



CQIP Static

Build Tutorial

A complete guide to building your first CQIP Static site

thecontentframework.com

Version 2.1.0 | H D Fraser MSc

CQIP (Content Quality and Intelligence Policy) is a methodology standard, not a technology stack. Implementations are available for static PHP sites, WordPress, page generators, and other document management systems. The platform is the implementation detail — the standard and the eight canonical block types are consistent across all implementations.



What you will build

This tutorial takes you through building a complete three-page website from scratch using the CQIP Static framework. The site belongs to Clearwater Security — a hypothetical access control installer based in Edinburgh.

By the end you will have:

- A home page with a hero section, content blocks, a stats strip, and a call to action
- A service detail page with technical specifications and an FAQ accordion
- An about page with compliance certifications and client reviews
- A contact page
- Correct schema.org JSON-LD on every page, validated against the Google Rich Results Test

This tutorial uses example content for a hypothetical business. Replace all values shown with your own before publishing.

What you need

- The CQIP Static files unzipped and ready to upload (included in this bundle as /cqip/)
- A web server running PHP 7.4 or higher — standard shared hosting is fine
- FTP access or a file manager to upload files
- A text editor — any editor works

You do not need a database, a build tool, composer, Node.js, or any other dependency.

1. Understanding the file structure

Before uploading anything, it helps to understand how a CQIP Static site is organised. Every page is a PHP file. The framework lives in a `cqip/` folder that sits alongside your page files.

The framework folder

The `cqip/` folder contains the engine. You edit one file inside it (`cqip-config.php`) and leave the rest alone.

<code>your-site/</code>	
<code>cqip/</code>	The CQIP Static framework
<code>cqip-config.php</code>	EDIT THIS – your site settings
<code>cqip-functions.php</code>	Do not edit
<code>cqip-schema.php</code>	Do not edit
<code>cqip-meta.php</code>	Do not edit
<code>blocks/</code>	Eight block renderers – do not edit
<code>answer-first.php</code>	
<code>capabilities.php</code>	
<code>solutions.php</code>	
<code>technical.php</code>	
<code>compliance.php</code>	
<code>faq.php</code>	
<code>entity-review.php</code>	
<code>content.php</code>	
<code>templates/</code>	Page templates – do not edit
<code>base.php</code>	Shared header/footer wrapper
<code>service-page.php</code>	Service pages
<code>product-page.php</code>	Product pages
<code>article-page.php</code>	Articles and guides
<code>organization-page.php</code>	About, Contact, Organization pages
<code>assets/</code>	
<code>css/</code>	
<code>cqip-blocks.css</code>	Block styles – do not edit
<code>site.css</code>	YOUR site styles – edit freely
<code>js/</code>	
<code>cqip-interactive.js</code>	FAQ accordion – do not edit
<code>images/</code>	
<code>logo.png</code>	REPLACE with your own logo

Your page files

Every page on your site is a PHP file. Each one sits at a URL-friendly path. An `index.php` file at a path becomes the page at that URL.

your-site/	
index.php	yourdomain.com/
services/	
access-control/	
index.php	yourdomain.com/services/access-control/
about/	
index.php	yourdomain.com/about/
contact/	
index.php	yourdomain.com/contact/
cqip/	The framework (shared by all pages)

Each index.php requires the cqip/cqip-config.php file at its start. The path to cqip-config.php changes depending on how deep the page is. A page at /services/access-control/ uses: `require_once __DIR__ . '/../../cqip/cqip-config.php';`

The anatomy of a page file

Every page follows this pattern. Understanding this pattern means you can build any page.

```
<?php
// 1. Load the framework
require_once __DIR__ . '/cqip/cqip-config.php';

// 2. Define FAQ items (if the page has FAQs)
$faqItems = [ ... ];

// 3. Define the content blocks
$blocks = [ ... ];

// 4. Build the schema
$schemaJson = cqip_build_schema(blocks: $blocks, ...);

// 5. Set page meta (title, description, canonical URL)
$meta = [ ... ];

// 6. Capture your page HTML
ob_start();
?>
<section>...your HTML...</section>
<?php
$bodyContent = ob_get_clean();

// 7. Require the base template – outputs the full page
require_once __DIR__ . '/cqip/templates/base.php';
```





2. Configuring your site

Open `cqip/cqip-config.php` in your text editor. This is the only file you need to edit in the `cqip/` folder. Everything else is driven by these settings.

