SAURASHTRA UNIVERSITY · RAJKOT

CS-17: Content Management System using WordPress

B.C.A. (Honours) & B.C.A. (Honours with Research) — Semester 3 & 4 · Effective from June 2024

Credit: 4 | CCE: 50 | SEE: 50 | Total: 100 | Type: MAJOR

5 Units · 26 Diagrams | Gujarati Explanations for Hard Topics

Course Objectives

- Learn how to create custom themes and pages
- Work with custom post types and taxonomies
- Gain in-detail knowledge of the WordPress CMS backend
- Work with widgets and widget areas
- Work with default CMS functions and extend its core

Prerequisites

- Basic knowledge of web development (HTML/CSS/PHP basics) and general CMS concepts

Course Outcomes

- Able to configure and administer the WordPress CMS backend confidently
- Able to decide when to use a Custom Post Type vs. a Custom Field
- Able to extend the WordPress core to match project requirements (hooks, shortcodes, widgets)
- Able to design and build a complete, dynamic WordPress theme from scratch

Reference Book

WordPress to Go: How to Build a WordPress Website on Your Own Domain, from Scratch, Even If You Are a Complete Beginner — Sarah McHarry (Paperback)

Syllabus at a Glance

01

Introduction & Setup

- What is CMS
- Installation
- Dashboard
- Database

02

Theme

- Install/ Activate
- Customizer

03

Widget & Plugin

- Widget areas
- Plugins

04

Theme Dev

- Template files
- The Loop
- Template tags

05

Advanced Dev

- Hooks
- CPT/ Taxonomy
- Widget areas (code)

1 Introduction, Installation & Configuration

1.1 What is a Content Management System (CMS)?

A **Content Management System** is software that lets people create, edit, organize, and publish digital content (text, images, video) without needing to write code for every page. It separates *content* (stored in a database) from *presentation* (theme/template files), so non-technical users can manage a website through a visual dashboard while the underlying HTML/CSS/PHP is generated dynamically.

Think of a CMS in two layers: the **back-end** (admin dashboard where content is authored and stored in a database) and the **front-end** (public-facing pages, assembled at request-time from that stored content plus the active theme's templates). This separation is what allows the same blog post to look completely different after simply switching a theme.

ગુજરાતીમાં સમજ

CMS એટલે શું? CMS (કન્ટેન્ટ મેનેજમેન્ટ સિસ્ટમ) એ એવું સોફ્ટવેર છે જેની મદદથી કોઈપણ વ્યક્તિ કોડિંગ જાણ્યા વગર વેબસાઈટ પર કન્ટેન્ટ (લખાણ, ફોટા, વિડિયો) બનાવી, બદલી અને પ્રકાશિત કરી શકે છે.

CMS ના બે ભાગ હોય છે: બેક-એન્ડ (એડમિન ડેશબોર્ડ, જ્યાં કન્ટેન્ટ ડેટાબેઝમાં સંગ્રહ થાય છે) અને ફ્રન્ટ-એન્ડ (વેબસાઈટની જાહેર બાજુ, જે થીમના ટેમ્પલેટ પ્રમાણે તૈયાર થાય છે). એટલે જ થીમ બદલવાથી એ જ પોસ્ટ સાવ અલગ દેખાય છે.

Introduction to WordPress

WordPress is a free, open-source CMS written in PHP and backed by a MySQL/MariaDB database. Originally launched in 2003 as a blogging tool, it has grown into the world's most popular CMS, powering a large share of all websites on the internet — from personal blogs to large e-commerce stores (via WooCommerce) and enterprise sites.

WordPress ships in two flavours students should be able to distinguish: **WordPress.org** (self-hosted — you download the software, install it on your own server, and have full

control/access to files & database — this is what this syllabus covers) versus **WordPress.com** (a hosted service where Automattic manages the server for you, with more restrictions on themes/plugins on free plans).

Features of WordPress

- **Open source & free** — free to download, use, and modify (GPL license)
- **Themes** — thousands of free/paid designs to change the site's look instantly
- **Plugins** — extend functionality (SEO, forms, e-commerce, caching) without coding
- **User-friendly dashboard** — WYSIWYG block editor (Gutenberg) for content creation
- **SEO friendly** — clean permalink structure, meta tags, sitemap support
- **Responsive & mobile-ready** themes out of the box
- **Multi-user & role management** — Admin, Editor, Author, Contributor, Subscriber
- **Media management** — built-in library for images, audio, video, documents
- **Custom Post Types & Taxonomies** — model any kind of content (products, portfolios, events)
- **Large community & documentation** — extensive support ecosystem

Advantages of WordPress

- Free and open source — low cost of ownership
- Easy for beginners; no coding required for basic sites
- Massive plugin/theme ecosystem — rapid development
- SEO-friendly architecture
- Scalable — from a single blog to enterprise (news portals, e-commerce)
- Regular security updates and an active community

Disadvantages of WordPress

- Frequent plugin/theme updates can cause compatibility conflicts
- Security is dependent on keeping core/plugins updated (common attack target due to popularity)
- Heavy use of plugins can slow site performance
- Customization beyond a point requires PHP/CSS/JS knowledge

- Database-driven — needs proper backup/maintenance discipline

1.2 Installation of WordPress

WordPress can be installed **locally** (for learning/development) using a stack like **XAMPP/WAMP** (Apache + MySQL + PHP), or on **live hosting** via cPanel/Softaculous or manual upload.

Local Installation Steps (XAMPP)

1. Install XAMPP and start the **Apache** and **MySQL** modules.
2. Download the latest WordPress ZIP from *wordpress.org* and extract it into `C:\xampp\htdocs\mysite`.
3. Open `phpMyAdmin` (http://localhost/phpmyadmin) and create a new database, e.g. `wp_mysite`.
4. Visit `http://localhost/mysite` in a browser — the WordPress installer (the "famous 5-minute install") launches.
5. Enter database name, username (`root`), password, and host (`localhost`).
6. Set the site title, admin username, password, and email — click **Install WordPress**.
7. Log in to `/wp-admin` with the created credentials.

> **Live hosting shortcut:** most hosting providers offer a one-click WordPress installer (Softaculous/Installatron) inside cPanel, which automates database creation and file setup.

> **Common installation error:** "Error establishing a database connection" almost always means wp-config.php has the wrong DB name/username/password/host — double-check these four values first.

wp-config.php

```
define( 'DB_NAME', 'wp_mysite' );
define( 'DB_USER', 'root' );
define( 'DB_PASSWORD', '' );
define( 'DB_HOST', 'localhost' );
define( 'DB_CHARSET', 'utf8' );

// Authentication unique keys and salts
define( 'AUTH_KEY', 'put your unique phrase here' );
$table_prefix = 'wp_';
```

These 4 values (DB_NAME, DB_USER, DB_PASSWORD, DB_HOST) are the #1 cause of "Error establishing a database connection"

Fig 1.0 — wp-config.php: the four database constants that installation writes automatically

ગુજરાતીમાં સમજ

ઈન્સ્ટોલેશન એરર કેમ આવે છે? મોટાભાગની "Database connection" ભૂલ એટલા માટે આવે છે કારણ કે `wp-config.php` ફાઈલમાં ડેટાબેઝનું નામ, યુઝરનેમ, પાસવર્ડ અથવા હોસ્ટ ખોટું લખાયેલું હોય છે. phpMyAdmin માં જઈને ડેટાબેઝનું સાચું નામ ચેક કરો અને wp-config.php માં એ જ નામ મેચ કરો.

1.3 WordPress Directory & File Structure

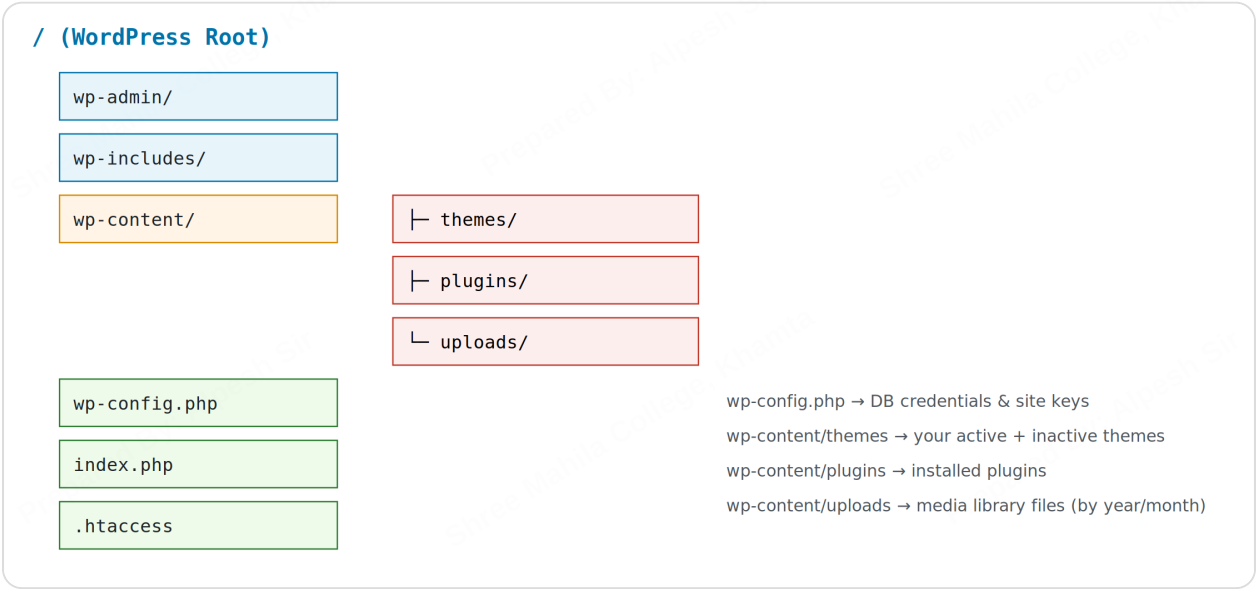


Fig 1.1 — Root-level folders/files of a WordPress installation

FOLDER / FILE	PURPOSE
	Core files that power the admin dashboard (never edit)

`wp-admin/`	
`wp-includes/`	Core WordPress library files, classes, functions (never edit)
`wp-content/themes/`	All installed themes live here — this is where custom theme development happens
`wp-content/plugins/`	All installed plugins
`wp-content/uploads/`	Media library files, organized by year/month
`wp-config.php`	Database connection details, security keys, debug settings
`index.php`	Front controller that loads the WordPress bootstrap
`.htaccess`	Apache rewrite rules for pretty permalinks

> Golden rule: the only folder you should ever edit inside is `wp-content`. Everything in `wp-admin` and `wp-includes` gets overwritten on every core update.

