Zephortech — Software Requirements Specification (SRS) v1

Muhammad Junaid (for Zephortech) **Project:** Zephortech Corporate Website (Professional, CMS-driven, SEO-optimized)

---

1. Project Overview

Zephortech is an IT services company offering a broad range of services (web & mobile apps, AI/ML agents, SaaS, eCommerce, Shopify/WordPress/WooCommerce integrations, digital marketing/SEO, cloud & DevOps). The goal of this project is to build a professional, production-ready corporate website that:

- Presents Zephortech services and portfolio in a polished way
- Generates and converts leads (quote requests / contact forms)
- Is fully maintainable by non-technical staff via a CMS
- Is SEO-ready and fast on mobile & desktop
- Is deployable to production with CI/CD and clear hosting plans

Note: Domain and Hostinger hosting have already been purchased by Zephortech.

---

2. Goals and Success Criteria

Primary Goals

- Credible corporate presence that converts visitors to leads
- Clear service pages for offerings: Web, Mobile, AI agents, SaaS, eCommerce (Shopify, WooCommerce), WordPress, Digital Marketing/SEO, Cloud & DevOps, and more as per reference sites
- **Reference Websites Used for Requirement Analysis:**
  https://fronxsolutions.com
  https://extrasol.co.uk
  These sites were analyzed to benchmark service offerings, content structure, and corporate presentation quality. Zephortech's site will follow similar professionalism while maintaining its own unique branding.
- CMS-driven content management for Services, Portfolio, Blog, Team, Testimonials
- Best-in-class SEO and analytics setup

Success Criteria

- Page Speed: Lighthouse Performance score $\geq$ 90 on desktop; mobile score $\geq$ 70 (realistic target)
- Time to First Byte (TTFB) low via SSR/Edge caching

- Contact form conversion tracking (GA4) and at least first 10 leads within first month (marketing dependent)
- Admins can add/edit Services/Portfolio/Blog within the CMS without developer help

---

3. Scope / Out of Scope

In scope (MVP for launch):

- Full frontend built in Next.js + TypeScript + Tailwind CSS
- Headless CMS (Strapi recommended) with PostgreSQL (production) or SQLite for dev
- Portfolio / Case studies, Services, Blog, About, Contact, Request a Quote
- Responsive design (mobile-first), accessibility basics
- SEO basics: meta tags, OG tags, sitemap, robots.txt, structured data (Organization + Breadcrumbs)
- Contact form with secure backend processing and spam protection (reCAPTCHA or honeypot)
- Analytics (GA4) + Google Search Console setup guidance
- Staging and production environments, CI/CD pipeline (GitHub Actions)
- Deployment-ready code and admin user guide

Out of scope (phase 1):

- Full multilingual rollout (reserve for phase 2)
- Client portal or paid user accounts (phase 2)
- Advanced marketing automations (CRM integration beyond email notifications) unless requested

---

4. Functional Requirements

4.1 Public Pages & Features

- **Home**: Hero, services summary, CTA, featured case study, client logos, testimonials, blog highlights
- **About**: Company story, mission, team previews, values
- **Services Overview**: List of all services with short descriptions
- **Service Detail Pages**: Dedicated page per service (Web Dev, Mobile Apps, AI Agents, SaaS, Shopify, WordPress/WooCommerce, eCommerce, SEO & Digital Marketing, Cloud & DevOps, etc.)
- **Portfolio / Case Studies**: Filterable list, each with images, problem, solution, tech used, outcomes
- **Blog**: Article list, categories, tags, article detail pages with reading time, OG data
- **Contact**: Contact form, map, address, phone, quick contact CTA
- **Request a Quote**: Multi-field form (name, company, email, service interest, budget range, timeline, description, file upload optional)
- **Technologies Page**: Technologies & tools grid (build credibility)
- **Privacy Policy / Terms** pages

4.2 Admin / CMS

- CRUD for Services, Portfolio, Blog Posts, Team Members, Testimonials, Client Logos
- Manage navigation/menu, site meta (title template, default description), hero banners

- View & export contact/quote submissions
- Role-based users (Admin, Editor)

4.3 Integrations

- **Email notifications** for form submissions (via SMTP provider or transactional email like SendGrid)
- **Google Analytics 4** + GTM optional
- **reCAPTCHA v3 or honeypot** for spam protection
- **Cloudinary (or similar)** for image optimization and CDN
- **Sitemap.xml generation** and robots.txt

---

5. Non-Functional Requirements

- **Performance**: SSR for critical pages (Next.js) + caching at Edge / CDN
- **SEO**: dynamic meta tags, server-rendered content for crawlers, structured data
- **Security**: HTTPS mandatory, sanitize inputs, rate limiting on forms, secure admin routes
- **Accessibility**: basic WCAG AA practices (alt tags, color contrast, keyboard focus)
- **Scalability**: decouple frontend and CMS; CMS can be hosted independently
- **Maintainability**: clean code, componentized UI, TypeScript types, documentation
- **Backup**: automated DB backups for CMS (daily/weekly policy)

