Building ACLHeal.com: Complete technical blueprint for a large medical static site

Astro emerges as the optimal static site generator for ACLHeal.com, offering the ideal balance of content management capabilities, SEO features, and interactive functionality for a 500-2000+ page medical resource site on GitHub Pages. This comprehensive guide provides a battle-tested implementation strategy covering architecture, deployment, SEO, interactive tools, and content workflows—enabling you to build a definitive ACL recovery resource that can scale from launch through years of content growth.

The recommended stack combines **Astro for site generation**, **Cloudflare Pages for hosting** (with GitHub Pages as a viable starting point), **Pagefind for search**, **Alpine.js for interactivity**, and **Decap CMS for content management**. This architecture delivers sub-second page loads, handles medical schema markup elegantly, and supports non-technical contributors through Git-based workflows.

Static site generator selection: Why Astro wins for medical content

For a large health content site like ACLHeal.com, the choice between Hugo's raw speed and Astro's modern content management creates a clear decision point. **Astro's Content Collections API with Zod schema validation** provides type-safe content management Astro that's invaluable for structured medical data—ensuring exercise metadata, recovery phases, and author credentials are always correct at build time.

Build performance benchmarks reveal the tradeoffs across major generators:

Pages	Hugo	Eleventy	Astro	Next.js (Static)
500	0.11s	0.68s	3.17s	9.55s
1,000	0.17s	0.91s	5.10s	14.30s
2,000	0.35s	1.25s	**9.79s**	25.04s
4,000	0.68s	1.94s	22.91s	70.65s

Hugo dominates raw speed—**20-40x faster** than JavaScript-based alternatives. However, Astro's ~10 second builds for 2,000 pages remain well within GitHub Actions' 10-minute timeout while providing superior developer experience for structured medical content. Astro's **Islands Architecture** enables selective JavaScript hydration, shipping zero JS by default Astro while allowing interactive components Criztec Technologies (exercise filtering, screening tools) to load on demand. CloudCannon

The content schema validation proves particularly valuable for YMYL medical content:

```javascript
// Astro Content Collections schema for exercises
const exerciseCollection = defineCollection({
  schema: z.object({
    title: z.string(),
    recoveryPhase: z.enum(['pre-surgery', 'week-0-2', 'week-3-6', 'month-2-3', 'month-4-6', 'month-7-12', 'year-1-2']),
    difficulty: z.enum(['beginner', 'intermediate', 'advanced']),
    bodyParts: z.array(z.string()),
    equipment: z.array(z.string()),
    medicalReviewDate: z.date(),
    reviewedBy: reference('authors'), // Type-safe author references
    videoUrl: z.string().url().optional(),
  })
});
```

Choose Hugo instead if: build speed is critical (multiple daily content updates), your team prefers Go templating, or the site is purely content with no interactive filtering.

GitHub Pages constraints and strategic workarounds

GitHub Pages imposes hard limits that directly impact architecture decisions for a large medical site with video content:

Limit	Value	Impact on ACLHeal.com
Published site size	**1 GB maximum**	Cannot host videos; images need external CDN
Bandwidth	100 GB/month (soft)	May need Cloudflare in front for caching
Build timeout	10 minutes	Astro's 2,000-page builds (~10s) work fine
Repository size	1 GB recommended	Use Git LFS or external storage for media
Individual file	100 MB max	All videos must be externally hosted

The 1GB site limit creates a fundamental architectural requirement: separate content from media. A 2,000-page site with optimized images will likely hit 500-800MB, leaving no room for video.

Video hosting strategy for exercise demonstrations

For ACL rehabilitation content requiring extensive exercise videos, implement a **hybrid hosting approach**:

YouTube (primary): Free unlimited hosting with excellent CDN performance. Videos appear in Google Video Carousel, increasing SERP visibility. Use `youtube-nocookie.com` embed domain for GDPR compliance. Add VideoObject schema markup and transcripts for SEO.

Bunny.net Stream (premium content): At ~$5/TB bandwidth plus $0.01/GB storage, this provides ad-free, customizable playback for branded exercise content requiring precise control.

Create separate video sitemaps to maximize indexation:

```xml
<sitemap>
  <loc>https://aclheal.com/video-sitemap.xml</loc>
  <lastmod>2024-12-19</lastmod>
</sitemap>
```

When to migrate to Cloudflare Pages

Monitor these thresholds—when any are approached, migrate to **Cloudflare Pages** (unlimited bandwidth, 20-minute builds, global CDN):

- Site output approaching 800MB
- Monthly bandwidth exceeding 80GB
- Build times exceeding 8 minutes
- Need for preview deployments on pull requests

Cloudflare Pages deployment requires minimal changes: update the GitHub repository connection and configure build settings. The migration path from GitHub Pages is straightforward.

