# Yadnus Consultant — Website Development Guide

**Prepared for:** Israelite

## Executive Summary

This guide walks you step-by-step through designing, building, and launching a professional website for **Yadnus Consultant** — a town planning and construction firm that will host live webinars, publish past webinar videos, maintain a blog, showcase portfolio projects, accept inquiries via forms, and integrate social media (WhatsApp, Twitter, Instagram). The plan focuses on a pragmatic, secure, and maintainable stack with clear implementation steps you can follow or hand to a developer.

## Goals & Requirements (MVP)

1. Professional public-facing website using the name **Yadnus Consultant**.
2. Pages: Home, About, Services, Projects (portfolio), Blog, Webinars (live + past recordings), Contact, Admin Dashboard.
3. Webinar capabilities: live webinar embed (Zoom/YouTube), webinar signup form, archive of recorded webinars (video posts).
4. Blog system with categories, tags, search, and SEO-friendly URLs.
5. Social media integration: WhatsApp share, Twitter share, Instagram profile/linking.
6. Multiple visitor forms: general contact, webinar signup, project inquiry (with optional file upload), newsletter subscription.
7. Image-rich design: high-quality photos that reflect town planning, construction, and community spaces.
8. Admin area to create/edit posts, upload media, schedule webinars, and view form submissions.
9. Authentication for admin (JWT or session-based) and role control for editors.
10. Responsive, accessible, and SEO-optimized site.

## Suggested Tech Stack

**Frontend** - React with **Next.js** (SSR/SSG for SEO and fast page loads) - Tailwind CSS for rapid, consistent styling - React Query / SWR for data fetching and caching

**Backend / API** - Node.js + Express (or Next.js API routes) for REST endpoints - Authentication: flask-jwt-extended if you prefer Python/Flask; otherwise jsonwebtoken for Node

**Database & Media Storage** - PostgreSQL for structured data (posts, users, submissions) - Cloud object storage for images/videos: AWS S3, DigitalOcean Spaces, or Cloudinary for images - Host videos on YouTube (unlisted) or Vimeo for streaming + low bandwidth or use S3 + CloudFront if you prefer self-hosting

**CMS / Admin** - Option A: Build a lightweight custom admin panel (React) connected to your API - Option B: Use a headless CMS (Strapi, Sanity, or Contentful) to manage blog, webinars, and projects (faster)

**Other** - CI/CD: GitHub Actions → deploy to Vercel (frontend) and Render / DigitalOcean App Platform (backend) or deploy both to Vercel (if using Next.js API routes) - Analytics: Google Analytics or Plausible/Matomo - Email: transactional (SendGrid / Mailgun) for webinar confirmations and newsletter

## Sitemap (High-level)

/ (Home)  
/about (About Yadnus Consultant)  
/services (Town planning, Construction management, Consultancy)  
/projects (Portfolio list)  
/projects/:id (Project detail)  
/blog (All blog posts)  
/blog/:slug (Post detail)  
/webinars (Upcoming + Past webinars list)  
/webinars/:id (Webinar detail + video)  
/contact (Contact & forms)  
/admin (Admin dashboard - protected)  
/admin/posts (Manage blog posts)  
/admin/projects (Manage portfolio)  
/admin/webinars (Schedule/upload webinars)  
/admin/submissions (Form submissions)

## Step-by-step Development Plan

The plan is split into phases: Discovery, Design, Development, QA, Deployment, and Post-launch.

### Phase A — Discovery & Planning (1–3 days)

1. Gather assets and content from the owner (logo, brand colors, hero images, company description, services list).
2. Define user journeys: visitor, webinar registrant, client requesting a quote, admin.
3. Create a prioritized feature list (MVP vs. Later).
4. Prepare a content map and SEO keywords for the main pages.

**Deliverables:** content inventory, sitemap, prioritized feature list.

### Phase B — Design & Prototyping (2–6 days)

1. Create moodboard with 8–12 images reflecting town planning and construction (Unsplash / Pexels suggestions). Label images like: hero-planning.jpg, construction-site.jpg, community-park.jpg.
2. Wireframes: Home, Projects, Blog, Webinar detail, Admin list pages (low-fidelity).
3. High-fidelity mockups: Desktop and mobile versions of Home and Webinar page.
4. Export a style guide: font choices (e.g., Inter, Space Grotesk), color palette, button styles, card styles.

