

The Vercel Launch *Kit*.

Copy-paste code for the fifty-seven things that stand between a site that looks finished and a site that is. Next.js App Router, TypeScript, ready to drop in.

```
app/sitemap.ts
app/robots.ts
app/layout.tsx  metadata, fonts, consent
app/api/contact/route.ts
components/JsonLd.tsx  and 13 more
```

18

FILES TO COPY, IN BUILD
ORDER

57

PRE-LAUNCH CHECKS THE KIT
COVERS

2 hrs

FROM BUILT SITE TO PROPERLY
LAUNCHED

START HERE

Eighteen files, in the order they unblock each other

This is the companion to the checklist. Same fifty-seven items, including the new data protection and fair-selling checks, except here they are code you can paste. Everything assumes **Next.js App Router with TypeScript, deployed on Vercel**, and everything degrades gracefully if your stack is slightly different.

Three conventions run through the whole kit. Learn them once and the rest reads itself.

1. One source of truth for the site URL

```
// lib/site.ts - every other file imports from here
export const site = {
  url: 'https://www.clientsite.co.uk', // canonical host, no trailing slash
  name: 'Client Name Ltd',
  phone: '+442080000000',
  whatsapp: '442080000000', // no plus sign for wa.me links
} as const
```

WHY Half the launch bugs in this kit come from the same URL being written four different ways. Write it once. Import it everywhere.

2. Placeholders you replace

`clientsite.co.uk`, `Client Name Ltd` and the Croydon addresses are placeholders. Find and replace them once per project and the whole kit is rebranded.

3. Every block is complete

No `// ... rest of your code` gaps. If a file appears here, it appears whole, so you can paste it and move on.

2 hours

That is the realistic total for the first site. Under an hour on the second, because most of these eighteen files are identical across projects and only `lib/site.ts` changes. This kit is designed to become your template, not your homework.

01 Sitemap and robots

CHECKS 1 & 2

Both are file conventions in the App Router, so they compile to real static files with correct content types. No route handlers, no XML strings.

app/sitemap.ts

```
import type { MetadataRoute } from 'next'
import { site } from '@lib/site'
import { getPages } from '@lib/content'

export default async function sitemap(): Promise<MetadataRoute.Sitemap> {
  // getPages returns [{ slug: 'services/rewiring', updatedAt: Date }]
  const pages = await getPages()

  return [
    {
      url: site.url,
      lastModified: new Date('2026-07-14'), // the real date
      changeFrequency: 'monthly',
      priority: 1,
    },
    ...pages.map((page) => ({
      url: `${site.url}/${page.slug}`,
      lastModified: page.updatedAt, // never Date.now()
      changeFrequency: 'monthly' as const,
      priority: 0.8,
    })),
  ]
}
```

THE TRAP Writing `new Date()` into every entry stamps the whole sitemap with build time. Google compares that against what actually changed, finds it lying, and starts ignoring the field. If you genuinely have no per-page date, omit `lastModified` entirely.

app/robots.ts

```
import type { MetadataRoute } from 'next'
import { site } from '@lib/site'

export default function robots(): MetadataRoute.Robots {
  return {
    rules: [{
      userAgent: '*',
      allow: '/',
      disallow: ['/api/', '/thank-you'], // keep the goal page out
    }],
    sitemap: `${site.url}/sitemap.xml`, // absolute, always
  }
}
```

02 Metadata and canonicals

CHECKS 3, 5 & 6

Set the base once in the root layout, then every page only has to declare what makes it different. Relative canonical and Open Graph paths resolve against metadataBase automatically.