Medical schema markup implementation for YMYL authority

As a YMYL (Your Money Your Life) health site, ACLHeal.com requires comprehensive structured data to establish E-E-A-T signals for Google's quality raters. Implement these schema types using JSON-LD with the `@graph` pattern to combine multiple types per page:

Core medical schema types

MedicalCondition for ACL injury information pages:

```json
```

```
{
  "@type": "MedicalCondition",
  "name": "Anterior Cruciate Ligament Tear",
  "alternateName": ["ACL tear", "ACL injury"],
  "signOrSymptom": [
    {"@type": "MedicalSymptom", "name": "Knee instability"},
    {"@type": "MedicalSymptom", "name": "Swelling within 24 hours"}
  ],
  "code": {
    "@type": "MedicalCode",
    "code": "S83.51",
    "codingSystem": "ICD-10"
  }
}
```

HowTo for exercise instructions (critical for rich results):

```json
{
  "@type": "HowTo",
  "name": "How to Perform Quad Sets for ACL Recovery",
  "totalTime": "PT10M",
  "step": [
    {
      "@type": "HowToStep",
      "position": 1,
      "name": "Position yourself",
      "text": "Sit or lie with leg extended straight"
    }
  ],
  "video": {
    "@type": "VideoObject",
    "name": "Quad Set Exercise Demonstration",
    "thumbnailUrl": "https://aclheal.com/images/quad-set-thumb.jpg",
    "uploadDate": "2024-06-15",
    "duration": "PT3M20S"
  }
}
```

Person schema for author and medical reviewer attribution—essential for E-E-A-T:

```json
```

```
{
  "@type": "Person",
  "@id": "https://aclheal.com/authors/dr-sarah-johnson#person",
  "name": "Dr. Sarah Johnson, PT, DPT, OCS",
  "jobTitle": "Physical Therapist",
  "hasCredential": [
    {
      "@type": "EducationalOccupationalCredential",
      "credentialCategory": "certification",
      "name": "Orthopedic Clinical Specialist (OCS)"
    }
  ],
  "alumniOf": {"@type": "EducationalOrganization", "name": "University of Southern California"}
}
```

E-E-A-T optimization checklist

Every content page must display:

1. **Author byline** with credentials (PT, DPT, MD, OCS)
2. **Medical reviewer attribution** with "Medically Reviewed By:" disclosure
3. **Review dates**: "Last Medical Review: November 15, 2024"
4. **Citations** to peer-reviewed sources (PubMed, ACSM, AAOS guidelines)
5. **Medical disclaimer** prominently placed

Implement build-time validation to catch stale content:

```javascript
// Flag articles with medical reviews older than 6 months
const reviewDate = new Date(frontmatter.medicalReviewDate);
const sixMonthsAgo = new Date();
sixMonthsAgo.setMonth(sixMonthsAgo.getMonth() - 6);
if (reviewDate < sixMonthsAgo) {
  console.warn(`⚠️ ${file}: Medical review expired`);
}
```

Interactive tools without server infrastructure

ACLHeal.com's screening questionnaires, progress trackers, and exercise filtering can all operate entirely client-side, avoiding HIPAA concerns while providing valuable user functionality.

Screening questionnaires (PHQ-9, GAD-7, ACL-RSI)

Both PHQ-9 (depression) and GAD-7 (anxiety) are freely available via Pfizer with no licensing restrictions. Pfizer The ACL-RSI (psychological readiness for return to sport) uses a 12-item scale where scores ≥65 correlate with successful return to sport.