1.4 Dashboard Overview

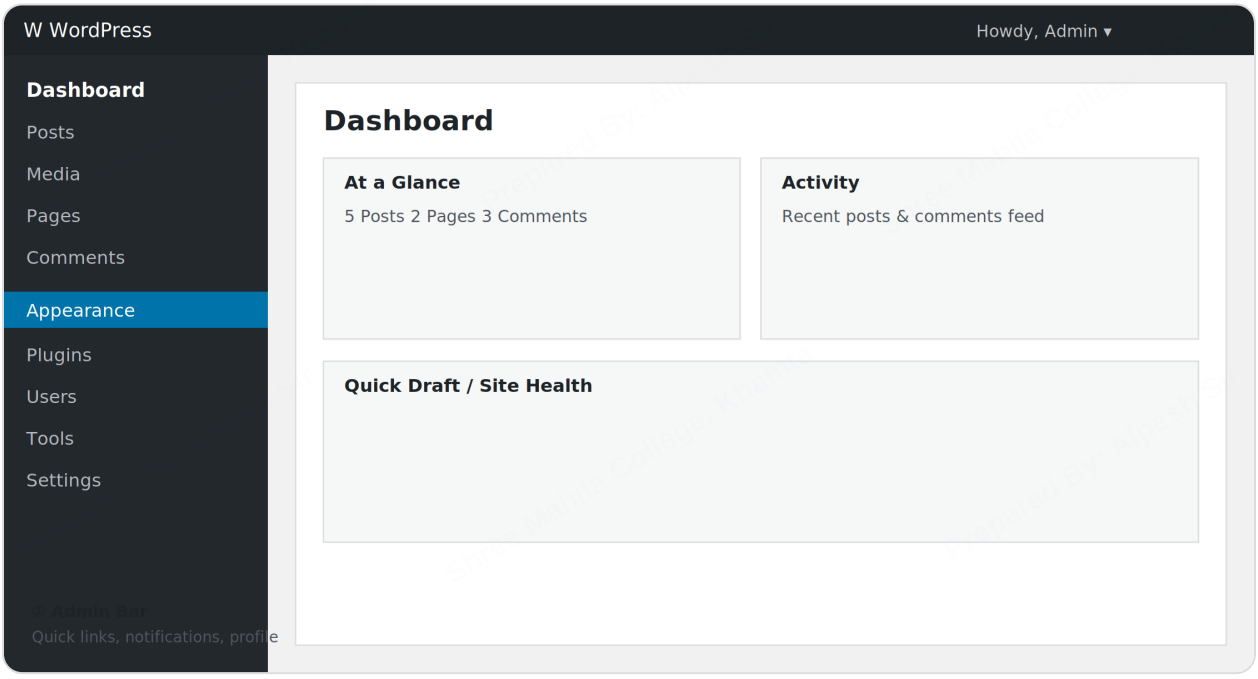


Fig 1.2 — Anatomy of the wp-admin dashboard: Admin bar, left navigation menu, main work area

The dashboard (`/wp-admin`) is the control panel of the site. The left menu gives access to **Posts, Media, Pages, Comments, Appearance, Plugins, Users, Tools, and**

Settings. The main panel shows widgets like "At a Glance" (content counts), "Activity," and "Quick Draft."

Adding, Editing & Deleting Content

- **Post**: Posts → Add New → write title/content → set Category & Tags → Publish. Edit via Posts → All Posts → click title. Delete moves it to Trash first (permanently deletable from Trash).
- **Page**: Similar to Posts but for static content (About, Contact) — Pages don't use categories/tags and support a Page Attributes panel (Template, Parent, Order).
- **Category**: Posts → Categories → add Name, Slug, Parent, Description. Categories are hierarchical (can have parent/child).
- **Tag**: Posts → Tags — flat (non-hierarchical) keywords used for cross-referencing content.
- **Media**: Media → Add New, or drag-and-drop directly inside the block editor. Supports images, PDFs, docs, audio, video; each file gets a Title, Alt Text, Caption, and Description, and can be attached to a specific post/page.

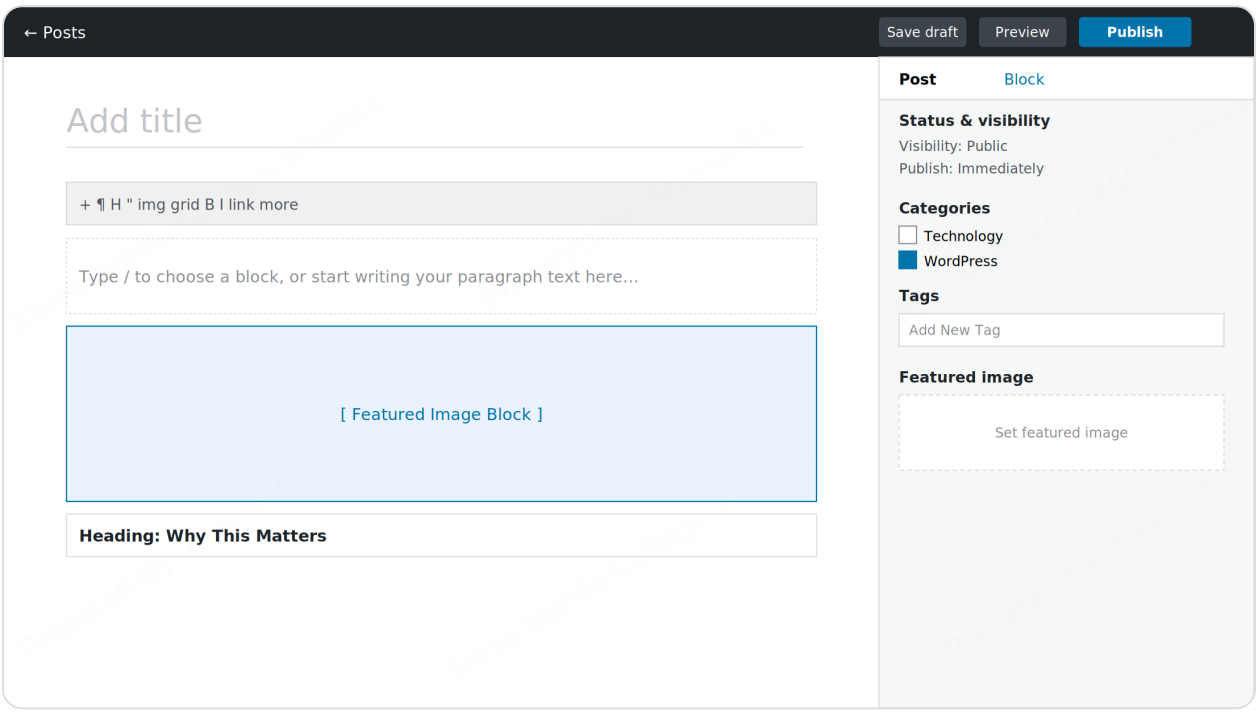


Fig 1.2a — Posts → Add New: block editor (left) + Post settings sidebar (right) with Status, Categories, Tags, Featured Image

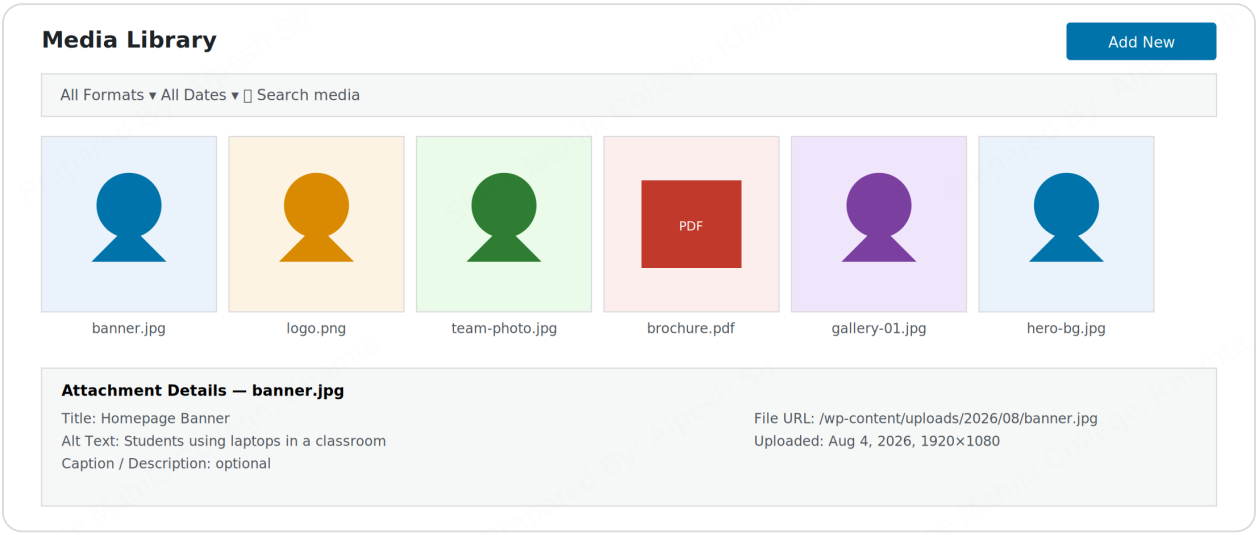


Fig 1.2b — Media Library grid view; clicking a file opens its Attachment Details (Title, Alt Text, URL)

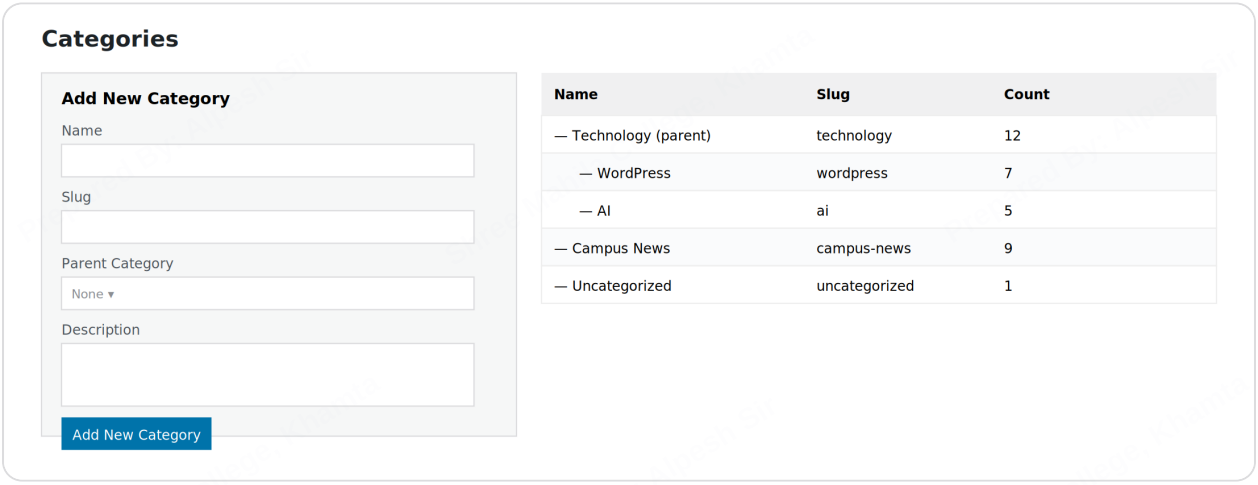


Fig 1.2c — Posts → Categories: add-new form (left) and the existing hierarchy with post counts (right)

Alt Text vs. Caption vs. Description (Media): Alt Text is read by screen readers and search engines (accessibility + SEO) and should describe the image; Caption displays visibly under the image; Description is an internal note, rarely shown on the front end.

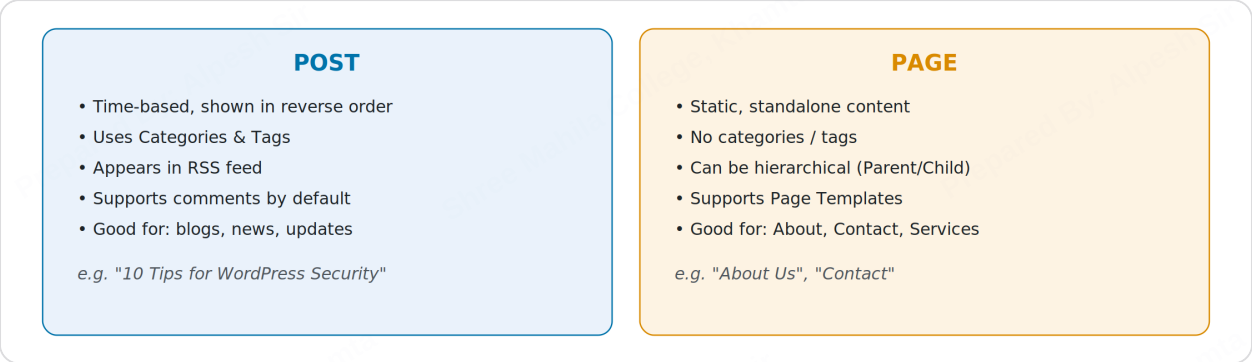


Fig 1.3 — Post vs. Page: when to use which

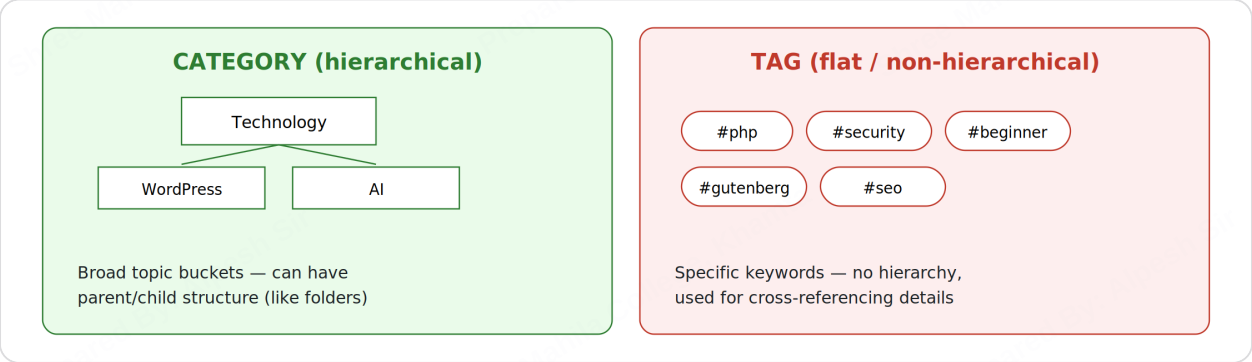


Fig 1.4 — Category (hierarchical) vs. Tag (flat)

1.5 Gutenberg — The Block Editor

Gutenberg is WordPress's default content editor (since v5.0), where every piece of content — a paragraph, image, or button — is a self-contained *block* that can be added, rearranged (drag handles), and styled individually.