app/layout.tsx (the metadata export)

```
import type { Metadata } from 'next'
import { site } from '@lib/site'

export const metadata: Metadata = {
  metadataBase: new URL(site.url),
  title: {
    default: 'Electrician in Croydon | Client Name Ltd',
    template: '%s | Client Name Ltd',
  },
  description: 'NICEIC registered electricians covering Croydon and '
    + 'south London. Same-day callouts, fixed quotes, no VAT surprises.',
  alternates: { canonical: '/' },
  openGraph: {
    type: 'website',
    locale: 'en_GB',
    url: '/',
    siteName: site.name,
    images: [{ url: '/og.jpg', width: 1200, height: 630 }],
  },
  twitter: { card: 'summary_large_image' },
  robots: { index: true, follow: true },
}
```

app/services/rewiring/page.tsx (every other page)

```
export const metadata: Metadata = {
  title: 'House Rewiring', // template adds the suffix
  description: 'Full and partial rewires in Croydon, from '
    + 'consumer unit to final socket. Fixed price after a free survey.',
  alternates: { canonical: '/services/rewiring' },
}
```

DO THIS Grep the project for `alternates` before you deploy. Every page file should return a hit. The pages that do not are the ones quietly competing with your homepage.

03 Structured data

CHECKS 8 & 9

JSON-LD is the only format Google recommends. One tiny component renders it, one plain object describes the business, and you validate both in the Rich Results Test before you ship.

components/JsonLd.tsx

```
export function JsonLd({ data }: { data: Record<string, unknown> }) {
  return (
    <script
      type="application/ld+json"
      dangerouslySetInnerHTML={{ __html: JSON.stringify(data) }}
    />
  )
}
```

lib/schema.ts

```
import { site } from '@lib/site'

export const localBusiness = {
  '@context': 'https://schema.org',
  '@type': 'Electrician', // most specific subtype that fits
  '@id': `${site.url}/#business`,
  name: site.name,
  url: site.url,
  telephone: site.phone,
  image: `${site.url}/og.jpg`,
  priceRange: '££',
  address: {
    '@type': 'PostalAddress',
    streetAddress: '1 High Street',
    addressLocality: 'Croydon',
    postalCode: 'CR0 1AA',
    addressCountry: 'GB',
  },
  areaServed: ['Croydon', 'Purley', 'Sutton', 'Caterham'],
  openingHoursSpecification: [{
    '@type': 'OpeningHoursSpecification',
    dayOfWeek: ['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday'],
    opens: '08:00',
    closes: '18:00',
  }],
}
```

THEN Render it once in the root layout with `<JsonLd data={localBusiness} />`, add a `Service` object per service page and an `FAQPage` wherever you answer real questions. Paste each rendered page into search.google.com/test/rich-results and fix what it flags.

04 Security headers and redirects

CHECKS 4, 7 & 50

One config file covers the headers a security scanner looks for and the 301s that carry the old site's authority across. Both are free and both take ten minutes.

next.config.ts

```
import type { NextConfig } from 'next'

const securityHeaders = [
  { key: 'Strict-Transport-Security',
    value: 'max-age=63072000; includeSubDomains; preload' },
  { key: 'X-Content-Type-Options', value: 'nosniff' },
  { key: 'Referrer-Policy', value: 'strict-origin-when-cross-origin' },
  { key: 'X-Frame-Options', value: 'SAMEORIGIN' },
  { key: 'Permissions-Policy',
    value: 'camera=(), microphone=(), geolocation=()' },
]

const nextConfig: NextConfig = {
  async headers() {
    return [{ source: '/*', headers: securityHeaders }]
  },
  async redirects() {
    // one line per URL the old site had indexed
    return [
      { source: '/our-services', destination: '/services', permanent: true },
      { source: '/contact-us', destination: '/contact', permanent: true },
      { source: '/blog/:slug', destination: '/advice/:slug', permanent: true },
    ]
  },
}

export default nextConfig
```

DO NOT CODE THE WWW REDIRECT Add both the apex and the www domain in the Vercel project's domain settings and mark one as the redirect target. The platform issues a 308 at the edge, which is faster and cannot be broken by a later code change.

GETTING THE OLD URLS Before you take the old site down, pull its indexed pages from Search Console, or fetch its sitemap. Every one of those URLs has accumulated authority. A 301 keeps it. A 404 throws it away.

