ReadyNest Real Estate

Developer Implementation Guide (v2)

Programmatic SEO at scale + Lead System — San Diego, Orange County, Los Angeles

Date: February 09, 2026

Key	Value
Goal	Generate free, high-intent Google leads by building topical authority for family-focused real estate across SoCal. This guide describes the technical build + SEO guardrails.
Core assets	1) Service pages (Buy / Sell & Upgrade / 45-Day Express) 2) County hubs 3) City pages (programmatic) 4) Scenario/Guide pages (linkable) 5) Google Business Profile support.
Inputs	CSV index plan + CSV generator inputs (city_json per city).
Critical SEO principle	Do not ship thousands of thin pages. Use indexing gates, internal linking clusters, and linkable guides to earn authority.

Non-negotiable: no guarantees about financing approvals or closing timelines. Use qualified language on all pages.

Table of Contents

1. Site architecture for maximum SEO impact

The site must be structured as topic clusters. City pages alone rarely earn links; linkable guide pages earn authority and pass it to city pages via internal links.

Required top-level sections

- **Services**: /buy, /sell-and-upgrade, /45-day-express
- **Areas**: /areas (hub) -> /areas/{county} -> /areas/{county}/{city}
- **Guides**: /resources/{slug} for scenario pages that attract links and build authority.
- **Conversion**: /get-started intake form used across pages.

Recommended additional page types (high ROI)

- **Neighborhood hub** (per top city): /areas/{county}/{city}/neighborhoods
- **Neighborhood pages**: /areas/{county}/{city}/{neighborhood}
- **Scenario pages** (USP): moving-before-baby, buying-while-pregnant, sell-first-vs-buy-first, 45-day-plan.

2. Programmatic page types (County / City / Neighborhood / Scenario)

Type	URL pattern	Primary search intent	Notes
County hub	/areas/{county}	Family homes + buyer agent in county	Topical hub linking to cities + services + guides
City page	/areas/{county}/{city}	Family-friendly neighborhoods in city	Must be unique + useful; include FAQs + CTA + links
Neighborhood hub	/areas/{county}/{city}/neighborhoods	Neighborhood exploration	Lists neighborhoods, links to 5-15 neighborhood pages
Scenario/Guide	/resources/{slug}	Problem intent (USP)	Linkable asset; funnels into county/city pages

Scenario page list (start with these 6)

- Moving Before Baby: a calm 30-day checklist
- Buying While Pregnant in California: realistic timeline and steps
- Sell First or Buy First? Decision framework for families
- 45-Day Plan: what must be true (qualification) and what we coordinate
- Baby-ready upgrades after closing (safe, quick wins)
- First-time buyer family edition: avoid common mistakes

3. Indexing strategy (Quality Gate) — when a page earns indexation

Instead of indexing everything, each city page must pass a Quality Gate. Pages that fail remain noindex,follow but can still be crawled and contribute to internal linking.

Gate item	Minimum requirement to index
Unique body content	900–1,400 words; not templated filler; includes city-specific considerations stated as considerations (not hard claims)
FAQ block	5–8 FAQs with short answers + FAQPage schema
Internal links	At least: county hub + 5 related cities + 2 guides + all 3 service pages
Value modules	At least 2: checklist, timeline, tradeoffs, questions-to-ask
Trust module	Short 'How ReadyNest helps' + CTA; no overpromises

Quality score recommendation

Optionally compute a QualityScore (0–100). Index only if score >= 80. This makes rollout safe at scale.

```
// lib/qualityGate.ts (example)
export function qualityScore(content: { wordCount: number; faqCount: number; internalLinks:
number; valueModules: number }) {
  let score = 0;
  if (content.wordCount >= 900) score += 30;
  if (content.wordCount >= 1200) score += 10;
  if (content.faqCount >= 5) score += 20;
  if (content.internalLinks >= 10) score += 20;
  if (content.valueModules >= 2) score += 20;
  return Math.min(score, 100);
}
```

4. Internal linking system (Topic clusters)

Internal links are the engine of programmatic SEO. Every page should push authority to the next layer.

Minimum internal links per page

- City page -> county hub, services, 2 guides, 5 related cities.
- County hub -> services, 6–15 city pages, 2–4 guides.
- Guide pages -> 1 county hub + 3–6 city pages + a single CTA to /get-started.

Related cities helper (deterministic)

```
// lib/relatedCities.ts (example heuristic - same county)
import { listCities } from "./cityData";

export function relatedCities(countySlug: string, citySlug: string, limit = 8) {
  const cities = listCities(countySlug).map(r => r.city_slug);
  // simple deterministic "nearby-like" set: take neighbors in sorted list
  const sorted = [...cities].sort();
  const idx = sorted.indexOf(citySlug);
  const picks: string[] = [];
  for (let i = 1; picks.length < limit && i < sorted.length; i++) {
    const left = sorted[(idx - i + sorted.length) % sorted.length];
    const right = sorted[(idx + i) % sorted.length];
    if (left !== citySlug) picks.push(left);
    if (picks.length < limit && right !== citySlug) picks.push(right);
  }
  return picks.slice(0, limit);
}
```

5. Technical SEO implementation (metadata, canonicals, sitemaps, robots)

Key rules: canonical must be correct, robots must respect index/noindex, and sitemaps should include only index-worthy URLs.