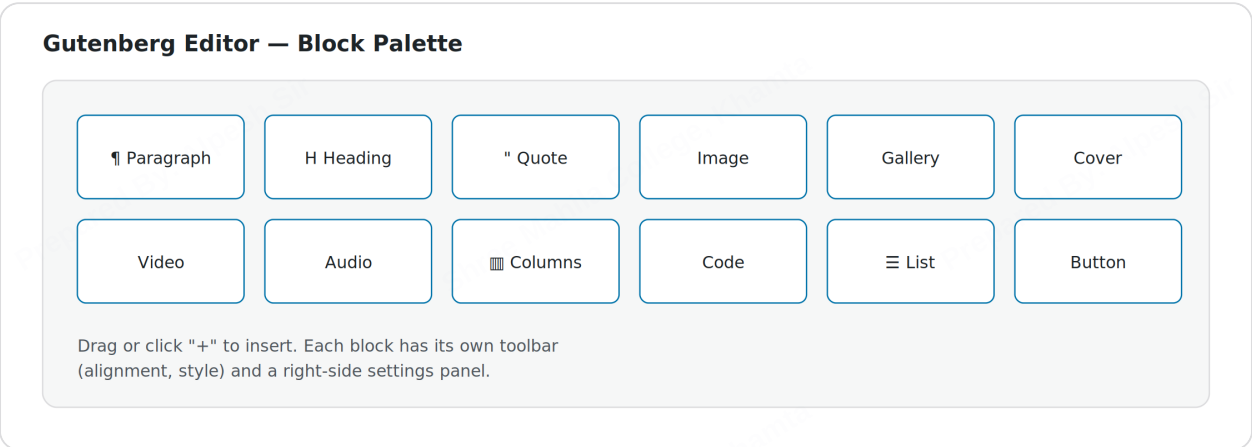


Fig 1.5 — Common Gutenberg blocks available from the "+" inserter

BLOCK	USE
Paragraph	Basic text content
Heading / Subheading	H1–H6 section titles
Quote	Blockquote styling for citations
Image / Cover Image	Single image or a full-width image with overlay text
Gallery	Multiple images in a grid
Video / Audio	Embed self-hosted or linked media
Columns	Multi-column layouts
Code	Preformatted code snippets
List	Bulleted/numbered lists
Button	Call-to-action links styled as buttons
Embeds	YouTube, Twitter/X, Instagram and other oEmbed content

1.6 User Roles & Capabilities

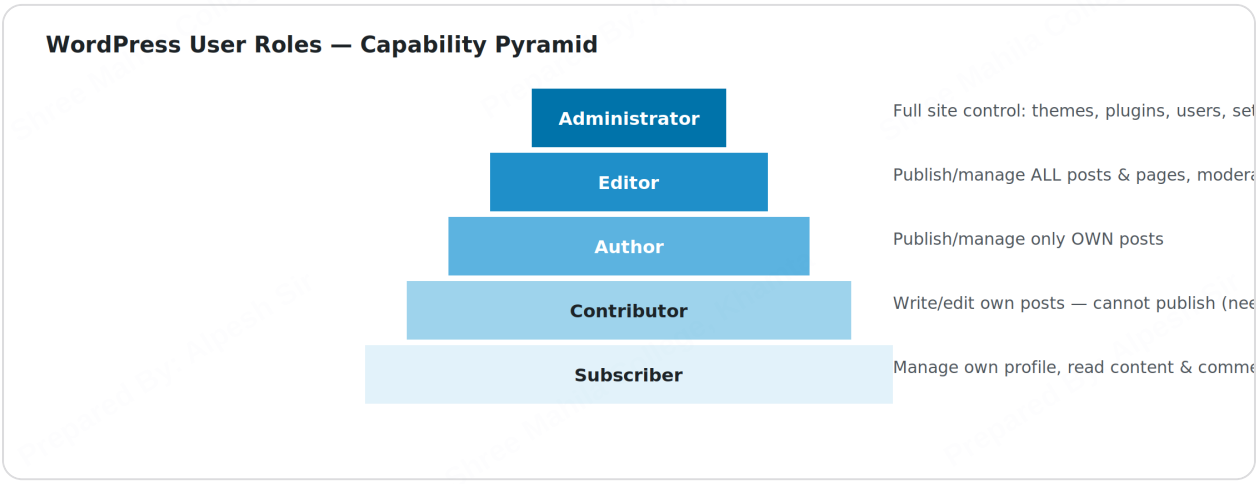


Fig 1.6 — Capability pyramid: each role down the list has fewer capabilities

ROLE	CAPABILITIES
Administrator	Full control — themes, plugins, users, settings, all content

Editor	Publish/manage all posts & pages (own and others'), moderate comments
Author	Publish and manage only their own posts
Contributor	Write and edit their own posts but cannot publish (awaits review)
Subscriber	Manage their own profile only; typically read content, leave comments

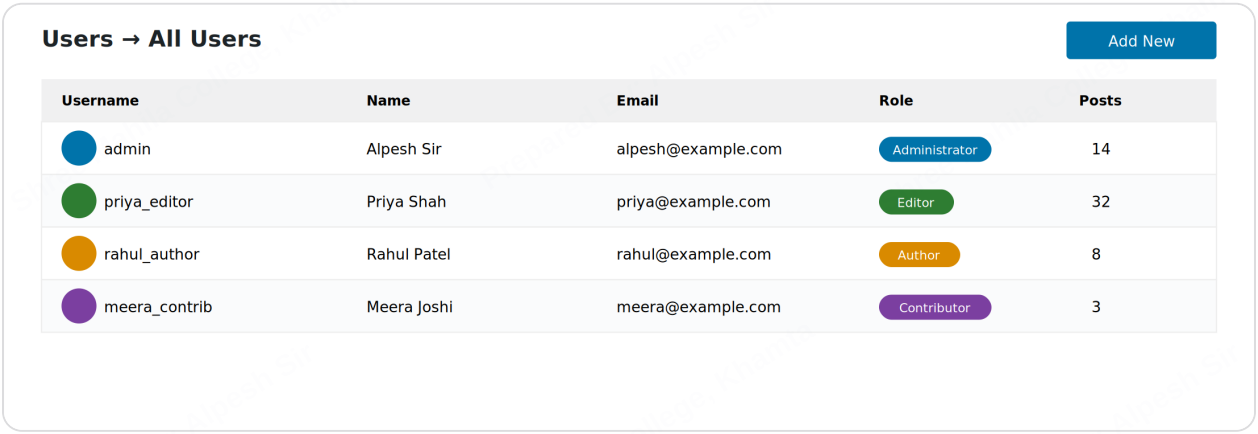


Fig 1.6a — Users → All Users: each account shows its assigned role and post count

1.7 Settings Overview

- **General** — Site Title, Tagline, WordPress/Site Address, Admin Email, Timezone, Date/Time format
- **Writing** — default post category/format, "Post via email," update services
- **Reading** — homepage display (latest posts / static page), posts per page, search engine visibility
- **Discussion** — comment settings, moderation, avatar display
- **Media** — image thumbnail/medium/large dimensions, uploads folder organization
- **Permalinks** — URL structure (Plain, Day/Name, Month/Name, Post name, Custom) — *"Post name"* is the SEO-recommended option

Settings → Permalinks — Structure Comparison

Plain	?p=123 (not SEO friendly)
Day and name	/2026/08/04/sample-post/
Month and name	/2026/08/sample-post/
Post name ★	/sample-post/ — recommended (clean & SEO friendly)
Custom Structure	/%category%/%postname%/ — user defined

Fig 1.7 — Permalink structures compared

ગુજરાતીમાં સમજ

Permalink એટલે શું? Permalink એટલે તમારી પોસ્ટ કે પેજની કાયમી (permanent) URL. જો "Plain" પસંદ કરો તો URL માં `?p=123` જેવું અઘરું દેખાય, જે ના તો યુઝરને સમજાય ના Google ને ગમે.

"Post name" પસંદ કરવાથી URL સાફ અને સમજી શકાય તેવું બને છે (દા.ત. `/my-first-post/`), જે SEO માટે શ્રેષ્ઠ ગણાય છે — તેથી જ પરીક્ષામાં અને પ્રેક્ટિકલમાં આ સેટિંગ સૌથી વધુ ભલામણ કરવામાં આવે છે.

1.8 Updating WordPress

- **One-click Update** — Dashboard → Updates → click "Update Now" (recommended, safest for minor/major releases)
- **Manual Update** — download the new version from wordpress.org, back up the database and files, replace `wp-admin` and `wp-includes` folders, then re-run `/wp-admin/upgrade.php`

> Always take a full backup (files + database) before any manual update.

1.9 Database Structure

WordPress uses a relational database (MySQL). Default core tables (prefix `wp_`):

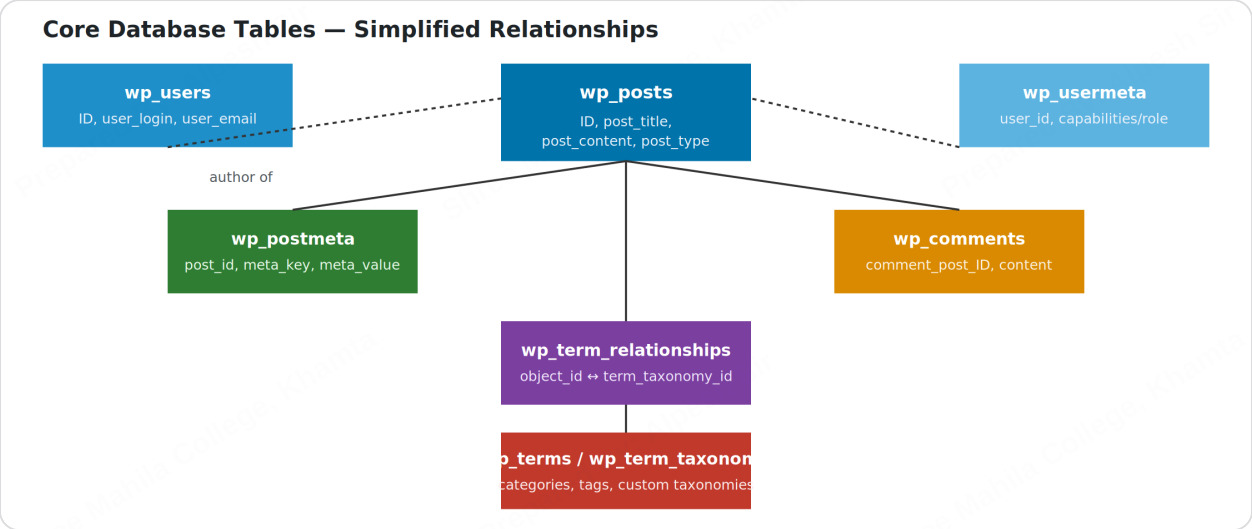


Fig 1.8 — Simplified relationships between core WordPress tables

TABLE	STORES
wp_posts	Posts, pages, custom post types, revisions, attachments
wp_postmeta	Custom fields / meta data linked to a post
wp_users	User accounts
wp_usermeta	Extra user meta data (role, nickname, capabilities)
wp_terms	Categories, tags, and custom taxonomy terms
wp_term_taxonomy	Describes the taxonomy each term belongs to
wp_term_relationships	Links posts to terms (categories/tags)
wp_comments	Comments on posts
wp_commentmeta	Meta data about comments
wp_options	Site-wide settings (site title, active theme, plugin options)
wp_links	Legacy "Blogroll" links feature

Notice that **a post's category/tag isn't stored directly on the post row** — it goes through `wp_term_relationships`, which links a `post ID` to a `term_taxonomy_id`. This indirection (many-to-many relationship) is what lets one post belong to multiple categories, and one category hold many posts, efficiently.