05 Cookie consent that passes

CHECKS 26 & 27

Only build this for ad pixels, embeds or chat widgets. Cookieless analytics, and since 5 February 2026 first-party analytics used only to improve the site, need no banner. The architecture matters more than the styling: the tags are children of the gate, so they cannot load before a choice is made.

components/ConsentGate.tsx

```
'use client'
import { useEffect, useState, type ReactNode } from 'react'
import { flushAds, dropAds } from '@lib/track'

type Choice = 'granted' | 'denied'
const KEY = 'cookie-consent'

export function ConsentGate({ children }: { children: ReactNode }) {
  const [choice, setChoice] = useState<Choice | null>(null)
  const [asked, setAsked] = useState(false)

  useEffect(() => {
    const saved = localStorage.getItem(KEY) as Choice | null
    setChoice(saved)
    setAsked(true)
    if (saved === 'granted') flushAds()
  }, [])

  function decide(value: Choice) {
    localStorage.setItem(KEY, value)
    setChoice(value)
    if (value === 'granted') flushAds() // send what waited for consent
    else dropAds()
  }

  return (
    <>
      {choice === 'granted' && children}
      {asked && choice === null && (
        <div className="consent" role="dialog" aria-label="Cookie choices">
          <p>We use analytics cookies to see which pages bring in work.</p>
          <div className="consent-actions">
            {/* same size, same weight, same one click */}
            <button onClick={() => decide('denied')}>Reject all</button>
            <button onClick={() => decide('granted')}>Accept all</button>
          </div>
        </div>
      )}
    </>
  )
}
```

THE COMPLIANCE TEST Reject must be as easy as accept: equal prominence, equal clicks, no dark patterns. A "Manage preferences" link where the reject button should be is the exact pattern regulators enforce against. Since 5 February 2026 the maximum UK penalty for cookie breaches is £17.5 million or 4% of global turnover.

06 Conversion tracking

CHECKS 36 & 37

Three events, one file. Reporting pageviews tells the client nothing. Reporting that fourteen people tapped the phone number last month tells them what you are worth.

lib/track.ts

```
'use client'
import { track } from '@vercel/analytics'

type Ad = {name: string, params?: Record<string, string | number>}
type W = Window & { fbq?: (...a: unknown[]) => void; __adsOn?: boolean; __adQueue?: Ad[] }

// Ad platforms (Meta and friends) only after consent. Until then, events wait.
function ad(name: string, params?: Ad[]) {
  const w = window as W
  if (w.__adsOn) w.fbq?.('track', name, params)
  else (w.__adQueue ||= []).push([name, params])
}
export function flushAds() {
  const w = window as W; w.__adsOn = true
  for (const [n, p] of w.__adQueue ?? []) w.fbq?.('track', n, p)
  w.__adQueue = []
}
export function dropAds() { (window as W).__adQueue = [] }

// Vercel events are cookieless: no consent needed. Never send form answers to either.
export const lead = {
  call: () => track('call_click'),
  whatsapp: () => track('whatsapp_click'),
  form: (service: string) => { track('form_submit', { service }); ad('Lead') },
}
```

Using it on the elements that earn

```
import { site } from '@lib/site'
import { lead } from '@lib/track'

<a href={`tel:${site.phone}`} onClick={lead.call}>
  {site.phone}
</a>

<a
  href={`https://wa.me/${site.whatsapp}?text=Hi, I found you on your website`}
  onClick={lead.whatsapp}
  rel="noopener"
>
  WhatsApp us
</a>
```

COOKIELESS BY DEFAULT Vercel Web Analytics is included on Pro and sets no cookies. The ad events only leave the browser once ConsentGate calls `flushAds()`, and a reject throws them away.

THE THANK-YOU PAGE Send successful submissions to a real `/thank-you` route rather than swapping in a message. A URL can be counted, reported on, and used as a Google Ads goal later. A div cannot.