Alpine.js provides the ideal lightweight framework (~8KB gzipped) for these interactive forms:

```html
<div x-data="phq9Questionnaire()" x-init="loadSavedProgress()">
  <form @submit.prevent="calculateScore">
    <template x-for="(question, index) in questions" :key="index">
      <fieldset>
        <legend x-text="question.text"></legend>
        <label x-for="option in options">
          <input type="radio"
                 :name="'q' + index"
                 :value="option.value"
                 x-model="responses[index]"
                 @change="saveProgress">
          <span x-text="option.label"></span>
        </label>
      </fieldset>
    </template>
    <button type="submit" :disabled="!isComplete">Calculate Score</button>
  </form>

  <div x-show="score !== null" class="results">
    <h2>Your Score: <span x-text="score"></span></h2>
    <p x-text="interpretation"></p>
  </div>
</div>
```

Critical disclaimers must accompany all screening tools:

- "This screening tool is for informational purposes only and is NOT a diagnostic instrument"
- "Your responses are stored ONLY on your device and never transmitted to any server"

- Crisis resources for concerning scores

Progress trackers using localStorage

Store user progress entirely client-side for privacy:

javascript

```javascript
class ProgressTracker {
  constructor(storageKey = 'aclheal_progress') {
    this.storageKey = storageKey;
    this.data = this.load();
  }

  toggleMilestone(milestoneId) {
    this.data.milestones[milestoneId] = !this.data.milestones[milestoneId];
    this.save();
  }

  markExerciseComplete(exerciseId) {
    const today = new Date().toISOString().split('T')[0];
    if (!this.data.exercises[exerciseId]) {
      this.data.exercises[exerciseId] = [];
    }
    this.data.exercises[exerciseId].push(today);
    this.save();
  }
}
```

Provide **export/import functionality** so users can backup their data or transfer between devices.

Exercise library search with Pagefind

Pagefind is the clear winner for static site search, indexing MDN's entire 14,000+ page site in under 300KB total bandwidth. It runs after any SSG build:

bash

```bash
npx pagefind --site public --output-path public/pagefind
```

Configure filtering with data attributes in your exercise templates:

html

```
<article data-pagefind-body>
  <h1 data-pagefind-meta="title">Quad Sets Exercise</h1>
  <span data-pagefind-filter="stage">Week 1-2</span>
  <span data-pagefind-filter="difficulty">Beginner</span>
  <span data-pagefind-filter="equipment">None</span>
</article>
```

Content architecture for recovery-stage navigation

Organize content using a **hub-and-spoke model with temporal hierarchy** that mirrors the patient's recovery journey:

```
/
├── pre-surgery/          # Hub: Pre-Surgery Preparation
│   ├── what-to-expect/
│   ├── prehab-exercises/
│   └── mental-preparation/
├── week-0-2/             # Hub: Immediate Post-Op
│   ├── pain-management/
│   ├── wound-care/
│   └── initial-exercises/
├── week-3-6/             # Hub: Early Recovery
├── month-2-3/            # Hub: Progressive Healing
├── month-4-6/            # Hub: Strength Building
├── month-7-12/           # Hub: Return to Activity
├── year-1-2/             # Hub: Long-Term Recovery
└── exercises/            # Cross-timeline Exercise Library
    ├── by-stage/
    ├── by-type/
    └── by-equipment/
```

Comprehensive frontmatter schema

```yaml
```

```yaml
---
title: "Quad Sets: Week 1-2 Post-Op"
description: "Learn proper quad set technique for early ACL recovery"
date: 2024-01-15
lastmod: 2024-12-01

# Recovery stage (controlled vocabulary)
stage: "week-0-2"
stageLabel: "Immediate Post-Op (Week 0-2)"
stageOrder: 2

# Content classification
contentType: "exercise"
exercise:
  type: "strengthening"
  difficulty: "beginner"
  duration: "5-10 minutes"
  reps: "3 sets of 10"
  frequency: "3-4x daily"

# Taxonomy
bodyParts: [quadriceps, knee]
equipment: [none]
protocols: ["standard-acl", "dr-smith-protocol"]
graftTypes: ["patellar-tendon", "hamstring", "allograft"]

# Medical attribution (required for E-E-A-T)
author: "dr-sarah-johnson"
reviewedBy: "dr-michael-chen"
medicalReviewDate: 2024-11-15
nextReviewDate: 2025-05-15

# Related content
relatedExercises: ["straight-leg-raises", "heel-slides"]
nextExercises: ["mini-squats", "step-ups"]

# Display settings
showInExerciseLibrary: true
featured: true
videoUrl: "https://youtube.com/..."
---
```

Internal linking for topical authority

Pillar pages (5,000+ words) should link to 30-70 cluster articles. Each **cluster page** links to:

- 3-5 same-stage articles
- 2-3 adjacent-stage articles
- 1-2 pillar page references

Implement automatic related content generation in your SSG templates based on shared taxonomy terms.