ગુજરાતીમાં સમજ

ડેટાબેઝ કેમ મહત્વનું છે? WordPress ની દરેક પોસ્ટ, પેજ, યુઝર અને સેટિંગ ડેટાબેઝ (MySQL) માં ટેબલ સ્વરૂપે સંગ્રહાય છે. `wp_posts` ટેબલમાં પોસ્ટ/પેજનું લખાણ હોય છે, `wp_users` માં યુઝરની માહિતી, અને `wp_options` માં સાઈટની સેટિંગ્સ (જેમ કે કઈ થીમ એક્ટિવ છે) સંગ્રહાય છે.

કેટેગરી અને પોસ્ટ વચ્ચેનો સંબંધ સીધો નથી હોતો — તે `wp_term_relationships` નામના અલગ ટેબલ મારફતે જોડાય છે, જેથી એક પોસ્ટને એકથી વધુ કેટેગરીમાં મૂકી શકાય.

Unit 1 — Lab Practicals

1. Install XAMPP/WAMP and set up a local WordPress installation using phpMyAdmin.
2. Explore the WordPress directory structure and identify wp-config.php settings.
3. Create 5 posts using different Gutenberg blocks (Heading, Image, Gallery, Quote, List).
4. Create categories (with a parent-child hierarchy) and tags; assign them to posts.
5. Upload media files (image, PDF) to the Media Library and attach one to a page.
6. Create 3 users with different roles (Editor, Author, Contributor) and test their permissions by logging in as each.
7. Configure General, Reading, and Permalink settings; verify the resulting URL structure.
8. Use phpMyAdmin to inspect wp_posts and wp_options tables after creating content.

2 Theme

2.1 What is a Theme?

A WordPress **theme** is a collection of template files, stylesheets, and (optionally) JavaScript/images that together define the visual presentation and layout of a site. Themes control how content stored in the database is displayed, without touching the content itself — the same posts can look completely different under a new theme.

ગુજરાતીમાં સમજ

થીમ શું છે? થીમ એ ફાઈલોનો સમૂહ છે જે નક્કી કરે છે કે વેબસાઈટ કેવી દેખાશે — રંગો, ફોન્ટ, લેઆઉટ. કન્ટેન્ટ (પોસ્ટ/પેજ) ડેટાબેઝમાં અલગ સંગ્રહાય છે, એટલે થીમ બદલવાથી કન્ટેન્ટ ડિલીટ નથી થતું, ફક્ત તેનો દેખાવ બદલાય છે.

2.2 How to Install & Activate a Theme

1. Go to **Appearance** → **Themes** → **Add New**.
2. Search the WordPress.org theme repository, or click **Upload Theme** to install a downloaded `.zip` file.
3. Click **Install**, then **Activate** to make it the live theme.
4. Alternatively, upload the theme folder directly to `wp-content/themes/` via FTP, then activate it from Appearance → Themes.

2.3 Theme Customize Options (Appearance → Customize)

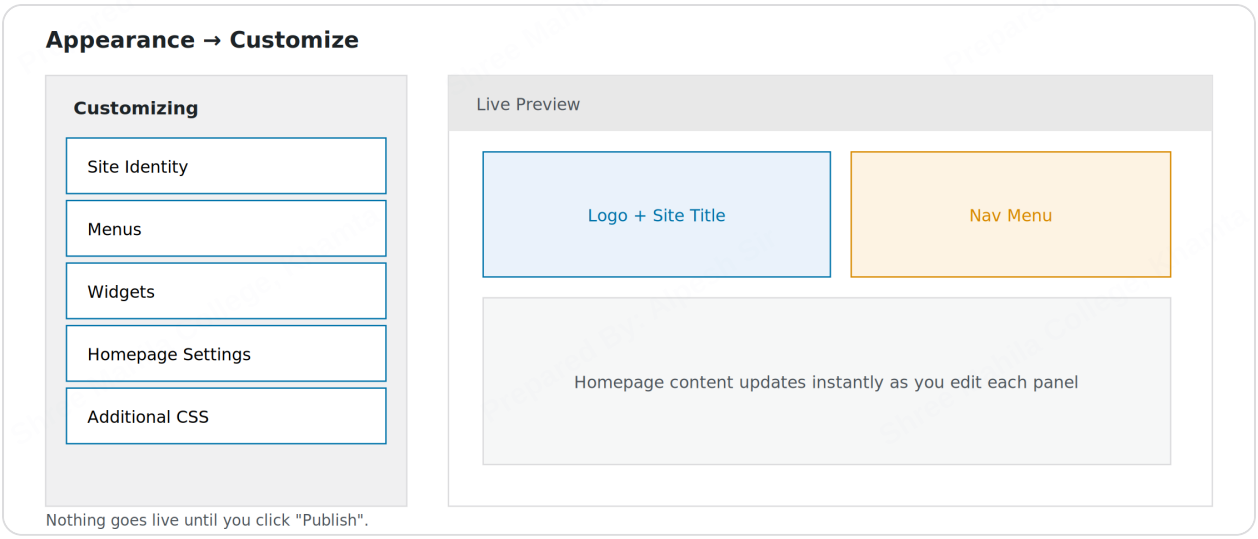


Fig 2.1 — The Customizer: sidebar panels update a live preview on the right

OPTION	WHAT IT CONTROLS
Site Identity	Site title, tagline, logo, and site icon (favicon)
Menus	Create/assign navigation menus to theme locations (header, footer)
Widgets	Add/arrange widgets into the theme's registered widget areas
Homepage Settings	Choose whether the homepage shows the latest posts or a static page
Additional CSS	Add custom CSS live, with instant preview, without editing theme files

> The Customizer applies changes in a live-preview pane — nothing goes live until you click **Publish**.

Building a Navigation Menu — Step by Step

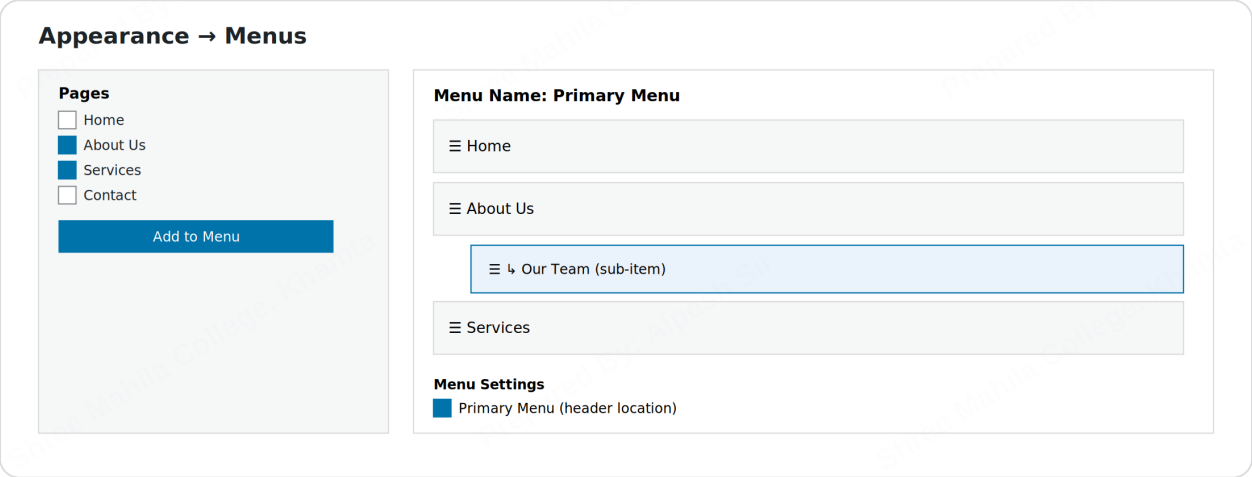


Fig 2.2 — Appearance → Menus: drag pages from the left into the menu structure, nest items to create dropdowns

1. Go to **Appearance → Menus → create a new menu**, give it a name (e.g. "Primary Menu").
2. Tick the Pages/Posts/Categories to add from the left panel, click **Add to Menu**.
3. Drag items to reorder; drag an item slightly right and drop it under another to make it a **sub-menu (dropdown)**.
4. Under **Menu Settings**, tick the theme location (e.g. "Primary Menu") to assign it, then **Save Menu**.

Unit 2 — Lab Practicals

1. Install and activate two different free themes from the repository; compare their layouts.
2. Set Site Identity — upload a logo and site icon, set title & tagline.
3. Create a custom navigation menu and assign it to the header location.
4. Configure the homepage to display a static "Home" page instead of the blog feed.
5. Add custom CSS via the Customizer to change the site's primary color/ fonts.

3 Widget & Plugin

3.1 What is a Widget & Widget Area?

A **widget** is a small, self-contained block of content or functionality (e.g., a search box, recent posts list, calendar) that can be dragged into a **widget area** — a theme-defined region such as a sidebar or footer — without writing code.

Available Widgets

Archive · Calendar · Categories · Navigation Menu · Meta · Pages · Recent Comments · Recent Posts · RSS · Search · Tag Cloud · Text · Image · Gallery · Video · Audio · Custom HTML

Widget Management

Managed via **Appearance → Widgets** (block-based widgets screen) or through the Customizer's Widgets panel. Widgets can be dragged between areas, reordered, and configured with their own settings (title, number of items, etc.).

