Technical Specification Document (Tech Spec)

Project: BuroHero (MVP) **Version:** 1.3 (Mock Payment Edition) **Based on:** PRD v1.6 **Stack:** Next.js 14 (App Router), TypeScript, Tailwind CSS, Supabase, DeepSeek API.

1. System Architecture Overview

1.1 High-Level Diagram

[User Browser] <-> [Vercel Edge Network (CDN)] <-> [Next.js Serverless Functions] ^ | | v [Supabase DB] [DeepSeek API (AI)]

1.2 Core Tech Stack

- **Frontend:** Next.js 14 (App Router).
- **UI Library:** Tailwind CSS + `shadcn/ui` (Radix Primitives).
- **Language:** TypeScript (Strict).
- **I18n:** `next-intl` (Middleware-based routing for 10 languages).
- **Database:** Supabase (PostgreSQL) - for storing order status and temporary content.
- **AI Provider:** DeepSeek V3 (via Vercel AI SDK).
- **Payment Simulation:** Custom "Mock Pay" API route (for Stage 1).
- **Hosting:** Vercel (Free Tier).

2. Frontend Architecture

2.1 Routing Structure (FileSystem)

Dynamic routing is key for pSEO and localization.

```
/src
  /app
    /[locale]                 // e.g., "es", "en", "uk"
      /page.tsx               // Landing Page
      /layout.tsx             // Main layout (Navbar with Lang Switcher)
      /[category]             // e.g., "baja", "reclamar"
        /[company]            // e.g., "vodafone", "zara"
          /page.tsx           // THE GENERATOR (Main Logic)
      /checkout
        /result
          /page.tsx           // Success/Download page (checks payment status)
```

2.2 Dual-View Component (`GeneratorView.tsx`)

- **State:** Uses `useChat` hook (Vercel AI SDK) to handle streaming JSON.

- Logic:
  - Parses incoming stream chunks. DeepSeek returns JSON string `{ "es": "...", "native": "..." }`. We must parse partial JSON on the fly using `ai/streams`.
  - Blurred State: If `isPaid === false`, apply `filter: blur(5px)` and `user-select: none` to the Left Pane (ES Document).
  - Payment Trigger: Button "Unlock Document" triggers `handlePayment()`.

2.3 Client-Side PDF (`PdfGenerator.ts`)

We do **not** use server-side generation to save costs and ensure privacy.

- Lib: `jspdf + jspdf-autotable`.
- Fonts: Need to load `TimesNewRoman-Regular.ttf` as Base64 string to support Cyrillic/Ukrainian characters in the "Translation" PDF.
- Action: Generates two files:
  1. `Legal_[Company].pdf` (Spanish).
  2. `Translation_[Lang].pdf` (Native).

3. Backend & API Routes

3.1 AI Generation Endpoint

- Route: `POST /api/generate`
- Flow:
  1. Receives `user_data + locale`.
  2. Builds System Prompt based on the specific Use Case (from `BuroHero_Prompts_and_Fields.md`).
  3. Calls DeepSeek API (`deepseek-chat`) with `response_format: { type: "json_object" }`.
  4. Streams response to client.

3.2 Mock Payment Endpoint (For MVP Testing)

- Route: `POST /api/orders/mock-pay`
- Condition: Only works if `NEXT_PUBLIC_ENABLE_TEST_PAYMENTS=true`.
- Input: `{ orderId: string }`
- Action:
  1. Updates Supabase: `UPDATE orders SET status = 'paid' WHERE id = orderId`.
  2. Returns: `{ success: true }`.
- Frontend Behavior: Upon success, redirect user to `/checkout/result?orderId=...`.

4. Database Schema (Supabase)

We need a simple table to gate the content.

Table: `orders`

- `id` (UUID, Primary Key) — Generated on client or server.
- `status` (Enum: 'pending', 'paid') — Default 'pending'.
- `amount` (Integer) — 299.
- `created_at` (Timestamp).
- `content_snapshot` (JSONB) — **Encrypted/Stored text.**
  - *Why?* When user pays and comes back, we need to show the text again to generate PDF. We don't want to re-generate it with AI (costs money).
  - *Privacy:* Auto-delete row after 24h via Supabase Extension `pg_cron` or manual policy.

5. Security & Privacy

- **Rate Limiting:** Use `@upstash/ratelimit` (Free tier) to prevent AI API abuse. Limit: 5 generations / hour / IP.
- **Content Security Policy (CSP):** Strict rules in `next.config.js`.
- **Data Retention:** We explicitly promise "24h deletion". Ensure Supabase RLS (Row Level Security) allows `SELECT` only by `orderId` (UUID is the secret key).

6. Infrastructure & Deployment Setup

6.1 Hosting: Vercel

- **Why:** Zero config, native Next.js support, edge caching for pSEO pages.
- **Setup:**
  1. Push code to GitHub.
  2. Import to Vercel.
  3. Add Env Vars.

6.2 Environment Variables (`.env.local`)

```
# App Config
NEXT_PUBLIC_APP_URL="http://localhost:3000"
NEXT_PUBLIC_ENABLE_TEST_PAYMENTS="true"  # <--- Mock Button Switch

# DeepSeek (AI)
DEEPSEEK_API_KEY="sk-..."

# Supabase (DB)
NEXT_PUBLIC_SUPABASE_URL="[https://....supabase.co](https://....supabase.co)"
NEXT_PUBLIC_SUPABASE_ANON_KEY="eyJ..."
SUPABASE_SERVICE_ROLE_KEY="eyJ..." # For admin updates (mock pay)
```

```
# Stripe (Reserved for later)
STRIPE_SECRET_KEY=""
```

6.3 Domain Strategy

- **Provider:** Cloudflare (cheaper renewals) or Vercel (easier setup).