Finding the Organisation Identity section

Scroll down past the comments at the top. You will see a section labelled **ORGANISATION IDENTITY**. These are the values to change.

Setting your values

Replace the placeholder values with your own. For the tutorial, use these Clearwater Security values:

```
define('CQIP_ORG_NAME',      'Clearwater Security');
define('CQIP_ORG_LEGAL',     'Clearwater Security Ltd');
define('CQIP_ORG_TYPE',      'LocalBusiness');
define('CQIP_SITE_URL',      'https://clearwatersecurity.co.uk');
define('CQIP_ORG_LOGO',      CQIP_SITE_URL . '/cqip/assets/images/logo.png');
define('CQIP_ORG_EMAIL',     'info@clearwatersecurity.co.uk');
define('CQIP_ORG_TEL',       '+441316200000');
define('CQIP_ORG_LOCALITY',   'Edinburgh');
define('CQIP_ORG_REGION',     'Scotland');
define('CQIP_ORG_COUNTRY',    'GB');
```

Important: do not move the `require_once` lines at the bottom of `cqip-config.php` above the `define()` lines. The framework files need the constants to already be set when they load. If you see a 500 error, this is the most likely cause.

Constant	What it does
CQIP_ORG_NAME	Appears in schema, page titles, and header/footer
CQIP_ORG_LEGAL	Legal entity name used in schema <code>legalName</code> field
CQIP_ORG_TYPE	Schema.org organisation type — <code>LocalBusiness</code> for most businesses
CQIP_SITE_URL	Your full domain — no trailing slash. Used in all canonical URLs and schema <code>@id</code> values
CQIP_ORG_LOGO	Absolute URL to your logo. Used in schema and Open Graph image
CQIP_ORG_EMAIL	Contact email — appears in Organization schema
CQIP_ORG_COUNTRY	ISO 3166-1 alpha-2 code. GB for United Kingdom, US for United States

Enable debug mode during development

While building, uncomment this line near the top of `cqip-config.php`:

```
// define('CQIP_DEBUG', true);
```

Change it to:

```
define('CQIP_DEBUG', true);
```

With debug on, CQIP will warn you if required keys are missing from your blocks, if FAQ items use the wrong key names, or if the schema contains null values. Always disable before going live.

Navigation

Further down `cqip-config.php`, find the `CQIP_NAV` constant. Update the navigation links to match your site structure:

```
define('CQIP_NAV', [  
    ['label' => 'Services', 'url' => '/services/'],  
    ['label' => 'About',    'url' => '/about/'],  
    ['label' => 'Contact',  'url' => '/contact/'],  
]);
```

Save `cqip-config.php`. Upload the entire `cqip/` folder to your server.

3. Visual design and layout

Your site appearance is controlled by `cqip/assets/css/site.css`. This file is yours to edit. The comments inside it explain every section.

Changing your brand colour

Find the Design tokens section near the top of `site.css`. Change `--brand-primary` to your brand colour. Everything else updates automatically.

```
:root {  
  --brand-primary:      #2c5282; /* Change this to your brand colour */  
  --brand-primary-dark: #1a365d; /* Darker shade – used for hover states */  
  --brand-primary-bg:   #ebf4ff; /* Very light tint – backgrounds, callouts */  
}
```

Contrast check: ensure your brand colour on white background meets WCAG 2.1 AA (4.5:1 ratio minimum for normal text). Use webaim.org/resources/contrastchecker/ to verify. The brand colour is used for links, buttons, and block accents throughout the site.

The header

The site header is defined in `cqip/templates/base.php`. It outputs:

- A sticky header that stays at the top of the screen as the user scrolls
- Your logo (from `cqip/assets/images/logo.png`)
- Navigation links from `CQIP_NAV` in `cqip-config.php`
- A CTA button — by default "Contact us" linking to `/contact/`

To change the CTA button, open `base.php` and find the `site-header__cta` section:

```
<div class="site-header__cta">  
  <a href="/contact/" class="btn btn-primary">Contact us</a>  
</div>
```

Replace the text and href with whatever fits your site. Common patterns: "Get a quote", "Book a call", "Request a demo".