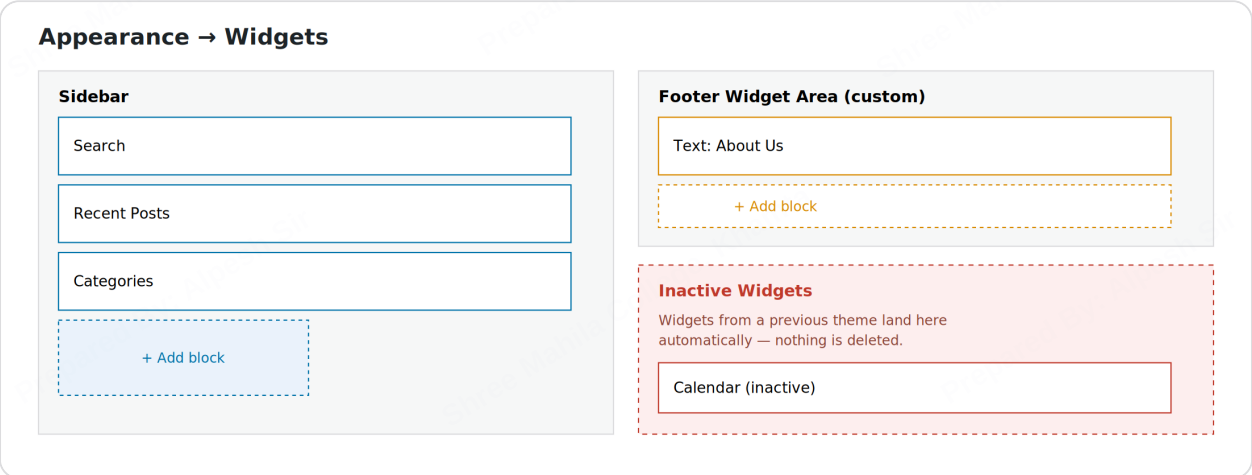


Fig 3.1a — Appearance → Widgets: multiple widget areas (Sidebar, Footer) plus the Inactive Widgets bin

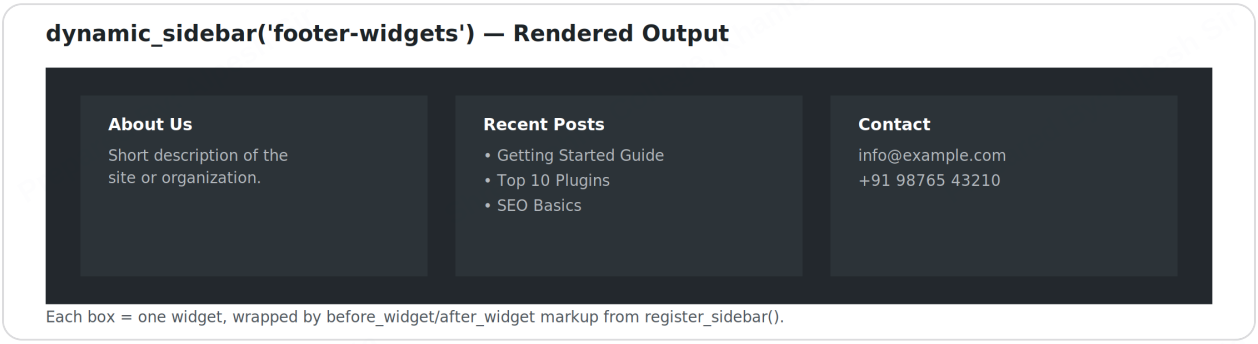


Fig 3.1 — Example of a footer widget area with three widgets rendered on the front end

Inactive Sidebar / Inactive Widgets

When a theme is switched, widgets from areas that no longer exist in the new theme are automatically moved to an **Inactive Widgets** bin rather than being deleted — so they can be restored later if you switch back or the new theme adds a matching area.

3.2 Plugins

What is a Plugin?

A **plugin** is a packaged piece of PHP code that hooks into WordPress core to add or extend functionality — e.g., SEO tools, contact forms, e-commerce, caching — without modifying core files, so features survive core/theme updates.

ગુજરાતીમાં સમજ

Widget અને **Plugin** વચ્ચે ફરક: Widget એ નાનું, તૈયાર content block છે (જેમ કે Search box) જે ફક્ત drag-and-drop કરીને sidebar/footer માં મૂકી શકાય — તેના માટે કોડિંગની જરૂર નથી.

Plugin એ મોટું, સંપૂર્ણ ફીચર ઉમેરતું સોફ્ટવેર છે (જેમ કે આખું Contact Form સિસ્ટમ, અથવા SEO ટૂલ) જે WordPress ની ક્ષમતા વધારે છે. ઘણી વખત એક plugin પોતાનું widget પણ બનાવે છે.

How to Install & Activate a Plugin

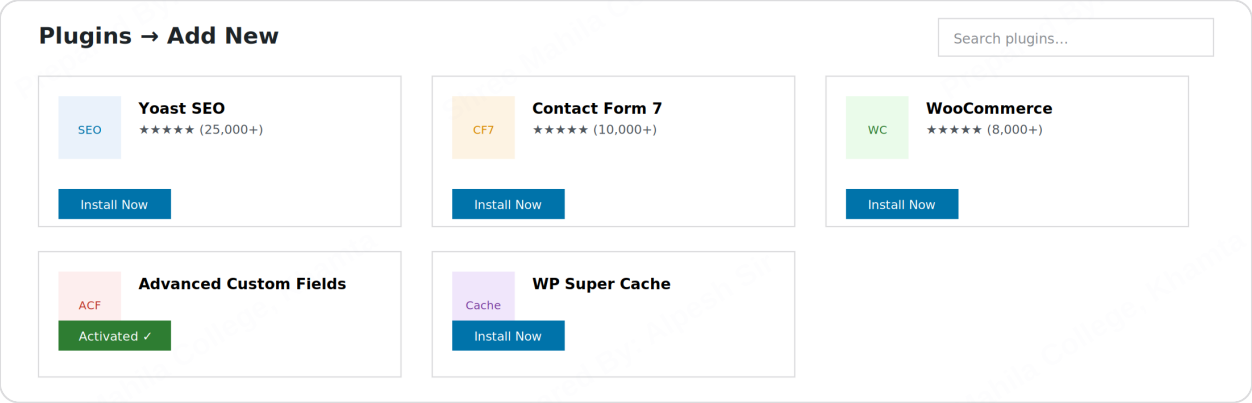


Fig 3.2a — Plugins → Add New: browse or search the repository, click Install Now then Activate

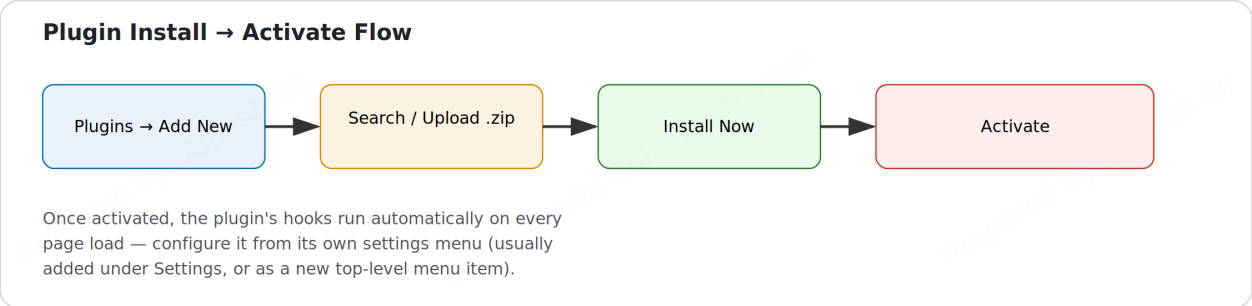


Fig 3.2b — Plugin install → activate workflow

1. **Plugins → Add New** → search the repository → **Install Now** → **Activate**.
2. Or **Upload Plugin** to install a downloaded `.zip` file.
3. Or upload the plugin folder to `wp-content/plugins/` via FTP, then activate from the Plugins screen.

Useful Plugins for a Website

PLUGIN	PURPOSE
SEO — Yoast	On-page SEO analysis, meta tags, XML sitemap
Contact Form 7	Build and manage contact forms
WooCommerce	Turns WordPress into a full e-commerce store
WP Super Cache	Page caching to improve performance
Regenerate Thumbnails	Rebuilds image thumbnail sizes after a theme change

Advanced Custom Fields (ACF)	Adds custom meta-box fields to posts/pages/CPTs
All-in-One WP Migration	Exports/imports an entire site (files + DB) for migration
Custom Post Type Widgets	Display custom post type content in widget areas

Unit 3 — Lab Practicals

1. Add Search, Recent Posts, and Categories widgets to the sidebar; reorder them.
2. Switch the active theme and observe widgets moving to the Inactive Widgets area; restore them.
3. Install and activate Yoast SEO; run an SEO analysis on a sample post.
4. Install Contact Form 7; build and embed a contact form on a page using its shortcode.
5. Install WooCommerce and add one sample product to the store.

4 Theme Development

4.1 Anatomy of a Theme

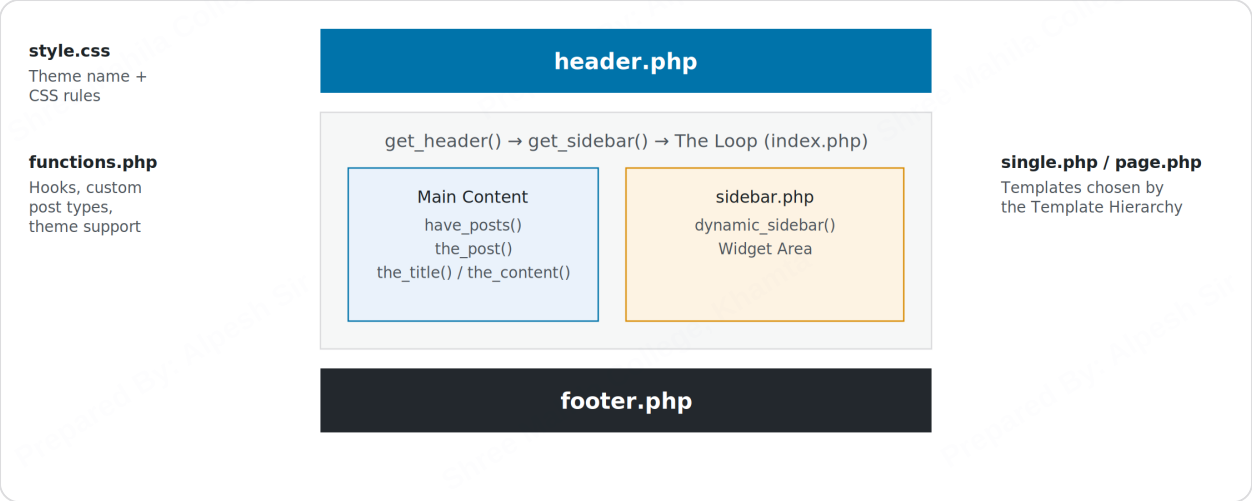


Fig 4.1 — How header.php, the Loop, sidebar.php and footer.php compose a page

Every page render in WordPress typically stitches together three re-usable pieces — `header.php`, `footer.php`, and optionally `sidebar.php` — around the page-specific template that runs the Loop.

Minimum Required & Common Template Files

FILE	ROLE
style.css	Required. Contains theme metadata comment header (Name, Author, Version) + CSS rules
index.php	Required fallback template — the main/blog template
header.php	Opening HTML, `<head>`, site branding, nav menu
footer.php	Closing HTML, footer widgets/credits, `wp_footer()` hook
sidebar.php	Sidebar widget area markup
page.php	Template for static Pages

home.php	Template for the blog posts index (if different from front page)
archive.php	Template for date/author/CPT archive listings
single.php	Template for a single blog post
comments.php	Comment form and comment list markup
search.php	Search results template
attachment.php	Template for viewing a single media attachment
404.php	"Page not found" template
category.php / tag.php	Category-specific / tag-specific archive templates
author.php	Author archive template
date.php	Date-based archive template

The Template Hierarchy

WordPress decides which template file to load for a given URL by walking a fixed priority order — the **Template Hierarchy** — from most specific to most general, always falling back to `index.php` if nothing more specific exists.

