WordPress Cheatsheet

A quick reference guide for WordPress users, developers, and administrators covering key concepts, functions, and best practices.

Core Concepts

WordPress Fundamentals

WordPress: A free and open-source content management system (CMS) based on PHP and MySQL.
Posts: Dynamic content entries displayed in reverse chronological order.
Pages: Static content like 'About Us' or 'Contact' pages.
Themes: Control the visual design of your website.
Plugins: Extend WordPress functionality with additional features.
Widgets: Add content and features to sidebars and other widget areas.
Users: Manage different user roles and permissions.

Key WordPress Files & Directories

`wp-config.php`	Contains database connection details, security keys, and other settings.
`wp-content/themes/`	Directory where WordPress themes are stored.
`wp-content/plugins/`	Directory where WordPress plugins are stored.
`.htaccess`	Server configuration file (if using Apache), used for permalinks and security.
`index.php`	Main WordPress file that handles requests.

Admin Dashboard Sections

Posts	Create, edit, and manage blog posts.
Media	Upload and manage images, videos, and other media files.
Pages	Create and manage static pages.
Appearance	Customize your site's design with themes and widgets.
Plugins	Install and manage plugins to extend functionality.
Users	Manage user accounts and roles.
Settings	Configure general site settings, permalinks, and more.

Theme Development

Basic Theme Structure

A basic WordPress theme requires at least two files: `style.css` and `index.php`.
`style.css`: Contains theme metadata (name, author, version) and CSS styles.
`index.php`: Main template file that displays content.

Common Template Files

`header.php`	Contains the website header (doctype, `<html>`, `<head>`, opening `<body>` tag).
`footer.php`	Contains the website footer (closing `<body>` and `<html>` tags).
`sidebar.php`	Contains the sidebar content (widgets, navigation).
`single.php`	Template for displaying a single post.
`page.php`	Template for displaying a single page.
`archive.php`	Template for displaying archive pages (categories, tags, dates).
`functions.php`	Theme functions file for custom PHP code.

Theme Functions

`wp_head()`	Hook for adding content to the `<head>` section.
`wp_footer()`	Hook for adding content to the footer.
`the_title()`	Displays the post title.
`the_content()`	Displays the post content.
`the_permalink()`	Displays the post permalink.
`get_template_directory_uri()`	Returns the theme directory URI.

Plugin Development

Basic Plugin Structure

A basic WordPress plugin requires at least one PHP file with plugin metadata.
Plugin metadata includes Plugin Name, Version, Author, and Description, defined in comments at the top of the file.

Plugin Header Example

```php
<?php
/**
 * Plugin Name: My Awesome Plugin
 * Description: A brief description of the plugin.
 * Version: 1.0.0
 * Author: Your Name
 */
```

Key Plugin Functions

`add_action()`	Hooks a function to a specific action.
`add_filter()`	Hooks a function to a specific filter.
`register_activation_hook()`	Runs a function when the plugin is activated.
`register_deactivation_hook()`	Runs a function when the plugin is deactivated.
`wp_enqueue_scripts()`	Enqueues scripts and styles for the front-end.
`admin_enqueue_scripts()`	Enqueues scripts and styles for the admin area.

Example: Adding a Custom Action

```php
<?php
function my_custom_function() {
  echo '<p>Hello from my custom action!</p>';
}
add_action( 'wp_footer', 'my_custom_function' );
```

Security Best Practices

Core Security Measures

Keep WordPress, Themes, and Plugins Updated: Regularly update to patch security vulnerabilities.
Use Strong Passwords: Protect user accounts with strong, unique passwords.
Limit Login Attempts: Implement a login attempt limiter to prevent brute-force attacks.
Enable Two-Factor Authentication (2FA): Add an extra layer of security to user logins.
Regular Backups: Schedule regular backups of your WordPress files and database.
Use a Security Plugin: Consider using a security plugin like Wordfence or Sucuri Security.

Database Security

Change the Database Table Prefix:	Use a prefix other than the default `wp_`.
Protect `wp-config.php`:	Restrict access to `wp-config.php` file.

File System Security

Disable File Editing:	Prevent users from editing theme and plugin files through the WordPress admin panel.
Restrict File Uploads:	Limit file upload types to prevent malicious uploads.

Additional Hardening

Disable Directory Listing: Prevent directory listing to avoid revealing file structure.
Implement a Web Application Firewall (WAF): Use a WAF to filter malicious traffic.