**Deliverables:** moodboard, wireframes, 2–3 high-fidelity mockups, style guide.

### Phase C — Project Setup & Core Implementation (5–14 days)

#### 1. Project scaffold

* Initialize Git repo, create README, set up branches: main, dev.
* Create Next.js app: npx create-next-app@latest yadnus-site --typescript
* Install deps: Tailwind CSS, axios, react-hook-form, react-player (for video embeds), and date libs.

#### 2. Layout & UI components

* Build global layout (Header, Footer). Header contains nav + social icons (WhatsApp, Twitter, Instagram).
* Implement responsive grid system using Tailwind.
* Create reusable components: Card, Hero, ProjectList, BlogList, VideoPlayer, FormField, Modal.

#### 3. Static pages & routes

* Implement Home, About, Services, Projects listing and detail pages, Blog listing and detail.
* Use Next.js getStaticProps / getStaticPaths for blog/project pages (fast, SEO-friendly).

#### 4. Forms

* Use react-hook-form on frontend.
* Form types:
  + **Contact Form**: name, email, phone, projectType, message, fileUpload (optional)
  + **Webinar Signup**: name, email, company, jobTitle, howHearAboutUs
  + **Project Inquiry**: name, email, budgetRange, description, upload attachments
  + **Newsletter**: email only
* Add client-side validation and reCAPTCHA (optional) to avoid spam.

#### 5. Backend & APIs

* If using Next.js API routes: create endpoints under /pages/api/\* to handle forms, admin, and file uploads.
* Example endpoints:
  + POST /api/contact — saves contact and triggers email
  + POST /api/webinar-signup — stores signup and sends confirmation
  + GET /api/posts — returns blog posts
  + POST /api/upload — accepts image uploads (admin only)
* Persist data to PostgreSQL with an ORM (Prisma recommended) or use a headless CMS to skip backend code.

### Phase D — Webinar & Video Integration (2–5 days)

**Options for live webinars** - **Embed YouTube Live**: easiest; create a YouTube Live stream, embed the iframe on the webinar page. Use YouTube’s API to dynamically show live status. - **Zoom + YouTube**: run Zoom webinar and stream to YouTube; embed YouTube stream. - **Third-party webinar platforms**: Demio, WebinarJam (paid). Not required for MVP.

**Recording & hosting past webinars** - Host recordings on YouTube (unlisted) and embed on the webinars/:id page. - Alternatively, upload MP4s to S3 and stream with react-player.

**Admin workflow** - Admin uploads webinar recording/YouTube link, adds title/description/tags, and sets date, speakers. - Frontend shows upcoming webinars and past recordings with thumbnails.

### Phase E — Social Media Sharing & Integrations

**WhatsApp share** - Use a share link: https://wa.me/?text=${encodeURIComponent(textAndUrl)} - Mobile-first; opens WhatsApp app if available.

**Twitter share** - Intent link: https://twitter.com/intent/tweet?text=${encodeURIComponent(text)}&url=${url}

**Instagram** - Instagram does not support direct web posting via URL. Provide: - A clear link to company Instagram profile - Share buttons using the Web Share API on mobile which allows users to share the page to Instagram if supported.

**Open Graph / Twitter Cards** - Add meta tags to each page for rich previews: - og:title, og:description, og:image, og:url - twitter:card, twitter:image, etc.

### Phase F — Media & Images

1. Use high-quality, license-free images (Unsplash, Pexels) for placeholders. Keep originals in a design-assets/ folder.
2. Use **Cloudinary** or S3 + an image CDN to serve optimized images & create responsive srcset.
3. Create thumbnails for videos and projects.
4. For hero images and project photos, pick imagery that conveys:
   * Urban planning maps and sketches
   * Construction site / cranes / teams at work
   * Community parks and finished developments

### Phase G — Admin Dashboard & Authentication (3–7 days)

1. Implement Admin Authentication:
   * Option A: JWT (stateless). Create /api/auth/login that returns an access token.
   * Option B: Session cookies (easier with server-side rendered admin panel).
2. Admin features:
   * Manage blog posts (Create, Edit, Publish, Unpublish)
   * Manage projects (CRUD)
   * Upload webinar videos and schedule live events
   * View and export form submissions
   * Manage newsletter subscribers and send broadcasts (or integrate SendGrid)
3. Protect admin routes with authentication and role checks.

