

Website Generator App - Design, Structure, and Tags

OVERVIEW

Background

The existing website generator application uses a single JSON file structure where content, layout, and design elements are tightly coupled. This approach creates several challenges:

- Business users can break pages by making syntax errors in layout configuration
- Content cannot be reused across different page layouts
- Design changes require editing content files, risking data corruption
- No clear separation of concerns between what business users edit vs. what developers control
- Difficult to maintain consistency across pages with similar layouts
- No validation to prevent invalid configurations

The current system evolved organically without a formal architecture, resulting in technical debt and maintenance challenges that limit scalability.

Purpose

The WebGen Phase 1 project establishes a clean separation between form and function by refactoring the JSON architecture into distinct layers:

Configuration Layer

Site-wide settings, layout templates, navigation patterns (admin-managed)

Content Layer

Pure content blocks with no layout information (business user-managed)

Mapping Layer

Page configurations that connect content to layouts (admin-managed)

Navigation Layer

Menu structure and hub/spoke relationships (admin-managed)

This separation ensures business users can safely edit content without risk of breaking the website, while administrators maintain full control

SCOPE

In Scope

- ✓ Define 8 standardized JSON file types with clear responsibilities
- ✓ Create reusable layout templates and page patterns
- ✓ Establish hub/spoke directory structure (hub00-hub99)
- ✓ Separate content files (.content.json) from page configurations (.page.json)
- ✓ Implement site-wide configuration system (_site.json)
- ✓ Define secondary navigation patterns for hub/spoke pages
- ✓ Create validation rules and error handling
- ✓ Document migration strategy from legacy format
- ✓ Maintain backward compatibility during transition

Out of Scope

- ✗ Visual content editor UI (Phase 2)
- ✗ Multi-language content management system (Phase 2)
- ✗ Version control and content approval workflows (Phase 2)
- ✗ Automated testing suite (Phase 2)
- ✗ Performance optimization (Phase 3)
- ✗ API for external integrations (Phase 3)

For Administrators (Developers/Designers)

- 🎯 Centralized control - Change all page layouts from one file
- 🎯 Reusable templates - Define once, use everywhere
- 🎯 Easy maintenance - Fix layout bugs in one place
- 🎯 Clear ownership - Know exactly who should edit what
- 🎯 Better validation - Catch errors before deployment

For the Organization:

- 🎯 Reduced errors - Fewer broken pages in production
- 🎯 Faster time-to-market - Business users can update content independently
- 🎯 Lower training costs - Business users only learn content structure
- 🎯 Scalability - Easy to add new pages and layouts
- 🎯 Consistency - All pages follow standardized patterns

WEBSITE ARCHITECTURE

WHO EDITS WHAT?

Layers / Files	File Type	Business User	Admin
Layer 1: Website Definition (_site.json)	_site.json	✗	✓
↓ defines structure, paths, branding	_layouts.json	✗	✓
	_page-templates.json	✗	✓
	_header-layouts.json	✗	✓
Layer 2: Design Components	_navigation-layouts.json	✗	✓
├── _header-layouts.json (page headers)	_menu.json	✗	✓
├── _navigation-layouts.json (secondary nav patterns)	*.content.json	✓	●
├── _layouts.json (section layouts)	*.page.json	✗	✓
└── _page-templates.json (page patterns with nav)			
Layer 3: Content & Configuration			
├── *.content.json (pure content)			
└── *.page.json (layout + content + nav mapping)			
Layer 4: Navigation			
└── _menu.json (primary + secondary nav config)			

WEBSITE FILE STRUCTURE

Files	Directory	Migration Status
Renderer files	./scripts/	pending
CSS files	./styles/	pending
Meta/config files	./framework/	already refactored
Media files (images, etc.)	./assets/	no change
Content and page configs	./data/en/	in progress
"Site level" hub and spoke content	./data/en/hub00/	in progress
"Markets" hub and spoke content	./data/en/hub01/	already refactored
"Features" hub and spoke content	./data/en/hub02/	pending