---

6. Recommended Technology Stack (finalized)

- **Frontend**: Next.js 14+ (app router if appropriate) + TypeScript + Tailwind CSS
- **Styling/UI**: Tailwind + headless UI components; Figma for design system
- **CMS**: Strapi (latest v5+) or Sanity (if prefer SaaS) — *Strapi recommended for self-host control*
- **Database**: PostgreSQL (production); SQLite for development
- **Image Hosting / CDN**: Cloudinary or Imgix; fronted by CDN (Cloudflare)
- **Hosting / Deployment**:
- Frontend: Vercel (preferred) or Cloudflare Pages
- Backend/CMS: Render / Railway / DigitalOcean App Platform (or Hostinger VPS if you have Node support)
- Domain & Email: Hostinger (DNS & company email) — point domain to Vercel/Render via DNS records
- **Auth**: Strapi built-in auth for CMS, NextAuth for any site auth needs
- **CI/CD**: GitHub Actions → automatic deploy to Vercel/Render
- **Monitoring**: Sentry for runtime errors; UptimeRobot for availability

  Rationale: Next.js + Strapi + PostgreSQL offers a modern, production-grade stack with SSR for SEO, a friendly CMS for editors, and a robust relational DB for structured content.

---

7. Page-level Content Model (CMS Schema)

Collections / Content Types

- **Service**: title, slug, summary, body (rich text), features (list), hero image, icons, tags, seoTitle, seoDesc, order
- **Portfolio**: title, slug, client, industry, summary, problem, solution, techStack (relation), images (gallery), date, outcome metrics
- **BlogPost**: title, slug, author (relation), category, tags, content, excerpt, featuredImage, publishedAt, seo meta
- **TeamMember**: name, role, bio, photo, social links
- **Testimonial**: clientName, role, company, quote, logo (optional)
- **Technology**: name, logo, category (frontend/backend/cloud), description
- **ContactSubmission**: name, email, company, message, serviceInterest, budgetRange, createdAt
- **SiteSettings** (singleton): siteTitle, defaultMeta, socialLinks, footerText, contactEmail

---

8. SEO & Content Strategy (High Level)

- Create target keywords for each service page (shortlist 3–5 primary keywords per service)
- Blog strategy: publish 4–8 posts in first 3 months covering case studies, industry trends, how-tos
- Implement structured data: Organization + WebSite + BreadcrumbList + Article schema
- Ensure fast mobile-first design and AMP-like speed optimizations where necessary
- Set up Google Search Console and submit sitemap after launch

---

9. Deployment & Hosting Plan (practical)

1. **Frontend deployment**: Push Next.js app to GitHub → Vercel connects to repo → automatic builds on push to `main`.
2. **CMS deployment**: Host Strapi on Render / DigitalOcean (Docker recommended). Use managed Postgres add-on.
3. **DNS & Domain**: Use Hostinger control panel to set DNS A/CNAME records. Point apex (zephortech.com) to Vercel and CMS subdomain (cms.zephortech.com) to Render IP/CNAME.
4. **Emails**: Use Hostinger email or third-party transactional provider (SendGrid / Mailgun) for outgoing forms.
5. **SSL**: Provided by Vercel/Render automatically; ensure Hostinger DNS points correctly.

---

10. Wireframes & Design

Deliverables in design phase:

- Desktop & Mobile wireframes for: Home, Services (list + detail), Portfolio, Blog index, Blog post, Contact, Admin screens (CMS content examples)

- Design system: colors, typography, spacing, UI components library

---

11. Development Roadmap & Milestones (detailed)

Sprint 0 — Setup & Planning (1 week)

- Repo + branch strategy, initial Next.js scaffold, Strapi skeleton, DB provisioning, CI/CD baseline

Sprint 1 — Core Pages & Components (2–3 weeks)

- Implement Home, About, Services list & detail skeleton; responsive components; SEO meta implementation

Sprint 2 — CMS & Portfolio (2 weeks)

- Model content types in Strapi, connect frontend to CMS endpoints, implement Portfolio pages and case studies

Sprint 3 — Blog & Contact (1–2 weeks)

- Blog CRUD, comments (if required later), Contact form backend, email integration

Sprint 4 — QA & Optimization (1 week)

- Accessibility checks, Lighthouse optimization, SEO validation, cross-browser testing

Sprint 5 — Staging & Launch (1 week)

- Final content population, DNS switch, analytics setup, monitoring

Total estimated: 8–10 weeks (depends on content availability and review cycles)

---

12. Deliverables

- Full source code (frontend + CMS configuration)
- Production deployment URLs
- Figma design files & style guide
- Admin / editor manual (how to add services, portfolio, blog posts)
- Post-launch 30-day support plan (bug fixes + minor content tweaks)

---

13. Risks & Mitigations

- **Hostinger shared hosting limitations**: If Hostinger is shared hosting without Node support, host CMS on Render/DigitalOcean and point domain via DNS. Use Hostinger only for email & DNS. (Mitigation: provision a VPS on Hostinger if preferred.)
- **Content delays**: Client must provide final content (case studies, logos, copy). Use placeholder content to continue dev.
- **Scope creep**: Enforce a change request process with impact & cost estimates.