ChatRAG Knowledge Base

ChatRAG Knowledge Base for Chatbot Assistants

Last updated: 2025-10-31
Maintainer: Carlos Marcial — x.com/carlosmarcialt

Purpose

This document centralizes the product knowledge your ChatRAG-powered chatbots need to answer questions about the platform, its pricing, features, setup process, and support options. Use it as the authoritative reference for Retrieval-Augmented Generation (RAG) assistants deployed on the ChatRAG landing page or related channels.

---

1. Product Overview

- **Product name:** ChatRAG — Production-ready AI chatbot platform.
- **Tagline:** Build unlimited RAG-powered AI chatbots in minutes.
- **Value proposition:** Pay once, own the full Next.js codebase, and deploy white-label chatbots with enterprise-grade features, visual configuration, and 200+ AI models.
- **Audience snapshot:** 250+ AI developers, founders, and agencies already use ChatRAG to ship client-facing AI assistants quickly.

1.1 Key Differentiators

- Visual configuration UI—no code edits required for day-to-day operations.
- Dual WhatsApp providers plus responsive web app and embeddable widget.
- HNSW-optimized Supabase vector search delivering 15–28× faster retrieval (<50 ms semantically rich queries).
- Commercial license with unlimited chatbot deployments for you and your clients.
- Continuous updates: latest flagship models (OpenAI GPT-4.1 & 4.1 Mini, Anthropic Claude Sonnet 4.5, Gemini 2.5 Flash, DeepSeek V3.1) and AI SDK 5 foundation.

---

2. Pricing & Licensing

License: Commercial. Pay once and deploy unlimited self-hosted chatbots for you or your clients. Full source code included.

Plan	Price (USD)	Includes	Excludes
ChatRAG Starter	$199 (launch special, down from $299)	Complete Next.js RAG boilerplate; LlamaCloud + OpenAI + Supabase vector DB pipeline; Stripe & Polar monetization; 200+ AI models via OpenRouter; Zapier MCP-ready; authentication & user management; production UI with Tailwind & shadcn	Discord community, lifetime updates
ChatRAG Complete	$269 (down from $369)	Everything in Starter **plus** Discord community access, lifetime updates (updated as recently as “2 days ago”)	—

Payment links route through `/api/checkout` for one-time purchases. Highlight to customers that there are **no recurring license fees**—only infrastructure/API usage costs managed directly with providers (OpenAI, Supabase, etc.).

---

3. Target Customers & Use Cases

- **Indie & startup developers:** Ship polished AI chat apps without scaffolding from scratch.
- **AI agencies & consultants:** Deliver custom-branded assistants for clients with recurring service retainers.
- **SaaS founders:** Embed AI copilots inside existing platforms to boost engagement.
- **Internal product teams:** Build knowledge assistants, support bots, or operations copilots that run on internal datasets.
- **Educators & enterprises:** Launch white-label training assistants with secure, multi-tenant isolation.

Typical use cases include knowledge-base Q&A, customer support bots, product onboarding guides, research copilots, agency deliverables, and WhatsApp concierge assistants.

---

4. Core Capabilities

4.1 AI & Model Support

- 100+ large language models via OpenRouter and direct APIs (latest OpenAI GPT-4.1 & GPT-4.1 Mini, Anthropic Claude Sonnet 4.5 and Claude Opus 4.1, Google Gemini 2.5 Flash & Flash Lite, DeepSeek V3.1, Meta Llama 4 Maverick, Venice: Uncensored, and more).
- Flexible deployment across proprietary, commercial, and open-source/uncensored models so teams can tailor compliance, cost, and tone.
- Real-time response streaming, tool/function calling, voice input/output, MCP (Model Context Protocol) integrations, and Artifacts for executable results.

Featured defaults from the ChatRAG dropdown (October 2025): - `GPT-4.1 Mini` — fast, cost-effective general assistant; ideal baseline for most workspaces.
- `GPT-4.1` — full-quality flagship OpenAI model with strong reasoning and tool use.
- `Claude Sonnet 4.5` — balanced Anthropic model with updated constitutional guardrails.
- `Claude Opus 4.1` — highest tier Claude with extended context for complex workflows.
- `Gemini 2.5 Flash Lite` and `Gemini 2.5 Flash` — Google's production-ready multimodal family optimized for latency vs. depth.
- `Llama 4 Maverick` — Meta's latest open-source flagship for teams that need self-hosting.
- `Venice: Uncensored` — community-requested uncensored model for creative or adult-tolerant use cases (ensure compliance with local policy).
- `DeepSeek V3.1` — next-generation DeepSeek release with competitive long-context reasoning.