The footer

The footer is also in `base.php`, in the Site Footer section. It has four columns:

- Column 1: Brand name, tagline, and copyright
- Column 2: Services links
- Column 3: Company links (About, Contact)
- Column 4: Legal links (Terms, Privacy)

Edit the links in each column to match your site structure. The footer background colour is set by `--brand-footer` in `site.css` (default: very dark, near black).

Page sections

The provided `site.css` includes styles for common section types. Use them by adding the appropriate class to your section elements:

Class	Visual effect
<code>.section</code>	Standard section padding — top and bottom
<code>.section--white</code>	White background
<code>.section--surface</code>	Off-white (<code>#f7fafc</code>) background — alternate sections
<code>.hero</code>	Full-width coloured hero with white text
<code>.page-hero</code>	Smaller coloured header for inner pages
<code>.cta-section</code>	Full-width coloured call-to-action section
<code>.stats-row</code>	Horizontal strip of stat numbers with labels
<code>.container</code>	Max-width centred content wrapper

Alternate `section--white` and `section--surface` to create visual rhythm down the page. Use the `hero` class for the home page hero and `page-hero` for all other pages.

Replacing the logo

The placeholder logo at `cqip/assets/images/logo.png` will display in the header and footer. Replace it with your own:

1. Create your logo as a PNG file, at least 72px x 72px
2. Name it `logo.png` (or update the filename in `cqip-config.php` at `CQIP_ORG_LOGO` and in `base.php`)
3. Upload it to `cqip/assets/images/`



4. Building the home page

Create `index.php` in your site root. This is the most important page — it sets the tone for the whole site and handles the majority of organic search traffic.

The page structure

The Clearwater Security home page uses this section order:

Section	Content
Hero	Headline, sub-headline, CTA buttons
Answer First block	Direct answer to primary question
Stats strip	4 key numbers
Capabilities block	6 service items in a grid
Solutions block	3 problem/solution pairs
FAQ block	5 FAQ items
CTA section	Headline and buttons

Step 1 — Start the file

```
<?php
declare(strict_types=1);
require_once __DIR__ . '/cqip/cqip-config.php';
```

Step 2 — Define FAQ items

Define your FAQ items before the `$blocks` array. The same variable feeds both the visible accordion on the page and the hidden `FAQPage` schema.

Always use 'q' and 'a' as key names. Using 'question' and 'answer' causes the FAQ block to render nothing and `FAQPage` schema to show null values. This is the single most common mistake when building with CQIP Static.

```
$faqItems = [
    [
        'q'      => 'What areas of Edinburgh do you cover?',
        'a'      => 'Clearwater Security installs access control systems across Edinburgh
and the Lothians.',
        'visible' => true,
```

```
],
[
  'q'      => 'How long does a standard installation take?',
  'a'      => 'A single-door installation takes four to six hours. Multi-door systems
are scoped individually.',
  'visible' => true,
],
[
  'q'      => 'Do you offer maintenance contracts?',
  'a'      => 'Yes. Annual contracts cover servicing and priority four-hour callout
from £199 per year.',
  'visible' => true,
],
];
```

Step 3 — Define the blocks

The `$blocks` array is the content data for the page. Each block has an `on` key. Set it to `true` for blocks you want to use, `false` (or omit) for blocks you do not need.

```
$blocks = [
  'answer_first' => [
    'on'      => true,
    'heading' => 'What does Clearwater Security install?',
    'answer'  => 'Clearwater Security installs access control systems in Edinburgh.',
    'context' => 'NSI Gold approved. Serving Edinburgh and the Lothians since 2009.',
    'label'   => 'Quick Answer',
    'level'   => 'h2',
  ],
  'capabilities' => [
    'on'      => true,
    'heading' => 'Access control services',
    'cols'    => 3,
    'items'   => [
      ['title' => 'New installations', 'description' => 'Paxton Net2 and SALTO XS4
installed.'],
      ['title' => 'System upgrades',   'description' => 'Legacy systems replaced with
modern access.'],
      ['title' => 'Maintenance',       'description' => 'Annual contracts with priority
callout.'],
    ],
  ],
  'faq' => [
    'on'      => true,
    'heading' => 'Common questions',
    'open_first' => true,
    'items'   => $faqItems,
```

```
],  
];
```