07 The contact route

CHECKS 45 TO 48

The most important fifty lines on the site. Validates, rate limits, catches bots without a CAPTCHA, writes to the database first and only then sends the email.

app/api/contact/route.ts

```
import { NextResponse } from 'next/server'
import { z } from 'zod'
import { Resend } from 'resend'
import { db } from '@lib/db'

const Body = z.object({
  name: z.string().trim().min(2).max(80),
  email: z.string().trim().email(),
  phone: z.string().trim().min(7).max(20),
  message: z.string().trim().min(10).max(2000),
  company: z.string().max(0), // honeypot: humans leave it empty
  startedAt: z.number(), // page-load time from the client
})

const hits = new Map<string, number[]>()

function tooMany(ip: string, max = 3, windowMs = 60_000) {
  const now = Date.now()
  const recent = (hits.get(ip) ?? []).filter((t) => now - t < windowMs)
  recent.push(now)
  hits.set(ip, recent)
  return recent.length > max
}

export async function POST(req: Request) {
  const ip = req.headers.get('x-forwarded-for')?.split(',')[0] ?? 'unknown'
  if (tooMany(ip)) {
    return NextResponse.json({ error: 'Slow down' }, { status: 429 })
  }

  const parsed = Body.safeParse(await req.json())
  if (!parsed.success) {
    return NextResponse.json({ error: 'Invalid' }, { status: 400 })
  }

  const { company, startedAt, ...enquiry } = parsed.data

  // Bot signals: honeypot filled, or submitted inside two seconds.
  // Return 200 so the bot never learns it was caught.
  if (company || Date.now() - startedAt < 2000) {
    return NextResponse.json({ ok: true })
  }

  await db.enquiry.create({ data: enquiry }) // 1. persist

  await new Resend(process.env.RESEND_API_KEY).emails.send({
    from: 'Website <enquiries@clientsite.co.uk>',
    to: process.env.LEAD_INBOX!,
    replyTo: enquiry.email, // 2. then notify
    subject: `New enquiry from ${enquiry.name}`,
    text: `${enquiry.name}\n${enquiry.phone}\n\n${enquiry.message}`,
  })

  return NextResponse.json({ ok: true })
}
```

08 The form component

CHECKS 22, 38 & 46

Real labels, a honeypot people never see, and the page-load timestamp that makes the two-second rule work.

components/ContactForm.tsx

```
'use client'
import { useRef, useState } from 'react'
import { useRouter } from 'next/navigation'
import { lead } from '@lib/track'

export function ContactForm({ service }: { service: string }) {
  const startedAt = useRef(Date.now())
  const [sending, setSending] = useState(false)
  const [failed, setFailed] = useState(false)
  const router = useRouter()

  async function onSubmit(e: React.FormEvent<HTMLFormElement>) {
    e.preventDefault()
    setSending(true)
    setFailed(false)

    const form = Object.fromEntries(new FormData(e.currentTarget))
    const res = await fetch('/api/contact', {
      method: 'POST',
      headers: { 'Content-Type': 'application/json' },
      body: JSON.stringify({ ...form, startedAt: startedAt.current }),
    })

    setSending(false)
    if (!res.ok) return setFailed(true)

    lead.form(service)
    router.push('/thank-you')
  }

  return (
    <form onSubmit={onSubmit} noValidate>
      <label htmlFor="name">Your name</label>
      <input id="name" name="name" autoComplete="name" required />

      <label htmlFor="phone">Phone</label>
      <input id="phone" name="phone" type="tel" autoComplete="tel" required />

      {/* invisible to people, irresistible to bots */}
      <input
        name="company" tabIndex=-1 autoComplete="off" aria-hidden="true"
        style={{ position: 'absolute', left: '-9999px' }}
      />

      <button disabled={sending}>
        {sending ? 'Sending' : 'Send enquiry'}
      </button>
      {failed && <p role="alert">That did not send. Call us instead.</p>}
    </form>
  )
}
```

09 Data protection in code

CHECKS 31, 33 & 34

Three promises the privacy policy makes that only code can keep: adults only, sensitive questions optional, and data deleted when you said it would be.