Sitemap strategy (recommended)

- Create a sitemap index: /sitemap.xml
- Split: /sitemap-main.xml (core pages), /sitemap-areas.xml (indexed city/county), /sitemap-guides.xml (resources).
- Only include rows where index == 'index' in /sitemap-areas.xml.

```
// app/sitemap.xml/route.ts (sitemap index example)
export async function GET() {
  const base = "https://YOUR_DOMAIN";
  const body = `<?xml version="1.0" encoding="UTF-8"?>
  <sitemapindex xmlns="http://www.sitemaps.org/schemas/sitemap/0.9">
    <sitemap><loc>${base}/sitemap-main.xml</loc></sitemap>
    <sitemap><loc>${base}/sitemap-areas.xml</loc></sitemap>
    <sitemap><loc>${base}/sitemap-guides.xml</loc></sitemap>
  </sitemapindex>`;
  return new Response(body, { headers: { "Content-Type": "application/xml" } });
}
```

Robots.txt route (Next.js)

```
// app/robots.txt/route.ts (example)
export async function GET() {
  const body = `User-agent: *
Disallow:

Sitemap: https://YOUR_DOMAIN/sitemap.xml
`;
  return new Response(body, { headers: { "Content-Type": "text/plain" } });
}
```

6. Schema (FAQ, Breadcrumbs, LocalBusiness, Service, Article)

- **City pages**: FAQPage + BreadcrumbList.
- **Service pages**: Service schema + LocalBusiness/RealEstateAgent sitewide.
- **Guides**: Article schema.

Breadcrumb schema (example)

```
<script type="application/ld+json">
{
  "@context":"https://schema.org",
  "@type":"BreadcrumbList",
  "itemListElement":[
    {"@type":"ListItem","position":1,"name":"Areas","item":"https://YOUR_DOMAIN/areas"},
    {"@type":"ListItem","position":2,"name":"Orange
    County","item":"https://YOUR_DOMAIN/areas/orange-county"},
    {"@type":"ListItem","position":3,"name":"Irvine","item":"https://YOUR_DOMAIN/areas/orange-county/irvine"}
  ]
}
</script>
```

7. Content generation workflow (prompt -> stored content -> render)

Do not generate content at request-time. Generate content in a batch, store it (MDX/JSON/DB), then render templates.

Recommended pipeline

- Batch job reads generator CSV -> sends city_json into the City Prompt -> receives structured output.
- Parse output into fields: metaTitle, metaDescription, h1, heroSubhead, sections[], faqs[].
- Persist: /content/areas/{county}/{city}.json (or CMS).
- City page renders from stored content + shared components (CTA, trust).

Stored content shape (example)

```
// content storage example (JSON shape)
{
  "metaTitle": "Family Friendly Neighborhoods in Irvine | ReadyNest",
  "metaDescription": "A calm guide for families planning, pregnant, or with a newborn...",
  "h1": "Family Friendly Neighborhoods in Irvine",
  "heroSubhead": "A calm plan for finding and preparing the right home...",
  "sections": [
    { "heading": "Why families choose Irvine", "body": ["...","..."] }
  ],
  "faqs": [
    { "q": "Do you work with buyers or sellers?", "a": "We help both..." }
  ]
}
```

8. Lead capture + tracking (forms, CRM, GA4)

Keep one conversion: /get-started. Embed the same form component on Home, Sell & Upgrade, Express, and City pages.

Tracking essentials

- Capture UTMs + referrer + landing page path. Save to CRM.
- GA4 events: form_start, form_submit_success, form_submit_error, cta_click, area_page_view.
- Do not send PII to analytics.

9. Google Business Profile workflow (free local leads)

GBP is a major free-lead channel. The website should support GBP with consistent NAP, service categories, and weekly content posts that link to a relevant guide or area page.

Weekly GBP cadence (recommended)

- 2 posts/week: one educational (guide), one listing/market update (no claims).
- Upload 5–10 new photos/month (office/team/neighborhood shots).
- Ask for reviews after each success milestone; respond to all reviews.

10. QA checklist and launch sequence

Launch sequence (safe)

1. Launch core pages first: Home + Services + County hubs + Guides.
2. Then index the first wave of cities (SD all; OC/LA top-20) only after Quality Gate passes.
3. Monitor Search Console: coverage, crawls, and performance; expand in batches.

Acceptance criteria

- No clipped text; all pages render cleanly; no broken links.
- Index/noindex matches CSV (spot-check 10 URLs).
- Sitemaps validate; robots.txt references sitemap index.
- Schema validates for FAQ + breadcrumbs on sample pages.
- Form -> CRM record -> internal notification works.

Appendix: Copy-paste code snippets

```
// app/areas/[county]/[city]/generateMetadata.ts (example)
import type { Metadata } from "next";
import { getCity } from "@/lib/cityData";
import { getStoredContent } from "@/lib/contentStore";

export async function generateMetadata({ params }): Promise<Metadata> {
  const row = getCity(params.county, params.city);
  if (!row) return {};
  const content = await getStoredContent(row.county_slug, row.city_slug); // optional
  const title = content?.metaTitle ?? row.primary_keyword;
  const description = content?.metaDescription ?? `Family-focused real estate guidance in
  ${row.city}.`;
  return {
    title,
    description,
    robots: row.index === "index" ? "index,follow" : "noindex,follow",
    alternates: { canonical: `https://YOUR_DOMAIN${row.url}` },
    openGraph: { title, description, url: `https://YOUR_DOMAIN${row.url}` }
  };
}
```