Step 4 — Build schema and set meta

The schema builder takes your \$blocks data and produces a complete, validated @graph. You pass it the entity type, the page URL, and the FAQ items.

```
$schemaJson = cqip_build_schema(  
    blocks:      $blocks,  
    entity_type: 'Service',  
    page_url:    CQIP_SITE_URL . '/',  
    page_title:  'Access Control Installation Edinburgh',  
    page_desc:   'Clearwater Security – Edinburgh and the Lothians.',  
    all_faqs:    $faqItems  
);  
  
$meta = [  
    'title'      => 'Access Control Installation Edinburgh',  
    'description' => 'Clearwater Security installs access control in Edinburgh. NSI Gold  
approved.',  
    'canonical'  => CQIP_SITE_URL . '/',  
];
```

Step 5 — Render the sections

Use ob_start() to capture your HTML, render the CQIP blocks into it, then pass it to the base template.

```
ob_start();  
?>  
  
<section class="hero" aria-labelledby="hero-heading">  
    <div class="hero__inner">  
        <h1 id="hero-heading">Access control you can trust.</h1>  
        <p class="hero__sub">NSI Gold approved. Edinburgh and the Lothians.</p>  
        <div class="hero_ctas">  
            <a href="/contact/" class="btn btn-white btn-lg">Get a free site survey</a>  
            <a href="/services/" class="btn btn-outline-white btn-lg">Our services</a>  
        </div>  
    </div>  
</section>  
  
<section class="section section--white">  
    <div class="container" style="max-width:860px;">
```

```
<?php cqip_render_answer_first($blocks['answer_first']); ?>
</div>
</section>

<div class="stats-row">
  <div class="stat-item">
    <span class="stat-item__number">15+</span>
    <span class="stat-item__label">Years in Edinburgh</span>
  </div>
  <div class="stat-item">
    <span class="stat-item__number">2,400+</span>
    <span class="stat-item__label">Doors installed</span>
  </div>
  <div class="stat-item">
    <span class="stat-item__number">NSI</span>
    <span class="stat-item__label">Gold approved</span>
  </div>
</div>

<section class="section section--surface">
  <div class="container">
    <?php cqip_render_capabilities($blocks['capabilities']); ?>
  </div>
</section>

<section class="section section--surface">
  <div class="container" style="max-width:780px;">
    <?php cqip_render_faq($blocks['faq']); ?>
  </div>
</section>

<section class="cta-section">
  <h2>Ready to secure your premises?</h2>
  <div class="cta-section__btns">
    <a href="/contact/" class="btn btn-white btn-lg">Request a free site survey</a>
  </div>
</section>

<?php
$bodyContent = ob_get_clean();
require_once __DIR__ . '/cqip/templates/base.php';
```

Upload index.php and load your site in a browser. You should see the hero, answer first block, stats, capabilities, FAQ, and CTA.

5. Building a service page

Create the folder `services/access-control/` and inside it create `index.php`.

Service pages use the `service-page.php` template. This template handles the HTML wrapper, schema, and meta automatically. You only set the variables and blocks.

What `service-page.php` expects

Variable	Type
<code>\$serviceUrl</code>	string
<code>\$serviceTitle</code>	string
<code>\$serviceDesc</code>	string
<code>\$parentLabel</code>	string
<code>\$parentUrl</code>	string
<code>\$blocks</code>	array
<code>\$allFaqs</code>	array

The FAQ Schema Iceberg

On this service page, we use more FAQ items in schema than we show on the page. This is the FAQ Schema Iceberg pattern.