components/ContactForm.tsx (add only what the service needs)

```
<label>
  <input type="checkbox" name="adult" value="true" required />
  I confirm I am 18 or over
</label>

<fieldset>
  <legend>Any condition we should plan around? (optional, used only for your booking)</legend>
  {[ 'Yes', 'No', 'Prefer not to say' ].map((v) => (
    <label key={v}><input type="radio" name="health" value={v} /> {v}</label>
  ))}
</fieldset>
```

app/api/contact/route.ts (extend the zod schema)

```
adult: z.literal('true'),           // enforced here, not just in the browser
health: z.enum([ 'Yes', 'No', 'Prefer not to say' ]).optional(), // never required
```

app/api/cron/purge/route.ts

```
import { NextResponse } from 'next/server'
import { db } from '@lib/db'

// Monthly: delete enquiries older than the privacy policy promises.
export async function GET(req: Request) {
  const secret = process.env.CRON_SECRET
  if (!secret || req.headers.get('authorization') !== `Bearer ${secret}`) {
    return NextResponse.json({ error: 'Unauthorised' }, { status: 401 })
  }
  const cutoff = new Date()
  cutoff.setMonth(cutoff.getMonth() - 24) // match your policy
  const { count } = await db.enquiry.deleteMany({ where: { createdAt: { lt: cutoff } } })
  console.log(`purged ${count} enquiries before ${cutoff.toISOString()}`)
  return NextResponse.json({ ok: true, deleted: count })
}
```

vercel.json

```
{ "crons": [{ "path": "/api/cron/purge", "schedule": "0 3 1 * *" } ] }
```

THE SECRET Add `CRON_SECRET` in the project's environment variables and Vercel sends it on every cron call. Without it the route refuses everyone, which is the safe failure.

10 Selling fairly in code

CHECKS 40, 41 & 43

Two small components that keep an offer page on the right side of the CMA and the ASA. Notice what is missing: there is no countdown component in this kit, on purpose.

components/PriceLine.tsx

```
// The first payment and what comes after, side by side. No clock, no fake "was".
export function PriceLine({ intro, regular, every }: {
  intro?: number; regular: number; every: string
}) {
  const gbp = (p: number) => `£${(p / 100).toFixed(2).replace(/\.00$/, '')}`
  return intro && intro < regular ? (
    <p>
      <strong>{gbp(intro)}</strong> for your first {every},
      then {gbp(regular)} every {every}
    </p>
  ) : (
    <p><strong>{gbp(regular)}</strong> every {every}</p>
  )
}

// <PriceLine intro={799} regular={1299} every="4 weeks" /> and call the plan "4-weekly"
```

components/PartnerBox.tsx

```
export function PartnerBox({ name, href, cta, copy }: {
  name: string; href: string; cta: string; copy: string
}) {
  return (
    <aside className="partner">
      <p className="label">Ad · We earn a commission if you sign up with {name}</p>
      <p>{copy}</p>
      <a href={href} target="_blank" rel="sponsored noopener">{cta}</a>
    </aside>
  )
}
```

FINANCIAL PARTNERS For investments, pensions, mortgages or credit, **copy** must be the provider's own FCA-approved wording, pasted unchanged. Writing your own pitch for a regulated product can be a criminal offence for an unauthorised business.

URGENCY THAT IS REAL If an offer genuinely ends, show the end date in plain text ("Price rises on 1 December") and keep it true for everyone. A timer that restarts per visitor is the exact pattern behind the CMA's Emma Sleep case.

11 The sticky mobile bar

CHECKS 54, 55 & 56

Sixty lines that out-convert every chat widget you can buy, because it does the one thing a visitor on a phone actually wants: connect them to a human in one tap.

components/StickyCta.tsx

```
import { site } from '@lib/site'
import { lead } from '@lib/track'

export function StickyCta() {
  return (
    <div className="sticky-cta">
      <a href={`tel:${site.phone}`} onClick={lead.call}>Call</a>
      <a
        href={`https://wa.me/${site.whatsapp}`}
        onClick={lead.whatsapp}
        rel="noopener"
      >WhatsApp</a>
      <a href="/contact">Get a quote</a>
    </div>
  )
}
```

app/globals.css

```
.sticky-cta {
  position: fixed;
  inset: auto 0 0 0;
  z-index: 40;
  display: grid;
  grid-template-columns: repeat(3, 1fr);
  gap: 1px;
  padding-bottom: env(safe-area-inset-bottom); /* iPhone chin */
}

.sticky-cta a {
  padding: 1rem;
  text-align: center;
  min-height: 48px; /* comfortable tap target */
}

body { padding-bottom: 4rem } /* bar never covers the footer */

@media (min-width: 768px) {
  .sticky-cta { display: none }
  body { padding-bottom: 0 }
}
```