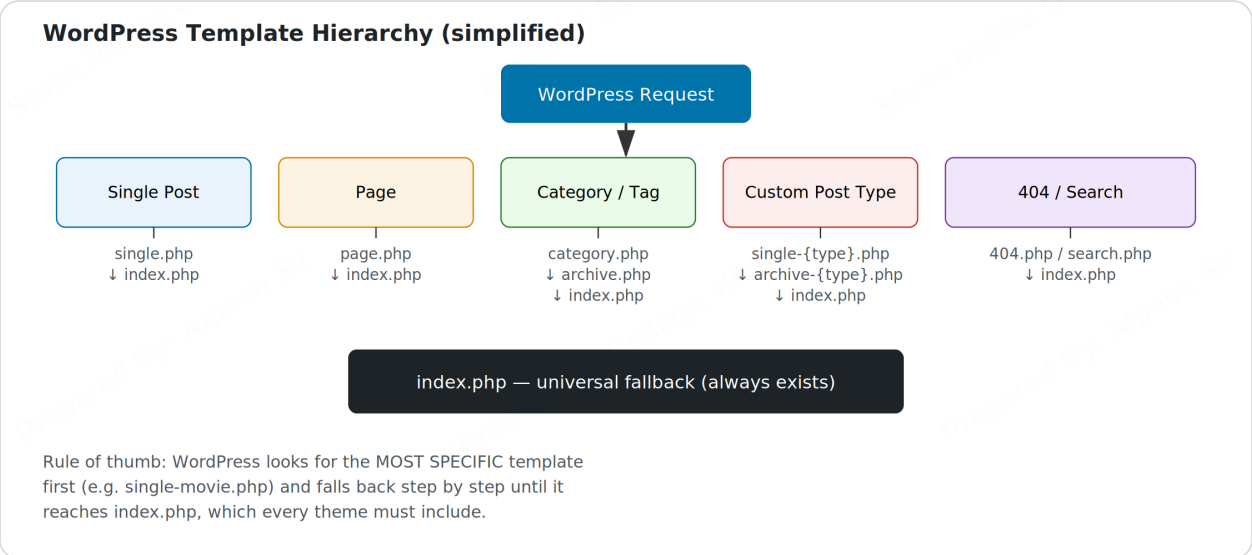


Fig 4.2 — Simplified Template Hierarchy showing fallback order per content type

ગુજરાતીમાં સમજ

Template Hierarchy એટલે શું? જ્યારે કોઈ યુઝર વેબસાઈટ પર કોઈ page ખોલે, ત્યારે WordPress નક્કી કરે છે કે કઈ ટેમ્પલેટ ફાઈલ વાપરવી. તે પહેલા સૌથી ચોક્કસ (specific) ફાઈલ શોધે છે — દા.ત. એક movie પોસ્ટ માટે `single-movie.php` — અને જો એ ના મળે તો સામાન્ય ફાઈલ તરફ જાય છે, છેલ્લે `index.php` સુધી, જે દરેક થીમમાં હોવી જ જોઈએ.

4.2 The Loop

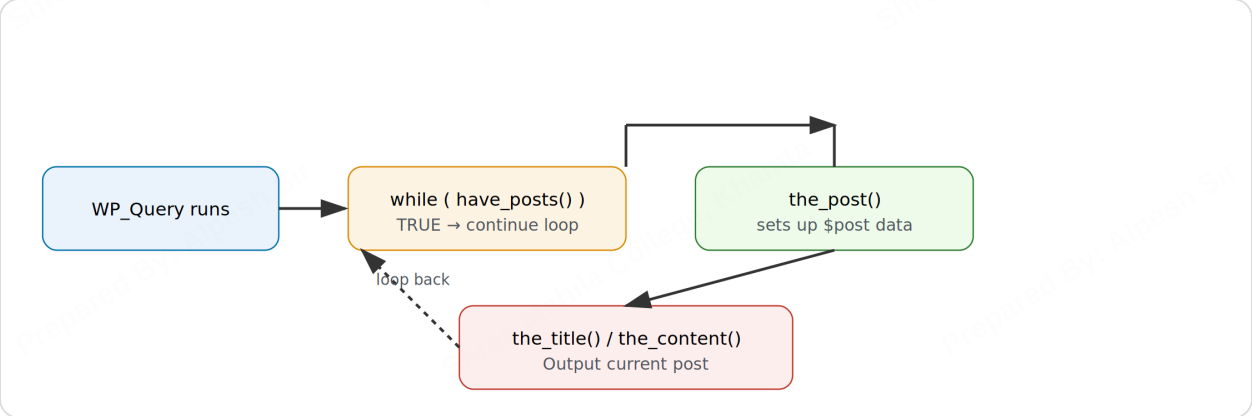


Fig 4.3 — have_posts() / the_post() cycle that outputs each matching post

The **Loop** is the core PHP construct that WordPress uses to display posts. It iterates over the post(s) matched by the current query and, for each one, sets up post data so template tags like `the_title()` work correctly.

```
<?php if ( have_posts() ) : while ( have_posts() ) : the_post(); ?>

    <h2><?php the_title(); ?></h2>
    <div><?php the_content(); ?></div>

<?php endwhile; else : ?>
    <p>No posts found.</p>
<?php endif; ?>
```

Every custom loop with `WP_Query` must end with `wp_reset_postdata();` to restore the main query's global `$post` object.

ગુજરાતીમાં સમજ

The Loop શું છે? The Loop એ PHP નો એવો કોડ છે જે એક પછી એક બધી પોસ્ટ તપાસે છે અને દરેકને સ્ક્રીન પર બતાવે છે. `have_posts()` ચેક કરે છે કે હજુ પોસ્ટ બાકી છે કે નહીં, અને `the_post()` આગલી પોસ્ટનો ડેટા તૈયાર કરે છે — ત્યાર પછી જ `the_title()` અને `the_content()` જેવા ટેગ સાચું આઉટપુટ આપે છે.

સહેલી રીતે યાદ રાખો: "જ્યાં સુધી પોસ્ટ છે ત્યાં સુધી — આગલી પોસ્ટ લો, અને એને છાપો."

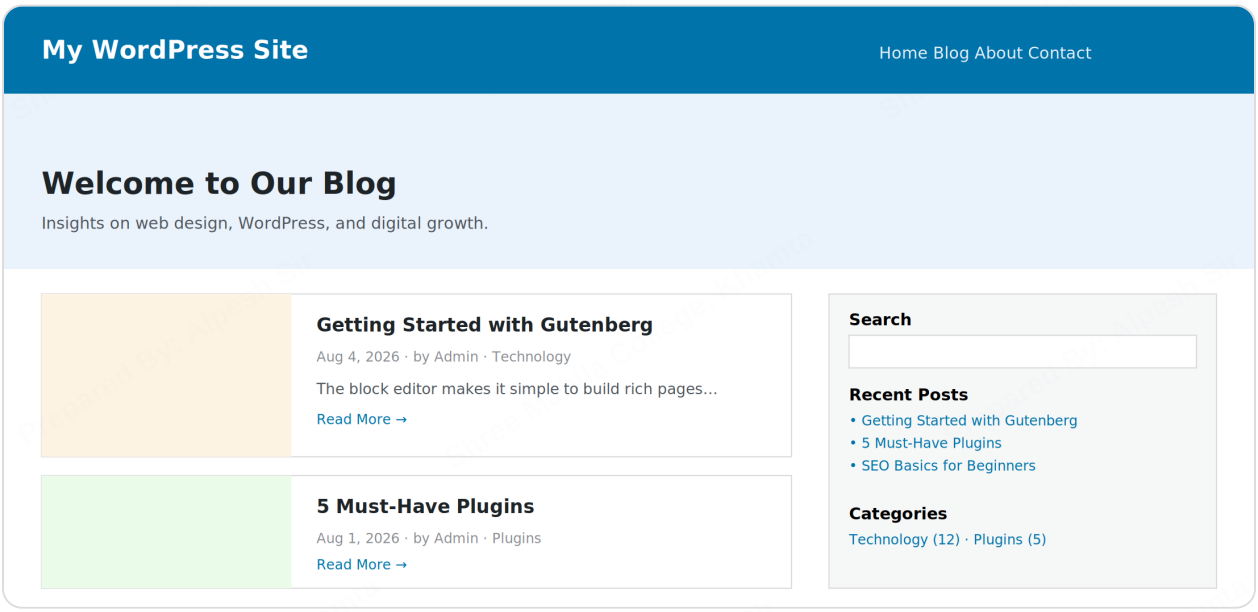


Fig 4.3a — What the Loop's output looks like on the live site: each post block below the header comes from one Loop iteration

4.3 Template Tags

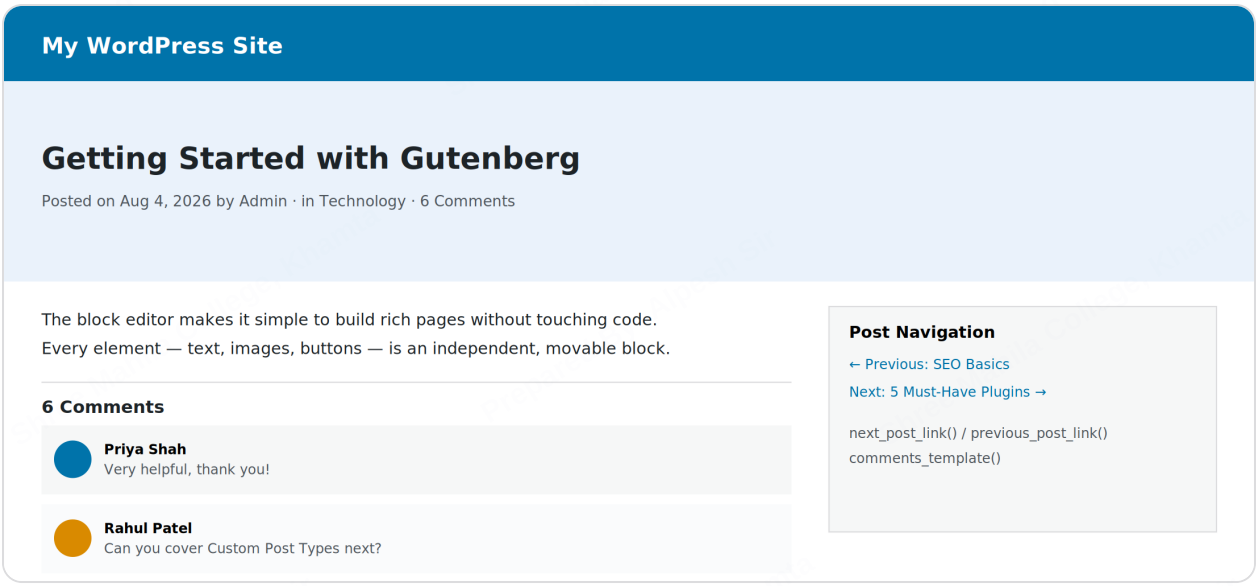


Fig 4.3b — single.php rendered on the front end: title, content, comments, and prev/next post links — all produced by Template Tags

ગુજરાતીમાં સમજ

Template Tag એટલે શું? Template Tag એ PHP નું ફંક્શન છે જે થીમ ફાઈલોમાં વાપરીને ડેટાબેઝમાંથી ડેટા (ટાઈટલ, કન્ટેન્ટ, તારીખ, લેખકનું નામ વગેરે) મેળવીને સ્ક્રીન પર બતાવે છે. દા.ત. `the_title()` પોસ્ટનું ટાઈટલ છાપે છે, `the_content()` આખું લખાણ છાપે છે.

યાદ રાખવાની સહેલી રીત: નામમાં **"the_"** હોય તે સીધું છાપે (echo), અને **"get_the_"** હોય તે વેલ્યુ પાછી આપે (return) — જેને તમે ચલ (variable) માં સંગ્રહી શકો.

General Tags

`wp_head()`, `get_footer()`, `get_header()`, `get_sidebar()`, `get_search_form()`, `bloginfo()`, `wp_title()`, `single_post_title()`, `wp_footer()`, `comments_template()`, `add_theme_support()`, `get_template_directory_uri()`, `body_class()`

These load reusable template parts, output `<head>`/footer hooks required by plugins, print site info (name, URL, description via `bloginfo()`), register theme features (`add_theme_support('post-thumbnails')`), and output dynamic CSS classes on the `<body>` tag.

Author Tags

`the_author()`, `get_the_author()`, `the_author_link()`, `get_the_author_link()`, `the_author_meta()`, `the_author_posts()` — display the post author's name, profile link, custom meta (bio, website) and post count.

Category Tags

`category_description()`, `single_cat_title()`, `the_category()` — output category descriptions, the current category archive title, and the list of categories a post belongs to.

Link Tags

`the_permalink()`, `get_permalink()`, `home_url()`, `get_home_url()`, `site_url()`, `get_site_url()` — return/print canonical URLs for the current post or the site root.