Website Generator App - Design, Structure, and Tags

1. WEBSITE CONFIGURATION (_site.json)

- **Required**: = Required, = Optional

Tag	Definition	Values	Required	Managed By
siteId	Unique website identifier	String (alphanumeric)		
version	Configuration version	"2.0"		
type	Website category	"business", "family", "community"		
tone	Communication style	"professional", "casual", "fun"		
theme.mode	Color scheme	"light", "dark", "seasonal"		
theme.primaryColor	Brand primary color	Hex color (#d90429)		
theme.secondaryColor	Brand secondary color	Hex color (#a10000)		
theme.accentColor	Accent/hover color	Hex color (#ff0000)		
theme.fontFamily	Typography	Font stack string		
branding.siteUrl	Website URL	URL string		
branding.siteName	Website name	String		
branding.tagLine	Site tagline	String		
branding.logoPath	Logo base URL	URL string		
directories.hubPattern	Hub naming pattern	"hub{00-99}"		
directories.maxHubs	Maximum hubs	Integer (100)		
directories.paths.styles	Styles root directory	Path string		
directories.paths.meta	Framework root directory	Path string		
directories.paths.data	Data root directory	Path string		
directories.paths.media	Media base URL	URL string		
languages.default	Default language	"en", "es", "fr"		
languages.supported	Supported languages	Array of language codes		
contact.companyName	Company legal name	String		
contact.address	Physical address	String		
contact.phone	Contact phone	String		
contact.email	Contact email	Email string		
footer.copyrightYear	Copyright year	Integer (2025)		
footer.copyrightHolder	Copyright owner	String		

2. LAYOUT TEMPLATES (_layouts.json)

Tag	Definition	Values	Required	Managed By
siteId	Website identifier	String (matches _site.json)	✓	👤
version	Configuration version	"2.0"	✓	👤
lastUpdated	Last modification date	ISO date string (YYYY-MM-DD)	✓	👤
layouts.{id}	Unique layout identifier	"alternating-image", "hero-banner", "feature-grid", etc.	✓	👤
layouts.{id}.name	Human-readable name	String	✓	👤
layouts.{id}.description	Layout purpose	String	✓	👤
layouts.{id}.defaultSettings	Default configuration	Object	✓	👤
layouts.{id}.defaultSettings.split	Width split ratio	"30-70", "40-60", "50-50", "60-40", "70-30"	∅	👤
layouts.{id}.defaultSettings.imageSide	Image position	"left", "right"	∅	👤
layouts.{id}.defaultSettings.columns	Column count	1, 2, 3, 4	∅	👤
layouts.{id}.supportedSplits	Valid split values	Array of split strings	∅	👤
layouts.{id}.supportedSides	Valid side values	["left", "right"]	∅	👤
layouts.{id}.contentSlots	Required content fields	Object defining slots	✓	👤
contentSlots.{slot}.required	Is field required	true, false	✓	👤
contentSlots.{slot}.type	Field data type	"text", "richtext", "array", "media", "link"	✓	👤

3. PAGE TEMPLATES (_page-templates.json)

Tag	Definition	Values	Required	Managed By
siteId	Website identifier	String (matches _site.json)	✓	👤
version	Configuration version	"2.0"	✓	👤
lastUpdated	Last modification date	ISO date string (YYYY-MM-DD)	✓	👤
pageTemplates.{id}	Template identifier	"alternating-sections-left-start", "hub-with-tab-menu", etc.	✓	👤
pageTemplates.{id}.name	Template name	String	✓	👤
pageTemplates.{id}.description	Template purpose	String	✓	👤
pageTemplates.{id}.headerLayout	Header style	Reference to _header-layouts.json ID	✓	👤
pageTemplates.{id}.secondaryNav	Navigation pattern	Reference to _navigation-layouts.json ID	✓	👤
pageTemplates.{id}.contentLayout.pattern	Content pattern	"repeating-alternating", "dynamic-sections", "static"	✓	👤
pageTemplates.{id}.contentLayout.firstSide	Starting image side	"left", "right"	∅	👤
pageTemplates.{id}.contentLayout.defaultSplit	Default width split	"30-70", "50-50", "70-30"	∅	👤