BEFORE YOU INSTALL A CHAT WIDGET Ask who is answering it. An unanswered bubble loses you leads, because the visitor tries it instead of calling, waits, and leaves. WhatsApp is a plain link with no script, no cookies and no consent implication, and local businesses genuinely answer it.

12 Fonts, images and motion

CHECKS 13 TO 15, 23

Three small things that decide whether you pass Core Web Vitals, which roughly half of all mobile sites currently do not.

app/layout.tsx (fonts belong on html, not body)

```
import { Playfair_Display, Manrope } from 'next/font/google'

const display = Playfair_Display({
  subsets: ['latin'], variable: '--font-display', display: 'swap',
})
const body = Manrope({
  subsets: ['latin'], variable: '--font-body', display: 'swap',
})

export default function RootLayout({ children }) {
  return (
    <html lang="en-GB" className={` ${display.variable} ${body.variable}`}>
      <body>{children}</body>
    </html>
  )
}
```

NEXT 16 WITH TAILWIND 4 Put the variables on body and half your components silently fall back to system fonts, because the utilities that reference them resolve higher up the tree. Tokens that reference a variable also need `@theme inline` rather than plain `@theme`.

Images: one priority, always a size

```
// the hero, and only the hero
<Image src={hero} alt="" fill priority sizes="100vw" />

// everything else
<Image
  src={job} width={800} height={600}
  alt="Rewired consumer unit in a Croydon semi"
  sizes="(min-width: 768px) 33vw, 100vw"
/>
```

app/globals.css

```
@media (prefers-reduced-motion: reduce) {
  *, *::before, *::after {
    animation-duration: 0.01ms !important;
    transition-duration: 0.01ms !important;
    scroll-behavior: auto !important;
  }
}
```

13 Prove it on the live domain

CHECKS 34, 35, 49, 53

Everything above is a claim until it answers on the production URL. Run these against the domain the client will actually use, not the preview deployment.

Eight commands, about two minutes

```
# 1. the sitemap exists and lists what you expect
curl -s https://www.clientsite.co.uk/sitemap.xml | grep -c "<url>"

# 2. robots points at the absolute sitemap URL
curl -s https://www.clientsite.co.uk/robots.txt

# 3. the security headers are genuinely being sent
curl -sI https://www.clientsite.co.uk \
  | grep -iE "strict-transport|x-content-type|referrer-policy"

# 4. the apex redirects to the canonical host (expect 308)
curl -sI https://clientsite.co.uk | head -1

# 5. email authentication is live: all three must return a record
dig +short TXT clientsite.co.uk | grep spf
dig +short TXT resend._domainkey.clientsite.co.uk
dig +short TXT _dmarc.clientsite.co.uk

# 6. an old URL still lands somewhere useful (expect 301)
curl -sI https://www.clientsite.co.uk/our-services | head -1

# 7. the purge cron refuses strangers (expect 401)
curl -s -o /dev/null -w "%{http_code}" https://www.clientsite.co.uk/api/cron/purge

# 8. no ad pixel in the page before consent (expect 0)
curl -s https://www.clientsite.co.uk | grep -c "connect.facebook.net"
```

TWO THAT CATCH PEOPLE OUT Read the real production domain from the platform rather than assuming `projectname.vercel.app`: names collide globally and get silently suffixed. And on a brand new project, check deployment protection is off, or every route 302s to a login screen that only you can pass, which looks exactly like a broken site to everybody else.

THEN THE HUMAN TEST Submit a real enquiry from your phone, on mobile data, with a real name containing an apostrophe. Confirm it arrives in the inbox, in the database, and in analytics as a conversion. Then delete the test row so the client's data starts clean.