Post Tags

`the_content()`, `the_excerpt()`, `the_ID()`, `the_tags()`, `the_title()`, `get_the_title()`, `the_date()`, `get_the_date()`, `the_time()`, `next_post_link()`, `previous_post_link()`, `posts_nav_link()`, `post_class()` — the core set used inside the Loop to print a post's body, summary, ID, tags, title, date/time, pagination links between posts, and dynamic CSS classes on the post wrapper.

Post Thumbnail Tags

`has_post_thumbnail()`, `get_post_thumbnail_id()`, `the_post_thumbnail()`, `get_the_post_thumbnail()` — check for and output the Featured Image (requires `add_theme_support('post-thumbnails')` in functions.php).

Navigation Menu Tag

`wp_nav_menu( array('theme_location' => 'primary') )` — outputs a menu previously registered with `register_nav_menu()` and assigned by the user under Appearance → Menus.

Conditional Tags

`is_archive()`, `is_category()`, `is_front_page()`, `is_home()`, `is_page()`, `is_single()`, `is_search()`, `is_attachment()`, `is_active_sidebar()` — booleans

used to branch template logic based on the type of page currently being viewed (e.g., show a different header on the front page).

```
<?php if ( is_front_page() ) : ?>
    <div class="hero">Welcome banner</div>
<?php elseif ( is_single() ) : ?>
    <div class="post-meta"><?php the_date(); ?></div>
<?php endif; ?>
```

4.4 functions.php

The **functions.php** file acts like a plugin built into the theme — it runs automatically on every page load and is the standard place to register menus, widget areas, theme support, enqueue styles/scripts, and hook in custom PHP functions (covered further in Unit 5).

```
<?php
function mytheme_setup() {
    add_theme_support( 'post-thumbnails' );
    add_theme_support( 'title-tag' );
    register_nav_menu( 'primary', 'Primary Menu' );
}
add_action( 'after_setup_theme', 'mytheme_setup' );
```

ગુજરાતીમાં સમજ

functions.php શા માટે ખાસ છે? આ ફાઈલ થીમની અંદર રહેલી "mini plugin" જેવી છે — તે દરેક પેજ લોડ થાય ત્યારે આપોઆપ ચાલે છે. તેમાં તમે મેનુ રજિસ્ટર કરી શકો, થીમના ફીચર્સ ચાલુ કરી શકો (જેમ કે Featured Image), અને પોતાના કસ્ટમ ફંકશન બનાવી શકો — આ બધું `wp-content` ની બહાર કંઈ ફેરફાર કર્યા વગર.

Unit 4 — Lab Practicals

1. Build a theme from scratch with style.css, index.php, header.php, footer.php, sidebar.php.
2. Write the Loop in index.php to list all posts with title, excerpt, and date.
3. Create single.php and page.php with distinct layouts using conditional tags.
4. Register a navigation menu in functions.php and output it with wp_nav_menu().
5. Add Featured Image support and display it using the_post_thumbnail() on single.php.
6. Use is_front_page(), is_single(), is_page() to conditionally change layout/ markup.

5 Advanced Development

5.1 Advanced Functions — Hooks & Shortcodes

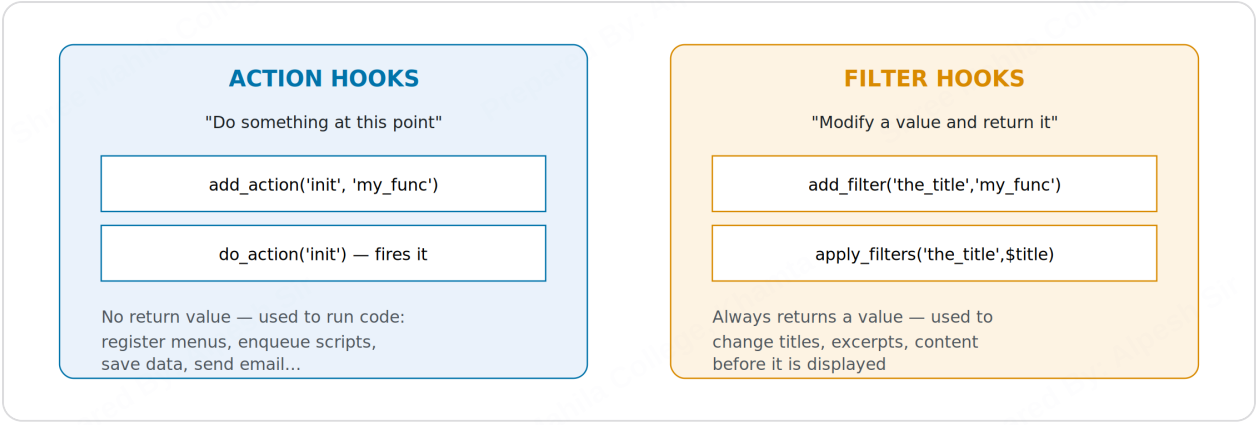


Fig 5.1 — Action hooks run code; filter hooks modify and return data

FUNCTION	PURPOSE
`add_action( $hook, $callback )`	Attaches a custom function to run when WordPress reaches a specific point (e.g. `init`, `wp_enqueue_scripts`)
`add_filter( $hook, $callback )`	Attaches a function that receives a value, modifies it, and must return it (e.g. altering `the_title`)
`add_shortcode( $tag, $callback )`	Registers a reusable `[shortcode]` that editors can insert into post/page content
`do_shortcode( $content )`	Manually parses and executes shortcodes inside a string (useful in template files)
`register_nav_menu( $location, $desc )`	Declares a menu location a theme supports, so it appears under Appearance → Menus

```
// Custom shortcode: [greet name="Alpesh"]
function greet_shortcode( $atts ) {
    $atts = shortcode_atts( array( 'name' => 'Guest' ), $atts );
    return "Hello, " . esc_html( $atts['name'] ) . "!";
```

```
}
add_shortcode( 'greet', 'greet_shortcode' );

// Filter example: force all titles to uppercase
add_filter( 'the_title', function( $title ) {
    return strtoupper( $title );
});
```

ગુજરાતીમાં સમજ

Action Hook vs Filter Hook નો ફરક: Action Hook (`add_action`) એ "કંઈક કરવા" માટે વપરાય છે — જેમ કે ઈમેલ મોકલવો, ડેટા સેવ કરવો. તે કંઈ પાછું (return) નથી આપતું.

Filter Hook (`add_filter`) એ "કોઈ વેલ્યુ બદલવા" માટે વપરાય છે — તે એક વેલ્યુ લે છે, તેમાં ફેરફાર કરે છે, અને ફરજિયાત પાછી (return) આપવી પડે છે. દા.ત. પોસ્ટનું ટાઈટલ મોટા અક્ષરોમાં બતાવવું.

સહેલી રીતે: *Action = "કંઈક કરો", Filter = "કંઈક બદલીને પાછું આપો".*

5.2 Custom Post Types & Taxonomies

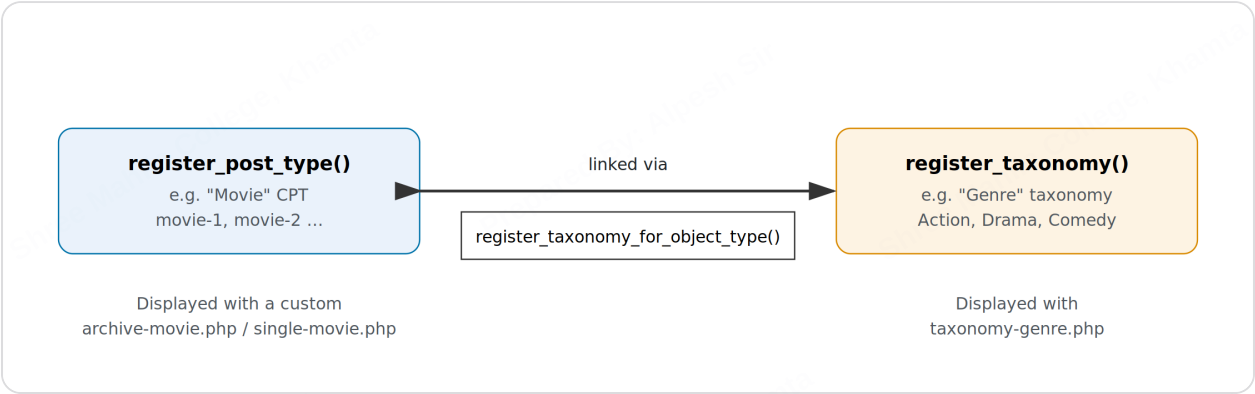


Fig 5.2 — A Custom Post Type linked to a custom Taxonomy

A **Custom Post Type (CPT)** lets you model content beyond posts/pages — e.g., "Movies," "Products," "Events." A **Custom Taxonomy** lets you classify that content — e.g., "Genre" for Movies.

```
// Register a "Movie" Custom Post Type
function register_movie_cpt() {
    register_post_type( 'movie', array(
        'labels' => array( 'name' => 'Movies', 'singular_name' => 'Movie' ),
```

```
        'public' => true,
        'has_archive' => true,
        'supports' => array( 'title', 'editor', 'thumbnail' ),
        'menu_icon' => 'dashicons-video-alt2',
    ));
}
add_action( 'init', 'register_movie_cpt' );

// Register a "Genre" taxonomy for Movies
function register_genre_taxonomy() {
    register_taxonomy( 'genre', 'movie', array(
        'label' => 'Genre',
        'hierarchical' => true,
        'public' => true,
    ));
}
add_action( 'init', 'register_genre_taxonomy' );
```

Displaying a CPT & its taxonomy: create `archive-movie.php` (list view, uses the normal Loop against the CPT), `single-movie.php` (detail view), and `taxonomy-genre.php` (genre archive) — WordPress' Template Hierarchy automatically picks these files up.

> ગુજરાતીમાં સમજ
>
> **Custom Post Type (CPT) અને Taxonomy શું છે?** સામાન્ય રીતે WordPress માં ફક્ત "Post" અને "Page" જ હોય છે. પણ જો તમારે "Movie", "Product" કે "Event" જેવો અલગ પ્રકારનો કન્ટેન્ટ બનાવવો હોય, તો `register_post_type()` વાપરીને પોતાનો Custom Post Type બનાવી શકાય.
>
> Taxonomy એ CPT ને વર્ગીકૃત (categorize) કરવાની રીત છે — જેમ કે "Movie" માટે "Genre" (Action, Comedy, Drama) — આ `register_taxonomy()` વડે બને છે. ટૂંકમાં: **CPT =** નવો કન્ટેન્ટ પ્રકાર**, Taxonomy =** એ કન્ટેન્ટને ગોઠવવાની રીત**.**

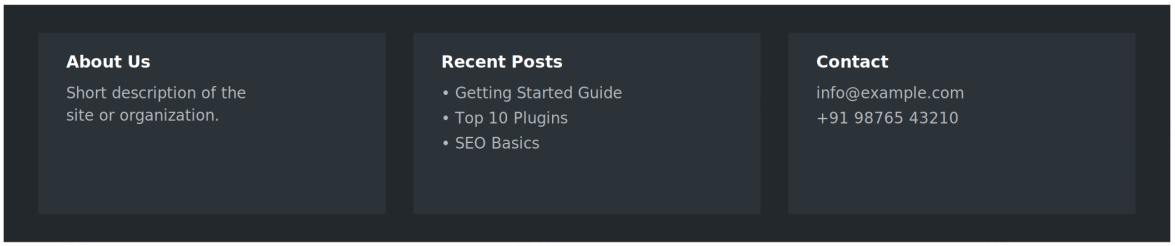
5.3 Widget Areas — register_sidebar() & dynamic_sidebar()

```
// Register a custom widget area in functions.php
function mytheme_widgets_init() {
    register_sidebar( array(
        'name'          => 'Footer Widget Area',
        'id'            => 'footer-widgets',
        'before_widget' => '<div class="widget">',
```

```
        'after_widget'  => '</div>',
        'before_title'  => '<h3>',
        'after_title'   => '</h3>',
    ));
}
add_action( 'widgets_init', 'mytheme_widgets_init' );

// Output the widget area in footer.php
<?php if ( is_active_sidebar( 'footer-widgets' ) ) : ?>
    <?php dynamic_sidebar( 'footer-widgets' ); ?>
<?php endif; ?>
```

dynamic_sidebar('footer-widgets') — Rendered Output



Each box = one widget, wrapped by before_widget/after_widget markup from register_sidebar().