4. HEADER LAYOUTS (_header-layou

Tag	Definition	Values	Required	Managed By
siteId	Website identifier	String (matches _site.json)	✓	👤
version	Configuration version	"2.0"	✓	👤
lastUpdated	Last modification date	ISO date string (YYYY-MM-DD)	✓	👤
headerLayouts.{id}	Header style identifier	"standard", "minimal", "breadcrumb-header", etc.	✓	👤
headerLayouts.{id}.name	Header name	String	✓	👤
headerLayouts.{id}.description	Header purpose	String	✓	👤
headerLayouts.{id}.alignment	Text alignment	"center", "left", "right", "split"	✓	👤
headerLayouts.{id}.components	Displayed elements	Array: ["title", "subtitle", "breadcrumb", "divider", "actions"]	✓	👤
headerLayouts.{id}.titleSize	Title font size	CSS size string ("42px", "2em")	✓	👤
headerLayouts.{id}.subtitleStyle	Subtitle styling	"uppercase-red", "normal", "italic"	∅	👤
headerLayouts.{id}.dividerStyle	Divider line style	"full-width", "centered", "none"	∅	👤
headerLayouts.{id}.integrated	Integrated with hero	true, false	∅	👤

5. NAVIGATION LAYOUTS (_navigation-layouts.json)

Tag	Definition	Values	Required	Managed By
siteld	Website identifier	String (matches _site.json)	✓	👤
version	Configuration version	"2.0"	✓	👤
lastUpdated	Last modification date	ISO date string (YYYY-MM-DD)	✓	👤
navigationLayouts.{id}	Nav pattern identifier	"tab-bar-sibling", "sidebar-left", "none", etc.	✓	👤
navigationLayouts.{id}.name	Navigation name	String	✓	👤
navigationLayouts.{id}.type	Navigation type	"secondary", "none"	✓	👤
navigationLayouts.{id}.style	Display style	"horizontal-tabs", "vertical-sidebar", "breadcrumb"	✓	👤
navigationLayouts.{id}.position	Screen position	"below-header", "left", "right", "above-header"	✓	👤
navigationLayouts.{id}.sticky	Sticky scroll behavior	true, false	▢	👤
navigationLayouts.{id}.behavior.showHubLink	Display hub link	true, false	▢	👤
navigationLayouts.{id}.behavior.showSiblings	Display sibling links	true, false	▢	👤
navigationLayouts.{id}.behavior.highlightActive	Highlight current page	true, false	▢	👤
navigationLayouts.{id}.behavior.centerAlign	Center navigation	true, false	▢	👤
navigationLayouts.{id}.visibility.hubPage	Show on hub pages	true, false	✓	👤
navigationLayouts.{id}.visibility.spokePage	Show on spoke pages	true, false	✓	👤
navigationLayouts.{id}.width	Sidebar width	CSS size string ("250px")	▢	👤
navigationLayouts.{id}.cssClasses	CSS class names	Array of strings	▢	👤

6. MENU CONFIGURATION (_menu.json)

Tag	Definition	Values	Required	Managed By
siteld	Website identifier	String (matches _site.json)	✓	👤
version	Configuration version	"2.0"	✓	👤
lastUpdated	Last modification date	ISO date string (YYYY-MM-DD)	✓	👤
menuItems.{id}	Menu item identifier	"markets", "features", "services", etc.	✓	👤
menuItems.{id}.label	Display name	String	✓	👤
menuItems.{id}.type	Page type	"hub", "spoke"	✓	👤
menuItems.{id}.hub	Parent hub ID	"hub00", "hub01", "hub02", etc.	✓	👤
menuItems.{id}.dataFile	Content file path	"hub01/markets.json"	✓	👤
menuItems.{id}.order	Display order in menu	Integer	▢	👤
menuItems.{id}.secondaryNav.layout	Nav layout ID	Reference to _navigation-layouts.json	✓	👤
menuItems.{id}.secondaryNav.showOnHub	Show on hub home	true, false	▢	👤
menuItems.{id}.secondaryNav.showOnSpokes	Show on spoke pages	true, false	▢	👤
menuItems.{id}.defaultSection	Default hub section	String (section ID)	▢	👤
menuItems.{id}.children	Child pages (spokes)	Array of page objects	▢	👤
children[].id	Child page ID	String	✓	👤
children[].label	Child display name	String	✓	👤
children[].dataFile	Child content file	"hub01/mlm.json"	✓	👤