THE NUMBERS

Environment and cost

Seven variables run the whole kit. Anything prefixed `NEXT_PUBLIC_` is compiled into the JavaScript bundle and served to every visitor, so it must never hold a secret.

VARIABLE	COMES FROM	SECRET	USED BY
DATABASE_URL	Neon or Supabase	Yes	The contact route
RESEND_API_KEY	Resend dashboard	Yes	The contact route
LEAD_INBOX	The client, in writing	Yes	The contact route
NEXT_PUBLIC_SITE_URL	lib/site.ts fallback	No	Metadata, sitemap
NEXT_PUBLIC_WHATSAPP	The client's number	No	Sticky bar, links
CRON_SECRET	A long random string	Yes	The purge cron
NEXT_PUBLIC_META_PIXEL_ID	Meta Events Manager	No	Ad events, after consent

And what the services behind them cost, checked October 2026.

SERVICE	JOB	START AT	UPGRADE WHEN
PAID			
Vercel Pro	Hosting	\$20/seat/mo	Day one. Hobby is non-commercial only
ICO data protection fee	Registration	£52/yr (tier 1)	Paid by the client, renewed yearly
FREE AT THE SCALE A LOCAL BUSINESS SITE RUNS AT			
Resend	Transactional email	Free, 3,000/mo	Past 3,000 emails, then \$20/mo
Neon or Supabase	Database	Free tier	Real volume, then \$19 to \$25/mo
Vercel Web Analytics	Traffic and events	Included on Pro	Past the included events
Cloudflare Turnstile	Spam, if honeypot fails	Free, no request cap	Only when spam gets through
Search Console	Indexing	Free	Never
Sentry Developer	Error tracking	Free, 5k errors/mo	When it becomes an application

\$20/mo

One Vercel Pro seat, covering every client site you host. The client pays their own ICO fee. Everything else in this kit is free at the scale a local business site operates at. Build the seat into the quote rather than finding it in month two.

Paste them in this order

Each step unblocks the next. Work down the column and nothing has to be revisited.

1

lib/site.ts

The canonical URL, name, phone and WhatsApp number. Everything else imports from here, so getting it wrong here gets it wrong everywhere.

2

Domains in Vercel

Add apex and www, pick the canonical one, let the platform issue the 308. Then the URL in step 1 is finally true.

3

next.config.ts

Security headers now, redirects as soon as you have the old site's URL list out of Search Console.

4

layout.tsx

metadataBase, title template, Open Graph, and the fonts on the html element. Then per-page metadata with canonicals.

5

sitemap.ts and robots.ts

Two files, five minutes, now that the canonical host is settled and cannot change under them.

6

JsonLd.tsx and lib/schema.ts

Render, then validate every page in the Rich Results Test before moving on. Fix what it flags now, not later.

7

Contact route, then the form

Route first so you can curl it, form second, with the age gate and optional questions if the service needs them. Connect the domain in Resend the same day: DNS propagation is not instant.

8

track.ts, consent, purge cron, StickyCta

Add PriceLine and PartnerBox if you sell. Then deploy, run the eight smoke tests, submit the sitemap in Search Console, and fire one real enquiry from your phone.

The template challenge: the second time you do this, fork it. Most of these eighteen files never change between clients. Turn them into your starter repo and this two-hour job becomes a twenty-minute one, permanently. That gap is your margin.

FINAL WORD

The code was never the moat. *The finishing is.*

Anyone can generate a good-looking site now. What still separates the people who get paid well from the people who compete on price is whether the thing actually works when it meets the real world: indexed, fast, legal, honest, measured, and connected to a human who can answer.

That is what these eighteen files do. They are not clever. They are the boring, specific, hard-to-remember details that clients cannot see and cannot do, and that almost nobody bothers to get right.

Paste them into your next build. Then turn them into your starter repo, and never do this from memory again.

Want this running across every site you ship?

If you would rather hand the launch process to somebody else, or you want the whole kit turned into a starter template and a repeatable handover for your agency, book a call with B2BLCC and we will look at where your current process leaks.

b2blcc.com · sam@b2blcc.com