:

|  |
| --- |
| /\* |
|  | Theme Name: Twenty Fourteen Child |
|  | Theme URI: http://mytheme.com |
|  | Description: My Child Theme |
|  | Author: Daniel Pataki |
|  | Author URI: http://danielpataki.com |
|  | Template: twentyfourteen |
|  | Version: 1.0.0 |
|  | Tags: light, dark, two-columns, right-sidebar, responsive-layout, accessibility-ready |
|  | Text Domain: my-child-theme |
|  | \*/ |
|  |  |
|  |  |
|  | /\* =Theme customization starts here  When file is loaded, wordpress looks for(=search) it in the child theme first. If it doesn’t exist, the same file from the parent theme is loaded. To load the styles of the parent theme, Feel free to copy-paste the code below into your child theme’s functions.php. Just be aware that this loads the styles of the parent.  add\_action( 'wp\_enqueue\_scripts', 'my\_parent\_styles' );  function my\_parent\_styles() {  wp\_enqueue\_style( 'parent-style', get\_template\_directory\_uri().'/style.css' );  } |

3. **The Query and the Loop**

**The query** is the system that “knows” which posts to show on a page and **the loop** is the part that actually goes through each post and displays them.

Let’s look at the basic code for a loop and go through it line by line:

|  |  |
| --- | --- |
|  | <?php if( have\_posts() ) ?> |
|  |  |
|  | <?php while( have\_posts() ) : the\_post() ?> |
|  | <div <?php post\_class() ?>> |
|  | <h2><?php the\_title() ?></h2> |
|  | <div class='content'> |
|  | <?php the\_content() ?> |
|  | </div> |
|  | </div> |
|  | <?php endwhile; ?> |
|  |  |
|  | <?php else: ?> |
|  | Oh no, there are no posts here |
|  | <?php endif ?> |

4. **Custom Queries**

You can create a custom query using the WP\_Query class.  Here’s an example which shows scheduled posts from a specific category. You can use this to show a “Coming soon in this category” section.

|  |
| --- |
| <?php |
|  | $args = array( |
|  | 'post\_type' => 'post', |
|  | 'post\_status' => 'future', |
|  | 'category\_name' => 'app\_reviews', |
|  | 'posts\_per\_page' => 3 |
|  | ); |
|  | $coming\_soon = new WP\_Query( $args ); |
| ?> |  |

Now that we have a custom query we can use a custom loop to display the content. All we need to do is prefix the have\_posts() and the\_post() functions with the name of the variable that holds the query and an “arrow”:

|  |  |
| --- | --- |
|  | <?php if( $**coming\_soon**->have\_posts() ) ?> |
|  |  |
|  | <ul> |
|  | <?php while( $**coming\_soon**->have\_posts() ) : $**coming\_soon**->the\_post() ?> |
|  | <li <?php post\_class() ?>> |
|  | <a href='<?php the\_permalink() ?>'><?php the\_title() ?></a> |
|  | </li> |
|  | <?php endwhile; ?> |
|  | </ul> |
|  |  |
|  | <?php endif ?> |

5. **Hooks**

Hooks allow you to modify how WordPress works. There are two types of hooks, action hooks, and filter hooks. Actions allow you to run a function of your own in specific places in the WordPress code. For example, by using a hook you can execute one of your own functions when WordPress publishes a post. This allows you to notify the author, for example.

|  |  |
| --- | --- |
|  | add\_action( 'wp\_footer', 'my\_tracking\_code' ); |
|  |  |
|  | function my\_tracking\_code() { |
|  | ?> |
|  | // Paste Google Analytics code here |
|  | <?php |
|  | } |

Filters allow you to modify data before it is used. For example, you could use a filter to modify the text shown to the user when a post is saved. Instead of “Post draft updated,” you could modify this to say “Your draft has been saved.”

|  |  |
| --- | --- |
|  | add\_filter( 'the\_content', 'my\_content\_filter' ); |
|  |  |
|  | function my\_content\_filter( $content ) { |
|  | $content .= ' (checked by Daniel Pataki)'; |
|  | } |