7. CONTENT FILE (*.content.json)

Tag	Definition	Values	Required	Managed By
siteId	Website identifier	String (matches _site.json)	✓	👤
contentId	Unique content identifier	String matching filename	✓	👤
language	Content language	"en", "es", "fr"	✓	👤
hub	Hub directory	"hub00", "hub01", "hub02"	✓	👤
lastUpdated	Last edit date	ISO date string (YYYY-MM-DD)	✓	👤
author	Content author	String	∅	👤
blocks	Content blocks container	Object	✓	👤
blocks.{blockId}	Individual content block	Unique block identifier	✓	👤
blocks.{blockId}.id	Block identifier	String (matches key)	✓	👤
blocks.{blockId}.source	Source of content block	"file", "database" (Defaults to "file" if omitted.)	∅	👤
blocks.{blockId}.title	Section title	String	✓	👤
blocks.{blockId}.paragraph	Main text content	String (plain text or markdown)	∅	👤
blocks.{blockId}.listItems	Bullet point list	Array of strings	∅	👤
blocks.{blockId}.media	Media object	Object	∅	👤
blocks.{blockId}.media.type	Media type	"image", "carousel", "video"	✓	👤
blocks.{blockId}.media.src	Image filename	"165507.jpg"	✓	👤
blocks.{blockId}.media.alt	Alt text	String	✓	👤
blocks.{blockId}.callToAction	Link object	Object	∅	👤
blocks.{blockId}.callToAction.text	Link text	"READ MORE", "Learn More"	✓	👤
blocks.{blockId}.callToAction.targetId	Link destination	Page ID ("mlm", "features")	✓	👤

8. PAGE CONFIGURATION (*.page.json)

Tag	Definition	Values	Required	Managed By
siteld	Website identifier	String (matches _site.json)	✓	👤
pageld	Unique page identifier	String matching filename	✓	👤
pageType	Page classification	"hub", "spoke"	✓	👤
language	Page language	"en", "es", "fr"	✓	👤
hub	Hub directory	"hub00", "hub01", "hub02"	✓	👤
pageTemplate	Page template ID	Reference to _page-templates.json	✓	👤
headerLayout	Header layout ID	Reference to _header-layouts.json	✓	👤
secondaryNav	Navigation layout ID	Reference to _navigation-layouts.json	✓	👤
contentSource	Content file reference	Content file ID (without extension)	✓	👤
metadata	Page metadata	Object	✓	👤
metadata.title	Page title	String	✓	👤
metadata.subtitle	Page subtitle	String	∅	👤
metadata.description	SEO description	String	∅	👤
metadata.keywords	SEO keywords	Array of strings	∅	👤
contentBlocks	Ordered content list (simple term)	Array of block IDs from content file	∅*	👤
sections	Section definitions (manual control)	Array of objects	∅*	👤
sections[].sectionId	Section identifier	String	✓	👤
sections[].layoutTemplate	Section layout	Reference to _layouts.json	✓	👤
sections[].layoutSettings	Layout overrides	Object	∅	👤
sections[].layoutSettings.split	Width split override	"30-70", "50-50", "70-30"	∅	👤
sections[].layoutSettings.imageSide	Image position override	"left", "right"	∅	👤
sections[].contentMapping	Content-to-layout map	Object	✓	👤
sections[].contentMapping.contentBlock	Block ID reference	String from content file	✓	👤
sections[].contentMapping.slots	Slot assignments	Object mapping slots to content	✓	👤

Note: Use contentBlocks OR sections, not both. Required field depends on pageTemplate.contentLayout.pattern:

- Use contentBlocks when pattern = "repeating-alternating" (auto-generates sections)
- Use sections when pattern = "dynamic-sections" (manual section control)

9. COMMON PATTERNS BY FILE TYPE

Patten Name	Content File	Page File
Simple Spoke Page (e.g., Services)	Pure content blocks	<pre>pageTemplate: "alternating-sections-right-start" headerLayout: "standard" secondaryNav: "none" contentBlocks: ["block1", "block2", "block3"]</pre>
Hub Page (e.g., Markets)	Overview content blocks	<pre>pageTemplate: "alternating-sections-left-start" headerLayout: "standard" secondaryNav: "none" contentBlocks: ["mlm-overview", "partyplan-overview", "b2b-overview"]</pre>
Spoke with Sibling Nav (e.g., MLM)	Detail content blocks	<pre>pageTemplate: "alternating-sections-right-start" headerLayout: "standard" secondaryNav: "tab-bar-sibling" contentBlocks: ["mlm-detail-1", "mlm-detail-2"]</pre>
Hub with Tab Menu (e.g., Features)	Multiple section groups	<pre>pageTemplate: "hub-with-tab-menu" headerLayout: "standard" secondaryNav: "tab-bar-hub-sections" sections: { "home": [...], "frontend": [...], "backoffice": [...] }</pre>

10. VALIDATION RULES

Check	Rule	Error Message
Site consistency	siteld matches across all files	"Site ID mismatch - file belongs to different project"
File naming	{pageld}.content.json matches contentId	"Content ID must match filename"
File naming	{pageld}.page.json matches pageld	"Page ID must match filename"
Hub reference	hub value exists in _site.json definitions	"Invalid hub reference"
Template reference	pageTemplate exists in _page-templates.json	"Template not found"
Layout reference	layoutTemplate exists in _layouts.json	"Layout not found"
Content reference	All contentBlocks exist in content file	"Content block not found"
Required slots	All required slots defined in layout are mapped	"Missing required content slot"
Image side	imageSide is "left" or "right"	"Invalid imageSide value"
Split ratio	split matches pattern \d+-\d+ and sums to 100	"Invalid split ratio"

11. DEPRECATED ASSETS (Phase 1 Migration)

Tags/Patterns No Longer Used

Old Tag/Pattern	Used In	Replaced By	Migration Notes
commonData	Old JSON structure	Content blocks in *.content.json	Extracted to separate content files; shared content now managed through content block
imageSide: "image-left"	Old section definitions	imageSide: "left"	Simplified to "left"/"right" only for consistency
imageSide: "image-right"	Old section definitions	imageSide: "right"	Simplified to "left"/"right" only for consistency
Embedded media base paths	Old JSON files	_site.json → directories.paths.media	Centralized in site configuration for easier updates
pageType: "spoke" without hub	Old menu structure	hub: "hub00" required	All pages must explicitly specify hub directory
Inline carousel configurations	Old sections	Standardized media object	All media now uses consistent structure with type, src, alt
Mixed content/layout JSON	Old single-file structure	Separated .content.json + .page.json	Clean separation of concerns achieved
tagline (lowercase)	_site.json	tagLine (camelCase)	Standardized casing for consistency

Files No Longer Used

Old File	Purpose	Replaced By	Status
markets.json (single file)	Combined content + layout	markets.content.json + markets.page.json	Keep for Phase 1 compatibility
services.json (single file)	Combined content + layout	services.content.json + services.page.json	Keep for Phase 1 compatibility
features2.json	Test/alternate features page	features.content.json + features.page.json	Can be deleted after migration
Old _menu.json format	Navigation without hub references	New _menu.json with hub structure	Deprecated - do not use

Structural Changes

Old Pattern	New Pattern	Reason
Single JSON per page	Two files: .content.json + .page.json	Separation of content from layout
Layout embedded in content	Layout defined in page templates	Reusability and consistency
No site-level configuration	_site.json with global settings	Centralized configuration
Ad-hoc section layouts	Defined in _layouts.json	Standardized components
Inline navigation config	_navigation-layouts.json patterns	Reusable navigation templates