Fig 5.3 — Output of the code above on the front end

ગુજરાતીમાં સમજ

register_sidebar() અને **dynamic_sidebar()** નો ફરક: `register_sidebar()` એ નવો *widget area* બનાવવા માટે વપરાય છે — તે functions.php માં ફક્ત એક જ વાર લખાય છે, જેમ કે "Footer Widget Area" જાહેર કરવું.

`dynamic_sidebar()` એ પેલા area માં યુઝરે જે widgets મૂક્યા હોય એ ખરેખર છાપવા માટે વપરાય છે — તે footer.php કે sidebar.php જેવી ટેમ્પલેટ ફાઈલમાં લખાય છે. ટૂંકમાં: **register = જગ્યા બનાવો, dynamic_sidebar = એ જગ્યાનું કન્ટેન્ટ બતાવો.**

Unit 5 — Lab Practicals

1. Register a custom shortcode and use it inside a page's content.
2. Write a filter that modifies post excerpts (e.g., custom "Read more" text).
3. Create a "Testimonial" Custom Post Type with a "Rating" custom taxonomy.
4. Build archive-testimonial.php and single-testimonial.php templates to display it.
5. Register a new widget area ("Footer Widgets") and display it with dynamic_sidebar().
6. Combine add_action + register_nav_menu to add a secondary "Footer Menu" location.

18 CS-18 · Consolidated Practical / Lab List

CS-18 is the dedicated MDC practical paper covering CS-15 (C++ OOP), CS-16 (RDBMS/Oracle) and CS-17 (WordPress). The WordPress-specific practical checklist below consolidates the unit-wise practicals above into an exam-ready sequence:

1. Install WordPress locally using XAMPP and phpMyAdmin.
2. Create posts/pages using various Gutenberg blocks; manage categories & tags.
3. Manage the Media Library — upload, edit metadata, insert into content.
4. Create and test multiple user roles and their capabilities.
5. Install, activate, and customize a theme (logo, menu, homepage, custom CSS).
6. Add and arrange widgets; observe inactive-widget behaviour on theme switch.
7. Install and configure at least 3 plugins (SEO, contact form, and one more of choice).
8. Build a custom theme from scratch: style.css, header.php, footer.php, sidebar.php, index.php, single.php, page.php.
9. Implement the Loop and at least 6 different template tags across templates.
10. Register a custom navigation menu and a custom widget area via functions.php.
11. Create a custom shortcode and a custom filter.
12. Register a Custom Post Type with a Custom Taxonomy and build its archive/single templates.

Exam Pattern & Model Question Paper

Based on the official Saurashtra University BCA (Honours) evaluation scheme, CS-17 carries **50 CCE (internal) + 50 SEE (semester-end) = 100 total marks**. Internal/unit tests generally follow this structure (25 marks per test, covering 2–3 units at a time):

QUESTION	INSTRUCTION	MARKS
Q.1	Attempt any TWO of the following (Unit 1)	10
Q.2	Attempt any TWO of the following (Unit 2 / relevant unit)	10
Q.3	Attempt any ONE of the following (Unit 3 / relevant unit)	05

The Semester-End Exam (SEE) typically extends this pattern across all 5 units for the full 50 marks, mixing short-answer, descriptive, and practical/coding-based questions (writing template tag code, explaining hooks, designing a CPT, etc.).

> Note: This is a representative structure based on the official syllabus and university exam pattern, built for revision purposes — always cross-check the exact question paper format with your department/current exam circular before the exam.

Important Exam-Oriented Questions & Answers

Short Answer Type (2 marks each — good for Q1/Q2 style)

Q1. What is a CMS? Give one example. 2 marks

A Content Management System lets users create/manage digital content through a dashboard without coding each page. Example: WordPress.

Q2. Differentiate between a Post and a Page. 2 marks

Posts are time-based, chronological, support categories/tags (used for blogs/news). Pages are static, hierarchical, standalone content (About, Contact) without categories/tags.

Q3. What is the difference between a Category and a Tag? 2 marks

Categories are hierarchical (broad topics with parent/child); Tags are flat, non-hierarchical keywords for cross-referencing specific details.

Q4. What is a Widget? 2 marks

A drag-and-drop content block (e.g., Search, Recent Posts) placed into a theme's widget area without coding.

Q5. What is a Plugin? Give two examples. 2 marks

A packaged PHP add-on that extends WordPress functionality without editing core files. Examples: Yoast SEO, Contact Form 7.

Q6. What is the Loop in WordPress? 2 marks

The PHP construct using have_posts() and the_post() that iterates through queried posts and outputs each one using template tags.

Q7. What is the difference between an Action hook and a Filter hook? 2 marks

An action (add_action/do_action) runs custom code at a point in execution and returns nothing; a filter (add_filter/apply_filters) receives a value, modifies it, and must return the modified value.

Q8. What is a Custom Post Type? 2 marks

A user-defined content type (beyond default Posts/Pages) registered with register_post_type(), e.g. "Product" or "Movie," used to model structured content.

Q9. What is the WordPress Template Hierarchy? 2 marks

The priority order WordPress follows to pick the right template file for a given request — from most specific (e.g. single-movie.php) down to the universal fallback index.php.

Q10. What happens to widgets when you switch to a theme with fewer widget areas? 2 marks

They are automatically moved to the "Inactive Widgets" area rather than deleted, so they can be restored later.

Long Answer / Descriptive Type (5 marks each — good for Q3 style)

Q11. Explain the WordPress directory structure in detail. 5 marks

Covers wp-admin (dashboard core), wp-includes (core libraries), wp-content (themes, plugins, uploads — the only folder meant for customization), wp-config.php (DB credentials/security keys), and .htaccess (permalink rewrite rules). See Fig 1.1.

Q12. Describe the anatomy of a WordPress theme along with its required and optional template files. 5 marks

A theme minimally needs style.css (metadata + CSS) and index.php (fallback template). Optional but common files include header.php, footer.php, sidebar.php, single.php, page.php, archive.php, search.php, comments.php, 404.php, and functions.php, all working together via the Template Hierarchy. See Fig 4.1 & Fig 4.2.

Q13. Explain the Loop with a code example. 5 marks

The Loop wraps a while(have_posts()) block calling the_post() each iteration to set up global post data, followed by template tags like the_title() and the_content() to output content. See section 4.2 code sample and Fig 4.3.

Q14. Explain Template Tags with categories and examples. 5 marks

Template tags are PHP functions used inside theme files to output dynamic content: General (get_header(), bloginfo()), Author (the_author()), Category (the_category()), Link (the_permalink()), Post (the_title(), the_content()), Post Thumbnail (the_post_thumbnail()), Navigation Menu (wp_nav_menu()), and Conditional tags (is_single(), is_page()).

Q15. How do you register and display a Custom Post Type with a Custom Taxonomy? Explain with code. 5 marks

Use register_post_type() hooked to 'init' to define the CPT (labels, supports, has_archive), and register_taxonomy() to define a classification for it. Display it using archive-{posttype}.php / single-{posttype}.php / taxonomy-{taxonomy}.php, which the Template Hierarchy automatically loads. See section 5.2 code sample and Fig 5.2.

Q16. Explain how to create and register a Widget Area, and how to display it in a theme. 5 marks

Call register_sidebar() inside a function hooked to 'widgets_init' in functions.php, specifying id, before/after widget and title wrappers. Then call dynamic_sidebar('id') inside the template file (commonly footer.php or sidebar.php), typically wrapped in is_active_sidebar() to avoid empty markup. See section 5.3 and Fig 5.3.

Q17. Explain the WordPress database structure with major tables. 5 marks

wp_posts stores posts/pages/CPTs; wp_postmeta stores custom field data; wp_users/wp_usermeta store accounts and their meta; wp_terms, wp_term_taxonomy, wp_term_relationships together implement categories/tags/ custom taxonomies; wp_comments/wp_commentmeta store comments; wp_options stores site-wide settings. See Fig 1.8.

Practical / Coding-Based

Q18. Write PHP code to register a custom shortcode that displays a welcome message.

See the add_shortcode() example in section 5.1 — register with add_shortcode('greet', 'greet_shortcode') and return the output string from the callback (never echo directly).

Q19. Write code to register a navigation menu and display it in header.php.

register_nav_menu('primary', 'Primary Menu') hooked to after_setup_theme, then wp_nav_menu(array('theme_location' => 'primary')) inside header.php.

Q20. Write a conditional-tag example that shows a hero banner only on the homepage.

if (is_front_page()) { echo ' ... '; } — see section 4.3 conditional tag example.

Quick-Revision Flashcards

CMS
Software to create/manage digital content without coding each page.

Theme
Controls presentation/layout of stored content.

Plugin
PHP add-on that extends functionality without editing core.

Widget
Drag-and-drop content block placed in a widget area.

The Loop
have_posts() + the_post() cycle that outputs queried posts.

Template Tag
PHP function used in theme files to output dynamic content.

Conditional Tag
Boolean function (is_single(), is_page()) to branch template logic.

Action Hook
add_action() — runs custom code at a specific point; no return value.

Filter Hook
add_filter() — modifies a value and must return it.

Shortcode
[tag] inserted in content, parsed by add_shortcode() callback.

Custom Post Type
register_post_type() — models content beyond posts/pages.

Custom Taxonomy
register_taxonomy() — classification scheme for a post type.

Template Hierarchy
Priority order WP uses to pick a template file, ending at index.php.

functions.php
Theme's built-in "plugin" file — hooks, menus, theme support.

wp_nav_menu()
Outputs a registered + user-assigned navigation menu.

register_sidebar()
Declares a new widget area in functions.php.

dynamic_sidebar()
Outputs the widgets placed in a registered widget area.

Permalink
URL structure for content; "Post name" is SEO-recommended.

Full Syllabus Summary

Unit 1 — Introduction, Installation & Configuration

CMS/WordPress fundamentals, features, pros/cons, local & live installation, directory structure, dashboard, content (posts/pages/media), Gutenberg blocks, user roles, core settings, updates, database structure.

Unit 2 — Theme

What a theme is, install/activate flow, and all five Customizer panels (Site Identity, Menus, Widgets, Homepage Settings, Additional CSS).

Unit 3 — Widget & Plugin

Widgets/widget areas, widget management, inactive widgets; what a plugin is, install/activate flow, and commonly used plugins (SEO, forms, e-commerce, caching, ACF).

Unit 4 — Theme Development

Theme anatomy, all template files, the Template Hierarchy, the Loop, and the full set of template tags (General, Author, Category, Link, Post, Post Thumbnail, Navigation Menu, Conditional) plus functions.php.

Unit 5 — Advanced Development

Hooks (add_action/add_filter), shortcodes (add_shortcode/do_shortcode), register_nav_menu(), Custom Post Types + Taxonomies (register_post_type/ register_taxonomy), and custom widget areas (register_sidebar/dynamic_sidebar).

> **Exam strategy:** Prioritize the Loop, Template Tags, Hooks (actions vs. filters), and Custom Post Type registration code — these appear most frequently as both theory and coding questions.

References

- Sarah McHarry — *WordPress to Go: How to Build a WordPress Website on Your Own Domain, from Scratch, Even If You Are a Complete Beginner*
- Official WordPress Developer Resources — developer.wordpress.org (Theme Handbook, Plugin Handbook, Template Tag reference)
- WordPress Codex & wordpress.org documentation
- Saurashtra University B.C.A. (Honours) & B.C.A. (Honours with Research) Syllabus, Semester 3 & 4, effective June 2024

CS-17: Content Management System using WordPress — Complete Study Guide & Exam Prep

Prepared By: Alpesh Sir (Shree Mahila College, Khamta)