4.2 RAG Pipeline

- Document ingestion through LlamaCloud for intelligent parsing, chunking, and metadata extraction.
- OpenAI embeddings (1536-dimensional `text-embedding-3-small` recommended).
- Supabase PostgreSQL with pgvector + HNSW indexes (`m=64`, `ef=200`) for 15–28× faster semantic search (<50 ms typical query time).
- Supported file types: PDF, DOCX, TXT, HTML, RTF, EPUB.
- Auto chunk overlap configuration, adaptive retrieval, and multi-pass search (`RAG_ADAPTIVE_RETRIEVAL`, `RAG_MULTI_PASS`, `RAG_FINAL_RESULT_COUNT`).

4.3 Configuration & Management

- Visual Config UI (`npm run config`, http://localhost:3333) to manage API keys, branding, feature toggles, RAG system prompts, admin users, and WhatsApp settings.
- Main app (`npm run dev`, http://localhost:3000) exposes ChatGPT/Claude-style interface with dark mode, chat history, folders, document dashboard, and admin utilities.
- Config UI writes to `.env.local`; restart dev server after saving to load the latest settings.

4.4 Communication Channels

- Responsive web app (desktop & mobile).
- Embeddable widget for any site (`test-widget.html` example).
- WhatsApp integration with dual-provider choice (Koyeb or Fly.io); QR auth, multi-device sync, auto reconnection, webhook tracking.
- API access for programmatic use and third-party integrations.

4.5 Monetization & Authentication

- Stripe subscriptions and one-time payments ready out of the box.
- Polar integration as an alternative payment provider.
- Supabase Auth with email/password, OAuth (GitHub, Google), and magic links.
- Admin dashboard for managing users, subscriptions, and feature access.

4.6 Media & Pro Features

- Multi-modal generation is toggle-controlled in the Config UI (`NEXT_PUBLIC_IMAGE_GENERATION_ENABLED`, `NEXT_PUBLIC_VIDEO_GENERATION_ENABLED`, `NEXT_PUBLIC_3D_GENERATION_ENABLED`). Enable the switches in **Features → Media** and supply provider keys.
- Default provider is Fal.ai (`FAL_API_KEY`, `FAL_IMAGE_MODEL`, `FAL_VIDEO_TEXT_MODEL`, `FAL_3D_MODEL`). Flip `USE_REPLICATE_PROVIDER=true` to route image/video/3D calls through Replicate (`REPLICATE_API_TOKEN`, `REPLICATE_IMAGE_MODEL`, `REPLICATE_VIDEO_MODEL`, `REPLICATE_3D_MODEL`). OpenAI (`OPENAI_API_KEY`, `OPENAI_IMAGE_MODEL`) remains available for image generation.
- Advanced controls let you tune resolution, frame duration, and mesh simplification directly from the dashboard; generated artifacts are automatically written to Supabase storage buckets (`chat-images`, `chat-videos`, `3d-models`).
- Web search can blend with RAG when `NEXT_PUBLIC_WEB_SEARCH_ENABLED=true`; combine with `EXA_API_KEY` to activate Exa's real-time search results in both web chat and WhatsApp (when `WHATSAPP_ENABLE_WEB_SEARCH=true`).
- ElevenLabs or OpenAI power speech features. Pick a provider in **Audio → Text-to-Speech**, then set `ELEVENLABS_API_KEY`/`ELEVENLABS_VOICE_ID` or rely on the OpenAI key already stored.
- Shareable chat links, manual knowledge updates ("Add to knowledge base"), Artifacts visualization, analytics events, and 14+ localized UI languages are all configurable from the same dashboard.

4.7 Configuration Dashboard Highlights

- **Feature Toggles:** Switch authentication, web search, multi-modal generators, MCP tools list, embed widget, and incognito chat modes on/off without editing `.env`.
- **AI Model Catalogs:** Manage OpenAI and OpenRouter model lists, mark defaults, and flag models as free, open-source, uncensored, or reasoning-capable. The UI can auto-fetch metadata from OpenRouter and keeps JSON in