Build and deployment optimization

GitHub Actions workflow with caching

```
yaml
```

```
name: Deploy to GitHub Pages

on:
  push:
    branches: [main]
  schedule:
    - cron: '0 6 * * *'  # Daily rebuild

jobs:
  build:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v4
        with:
          fetch-depth: 0  # Full history for lastmod dates

      - name: Setup Node
        uses: actions/setup-node@v4
        with:
          node-version: '20'
          cache: 'npm'

      - name: Cache build outputs
        uses: actions/cache@v4
        with:
          path: |
            .cache
            node_modules/.cache
            public
          key: ${{ runner.os }}-build-${{ hashFiles('**/package-lock.json') }}-${{ hashFiles('content/**') }}

      - run: npm ci
      - run: npm run build
      - run: npx pagefind --site public  # Generate search index

      - name: Deploy to GitHub Pages
        uses: actions/deploy-pages@v4
```

Preview deployments for content review

Enable content team reviews before publishing:

yaml

```
on:
  pull_request:
    branches: [main]

jobs:
  preview:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v4
      - run: npm ci && npm run build
      - name: Deploy Preview
        uses: peaceiris/actions-gh-pages@v4
        with:
          publish_dir: ./public
          destination_dir: preview/${{ github.event.pull_request.number }}
```

Content management for medical teams

Decap CMS configuration for non-technical contributors

yaml

```
# admin/config.yml
backend:
  name: git-gateway
  branch: main

publish_mode: editorial_workflow  # Draft → In Review → Ready

collections:
  - name: "exercises"
    label: "Exercises"
    folder: "content/exercises"
    create: true
    fields:
      - {label: "Title", name: "title", widget: "string"}
      - {label: "Stage", name: "stage", widget: "select",
         options: ["pre-surgery", "week-0-2", "week-3-6", "month-2-3", "month-4-6", "month-7-12", "year-1-2"]}
      - {label: "Difficulty", name: "difficulty", widget: "select",
         options: ["beginner", "intermediate", "advanced"]}
      - {label: "Medical Review Date", name: "medicalReviewDate", widget: "date"}
      - {label: "Reviewed By", name: "reviewedBy", widget: "relation",
         collection: "authors", search_fields: ["name"], value_field: "slug"}
      - {label: "Content", name: "body", widget: "markdown"}
```

Automated content validation

Add to your CI pipeline:

```yaml
- name: Validate Frontmatter Schema
  run: npx remark content/**/*.md

- name: Check Medical Review Dates
  run: node scripts/check-review-dates.js

- name: Verify Broken Links
  uses: lycheeverse/lychee-action@v1
```

Analytics and performance monitoring

Privacy-respecting analytics

Plausible ($9/month) or **Fathom** ($14/month) provide GDPR-compliant, cookie-free analytics. For free options, **Umami** (self-hosted) or **Cloudflare Web Analytics** (if using Cloudflare) work well.

Core Web Vitals optimization

Static sites inherently excel at performance. Ensure:

- Images use WebP format with proper `width`/`height` attributes
- Lazy load images and video embeds below the fold
- Inline critical CSS in `<head>`
- Defer non-essential JavaScript

Target metrics: **LCP < 2.5s**, **FID < 100ms**, **CLS < 0.1**

Lessons from successful large static sites

MDN Web Docs (14,000+ pages) demonstrates the viability of massive static sites, storing content in Markdown with structured metadata, using Rust-based build tools for speed, and distributing via CDN.

Healthline (40,000+ ranking pages) provides the E-E-A-T template for medical sites: clear topical silos, comprehensive author/reviewer attribution on every page, extensive structured data, and editorial policies detailing fact-checking processes.

Pagefind proves search scales elegantly—indexing MDN's entire site in under 300KB while maintaining fast queries across thousands of pages.

Conclusion: Implementation roadmap

Weeks 1-2: Set up Astro with Content Collections, define frontmatter schemas, configure validation scripts and CI pipeline.

Weeks 3-4: Deploy Decap CMS, configure editorial workflow, create content templates for exercises, guides, and protocols.

Weeks 5-8: Build initial content library, implement medical schema markup, add Pagefind search and Alpine.js interactive tools.

Weeks 9-12: Launch on GitHub Pages (or Cloudflare Pages if media-heavy), monitor Core Web Vitals, refine based on Search Console data.

The combination of Astro's type-safe content management, Pagefind's efficient search, and comprehensive medical schema markup positions ACLHeal.com to become a definitive ACL recovery resource—one that earns Google's trust for YMYL queries while serving patients with fast, accessible, interactive content throughout their recovery journey.