Items with `visible: true` appear in the accordion. Items with `visible: false` exist only in the `FAQPage` schema — AI and search crawlers read them, but visitors do not see them.

```
// Visible items – shown on page
$faqItems = [
    ['q' => 'Which Paxton Net2 products do you install?',
     'a' => 'We install the full Paxton Net2 range including Net2 Pro.',
     'visible' => true],
];

// Schema-only items – not shown, but in FAQPage schema
$schemaOnlyFaqs = [
    ['q' => 'Does Paxton Net2 integrate with CCTV?',
     'a' => 'Yes. Paxton Net2 integrates with most IP CCTV platforms.',
     'visible' => false],
];
```

```
];

// Pass all items to cqip_build_schema via $allFaqs
$allFaqs = array_merge($faqItems, $schemaOnlyFaqs);
```

Complete service page file

```
<?php
declare(strict_types=1);
require_once __DIR__ . '/../../cqip/cqip-config.php';

$faqItems = [
    ['q' => 'Which Paxton products do you install?',
     'a' => 'We install the full Paxton Net2 range including Net2 Pro.',
     'visible' => true],
];

$blocks = [
    'answer_first' => [
        'on' => true,
        'level' => 'h1',
        'heading' => 'What does Paxton Net2 access control involve?',
        'answer' => 'Paxton Net2 manages entry to multiple doors. Cards, fobs, or mobile
credentials restrict access per user and time.',
    ],
    'technical' => [
        'on' => true,
        'heading' => 'System specifications',
        'specs' => [
            ['name' => 'Doors per controller', 'value' => 'Up to 2', 'unit' => ''],
            ['name' => 'Users per system', 'value' => 'Up to 50,000', 'unit' => ''],
            ['name' => 'Network protocol', 'value' => 'TCP/IP', 'unit' => ''],
        ],
    ],
    'faq' => ['on' => true, 'items' => $faqItems],
];

$serviceUrl = CQIP_SITE_URL . '/services/access-control/';
$serviceTitle = 'Paxton Net2 Access Control Installation Edinburgh';
$serviceDesc = 'Authorised Paxton Net2 installers in Edinburgh. NSI Gold approved.';
$parentLabel = 'Services';
$parentUrl = CQIP_SITE_URL . '/services/';
$allFaqs = $faqItems;

require __DIR__ . '/../../cqip/templates/service-page.php';
```




Note the require path: `'../../cqip/cqip-config.php'` goes up two directory levels from `services/access-control/` to reach the site root, then down into `cqip/`. Get this path right or the page will not load.

6. Building an about page

Create about/index.php. The about page uses the organization-page.php template and introduces two new blocks: compliance and entity_review.

The compliance block

The compliance block displays formal certifications and accreditations. Each item maps to a hasCredential node in the Organization schema — important for industries where certification is a buying signal.

```
'compliance' => [
  'on'      => true,
  'heading' => 'Accreditations',
  'items'   => [
    [
      'name'          => 'NSI Gold – Access Control',
      'issuer'        => 'National Security Inspectorate',
      'cert_number'   => 'ACM 12345',
      'valid_from'    => '2024-01-15',
      'valid_until'   => '2025-01-14',
      'scope'         => 'Electronic access control installation',
    ],
  ],
],
```

The entity_review block

The entity_review block displays client review cards and generates AggregateRating and Review schema. Set visible: true on reviews you want shown on the page.

```
'entity_review' => [
  'on'      => true,
  'heading' => 'What clients say',
  'layout'  => 'cards',
  'items'   => [
    [
      'reviewer_name'      => 'James Mackintosh',
      'reviewer_title'     => 'Facilities Manager',
      'reviewer_organization' => 'Forth Valley College',
      'rating'             => 5,
      'review_text'        => 'Project came in on time and within budget.',
      'review_date'        => '2024-09-12',
      'verified'           => true,
    ],
  ],
],
```



```
        'visible'          => true,  
      ],  
    ],  
  ],
```

7. Validating your schema

Once your pages are uploaded, validate the schema before considering them live. This takes about five minutes per page and confirms the content is being read correctly by Google.