### Phase H — Testing & QA (2–5 days)

* Unit tests for critical API routes.
* Integration tests for form submission flows.
* Manual QA for responsive layout on mobile, tablet, and desktop.
* Accessibility tests (axe, Lighthouse) and performance tests (Lighthouse).

### Phase I — Deployment & Monitoring (1–3 days)

* Deploy frontend to Vercel for Next.js (auto-deploy on push to main branch).
* Deploy backend (if separate) to Render, Heroku, or DigitalOcean App Platform.
* Configure environment variables securely (DATABASE\_URL, S3 keys, JWT secret).
* Set up automated backups for your DB and media storage (daily snapshots).
* Add monitoring: Sentry for error tracking, and Google Analytics for traffic.

### Phase J — Launch Checklist

* Content finalized (copy, images, speaker bios)
* SEO meta tags in place
* Forms tested & email confirmations working
* Admin tested and secure
* Backups enabled
* Analytics + error tracking enabled

## Suggested Database Schema (Simplified)

**blog\_posts** - id (UUID) - title - slug - content (rich text / HTML) - excerpt - image\_url - tags (array) - category - published (boolean) - created\_at, updated\_at

**projects** - id (UUID) - title - description - location - category - images (array) - completed\_at - created\_at

**webinars** - id (UUID) - title - description - date - speakers (JSON) - video\_url (YouTube or S3 link) - thumbnail\_url - created\_at

**submissions** (contact + project inquiries) - id (UUID) - type (enum: contact, project, webinar) - payload (JSON) - attachments (array) - created\_at

**users** (admins) - id - username - password\_hash - role

## Example API Endpoints (REST)

* GET /api/posts — list posts
* GET /api/posts/:slug — single post
* POST /api/posts — create (admin)
* PUT /api/posts/:id — update (admin)
* DELETE /api/posts/:id — delete (admin)
* POST /api/upload — admin image upload
* POST /api/contact — public contact form
* POST /api/webinar-signup — signup
* POST /api/newsletter — subscribe

## UX Notes & Copy Tips

* Hero tagline example: *“Designing sustainable cities, one neighborhood at a time — Yadnus Consultant”*
* CTA examples: *Schedule a Consultation*, *Join Our Next Webinar*, *View Past Projects*.
* Keep forms short and focused; ask for follow-up details only when necessary.

## Image & Asset Recommendations

* Unsplash searches: “urban planning”, “construction site”, “community park”, “architect blueprint”.
* Use 16:9 images for hero sections and 4:3 or square for grid cards.
* Keep images under 200KB (use CDN transformations) for faster loading.

## Timeline & Team Suggestions (Example)

* Solo developer: 6–10 weeks (design, build, test, deploy).
* Small team (designer + dev): 3–5 weeks.
* If using a headless CMS + YouTube for video hosting: accelerate by 30–40%.

## Maintenance & Next Steps

* Plan monthly content publishing schedule (blogs + webinars).
* Regular security updates & dependency scans.
* Consider paid video/webinar platforms as audience grows.

## Visual Flow Diagram (Request → Store → View)

[Visitor] --(visit page)--> [Next.js Frontend]  
 | |  
 |--(submit form)--------> [API Endpoint /api/\*] --(store)--> [Postgres]  
 | |---> [S3/Cloudinary for images]  
 |--(view video)---------> [Video embed player] --(source)--> [YouTube or S3]

## Final Notes

* Keep the first release focused on the MVP features (home, projects, blog, webinars archive, forms, admin). Add advanced features (live webinar platform, full video hosting, multi-language) after you have stable traffic and feedback.
* If you want, I can:
  + provide a Next.js starter repo template for Yadnus Consultant,
  + scaffold the database schema with Prisma,
  + or write the Admin login and file upload API endpoints in your preferred backend language.

*End of guide.*