Google Rich Results Test

Visit search.google.com/test/rich-results and enter your page URL. For each page you should see:

4. Organization entity detected, no errors
5. Service or WebPage entity detected, no errors
6. FAQPage entity detected, with questions populated (not null)
7. BreadcrumbList detected on service and about pages

Pass the Rich Results Test with zero errors before you consider the page live. Warnings are acceptable. Errors mean the schema is broken and will not generate rich results.

Checking the raw schema

In any browser, right-click the page and choose View Page Source (Ctrl+U on Windows, Cmd+U on Mac). Search for `application/ld+json`. You should see exactly one script block. If you see two, there is a duplicate schema source — check that no other tool on your server is outputting JSON-LD.

Fixing null FAQ values

If the Rich Results Test shows FAQPage with null question names, your FAQ items are using the wrong key names:

```
// Wrong – produces null values
['question' => 'Your question?', 'answer' => 'Your answer.']

// Correct
['q' => 'Your question?', 'a' => 'Your answer.']
```

Change all instances from question/answer to q/a throughout the page file and re-upload.

8. What you have built — and why it matters

When your Clearwater Security site is live and validated, here is exactly what you have achieved.

What is on the page

Element	What CQIP built
Organization schema	Entity graph with name, legal name, address, contact, and @id cross-referencing
Service schema	hasOfferCatalog from capabilities block, serviceOutput from solutions block
FAQPage schema	mainEntity populated from \$faqItems — eligible for Google FAQ rich results
BreadcrumbList	Automatic from \$parentLabel and \$parentUrl on service pages
Speakable schema	answer_first block signals AI Overview systems this content is citation-ready
AggregateRating	From entity_review block — star rating visible in search results
hasCredential	From compliance block — NSI certification in Organization schema
Canonical URL	Set per-page, output in <head> and og:url — no duplicate content issues
Open Graph	og:title, og:description, og:image, og:type — correct social preview on every page
Twitter Card	summary_large_image with title, description, and image
Robots meta	Output only when non-default — no unnecessary tags on indexed pages

How this compares to a site built without CQIP

The same site built with standard HTML and a separate schema plugin would typically have:

- Schema output as isolated types — Organization on one page, FAQPage on another — with no @id cross-referencing between them. Google processes these as unconnected fragments rather than an entity graph.
- Client-side schema injection via JavaScript — the crawler has to return for a second pass to index it. Server-side schema in <head> is indexed on the first pass.
- No Speakable markup — the page cannot signal citation readiness to AI Overview systems without it.
- No automatic BreadcrumbList — bread crumb rich results require manual markup or plugin configuration.
- Schema that needs manual per-page configuration — adding a new FAQ requires updating both the content and the schema separately.

With CQIP Static, the schema is a direct output of the content data. Adding a new FAQ item to \$faqItems updates both the visible accordion and the FAQPage schema in a single edit. The @graph is assembled automatically with correct @id cross-referencing on every page.

What to expect after launch

Page speed

CQIP Static serves flat PHP files with no database queries. A typical shared hosting server will return pages in under 100ms to first byte. Google PageSpeed Insights scores of 90+ are achievable without further optimisation.

Rich results in Google Search

FAQPage rich results and other schema-driven features appear after Google crawls and indexes the page. On a new domain this typically takes one to four weeks. On an established domain it can be faster. You can request indexing via Google Search Console to accelerate this.

AI citation (AEO)

Structured content with Speakable markup and clear entity relationships performs measurably better in AI retrieval benchmarks than unstructured HTML. Expect improved AI Overview and Perplexity citation rates within four to eight weeks of deployment. This is not guaranteed — AI systems choose citations based on authority, freshness, and relevance as well as structure — but CQIP gives the content every possible structural advantage.

Ongoing maintenance

Because CQIP separates content data (\$blocks) from rendering logic (templates and block files), updating content means editing the data arrays at the top of a page file. The schema, meta, and markup update automatically. There is no separate schema editor, no per-page configuration panel, and no plugin to update.

CQIP (Content Quality and Intelligence Policy) is a methodology standard, not a technology stack. Implementations are available for static PHP sites, WordPress, page generators, and other document management systems. The platform is the implementation detail — the standard and the eight canonical block types are consistent across all implementations.
thecontentframework.com