sync with `NEXT_PUBLIC_OPENAI_MODELS`/`NEXT_PUBLIC_OPENROUTER_MODELS`.
- **Exa Web Search:** The **AI Models → Exa** panel stores `EXA_API_KEY`. Once populated, Exa powers `/api/tools/web-search` and acts as the default live search provider when users toggle the globe icon.
- **MCP System:** Enable MCP via `NEXT_PUBLIC_MCP_SYSTEM_ENABLED` (with optional `NEXT_PUBLIC_MCP_TOOLS_LIST_ENABLED` to expose the tools modal). Built-in Zapier MCP is ready—add `MCP_ZAPIER_ENDPOINT` and the dashboard shows health metrics, tool counts, and per-surface enablement (web app, embed, WhatsApp). You can register additional MCP servers with bearer/API key auth, monitor status, and run discovery from the UI.
- **Branding & UX:** Update logos, color palettes, welcome carousels, gradient themes, suggestions, and chat input styles. Translation helpers auto-localize welcome messages and prompts.
- **Document Processing:** Upload datasets with LlamaCloud parsing, monitor processing queues, reset embeddings, or translate prompts from a single tab.
- **Admin & Access Control:** Manage admin emails, configure Stripe/Polar, adjust folder defaults, and tweak saved chat title generation (model, temperature, format rules) directly in the Config UI.

---

5. Architecture Overview

- **Frontend:** Next.js 16 (App Router) + React 19 + TypeScript + Tailwind + shadcn/ui.
- **Backend:** Next.js API routes with Supabase as primary data layer.
- **AI runtime:** Vercel AI SDK 5 foundation with MCP support and tool execution.
- **Database:** Supabase PostgreSQL with pgvector HNSW indexes; 14 production tables (documents, document_chunks, chats, folders, subscriptions, etc.) plus storage buckets.
- **Infrastructure:** Designed for Vercel/Next hosting; Supabase Cloud recommended but self-hosting supported if pgvector with HNSW is available.
- **Deployment footprint:** Works locally, self-hosted, or on managed platforms (Vercel, Render, Fly). WhatsApp adapters deploy on Koyeb or Fly.io.

---

6. Getting Started

6.1 Prerequisites

- Node.js 18+ and npm.
- Git for repo access.
- Supabase project (free tier works).
- OpenAI API key (mandatory for embeddings).
- At least one chat model provider key (OpenRouter recommended).
- LlamaCloud API key for document processing (required for document ingestion).
- Optional: Stripe, Polar, ElevenLabs, Fal.ai, Replicate credentials depending on features.

6.2 Quick Start Workflow

```
git clone <YOUR_PRIVATE_REPO_URL>
cd chatrag
npm install
npm run dev      # http://localhost:3000 (creates .env.local
        baseline on first run)
npm run config   # http://localhost:3333 (visual configuration UI)
```

1. Keep both servers running in separate terminals.
2. Open the Config UI, fill in Supabase URL/keys, OpenAI, OpenRouter, LlamaCloud, and optional providers.
3. Save settings, then restart the dev server (`Ctrl+C`, `npm run dev`) to load new environment values.

6.3 Database Setup (Supabase)

1. Create a Supabase project and wait for provisioning.
2. Open the SQL Editor and paste the entire contents of `supabase/complete_setup.sql`.
3. Execute once to enable pgvector, create 14 tables with RLS, storage buckets, functions, and HNSW indexes.
4. From Project Settings → API, copy:
   - `NEXT_PUBLIC_SUPABASE_URL`
   - `NEXT_PUBLIC_SUPABASE_ANON_KEY`
   - `SUPABASE_SERVICE_ROLE_KEY` (keep secret; required for admin operations)
5. Verify connection by restarting the app; check browser console for errors.

6.4 Document Processing

- Upload via main app document dashboard or Config UI (admin workflows).
- LlamaCloud parses and chunks documents; OpenAI embeddings generated automatically.
- Chunks stored in `document_chunks` with HNSW index; retrieval feed powers contextual responses.
- For multi-tenant deployments, Supabase Row Level Security ensures users access only their data.
- Recommended flags:
  - `NEXT_PUBLIC_HIDE_DOCUMENT_DASHBOARD=false` to expose dashboard.
  - `NEXT_PUBLIC_READ_ONLY_DOCUMENTS_ENABLED=false` to allow user uploads (set true for admin-only workflows).

6.5 RAG System Prompt Requirements

- Prompts must include a literal `{{context}}` placeholder.
- Recommended template:

```
You are an AI assistant with access to documents.

Context:
{{context}}
```

```
Answer based on the context above. If the context is empty,
explain what info is needed.
```

- Configure via Config UI → System Prompt.
- Restart dev server after saving to ensure `.env.local` updates load.
- Use provided templates (sales, customer support, research, code assistant, WhatsApp conversational, etc.) as starting points.

6.6 Verification Checklist

1. **Basic chat:** Open http://localhost:3000, send a test prompt, confirm streaming response.
2. **Document QA:** Upload a sample PDF, ask about unique content, verify grounded answer.
3. **Persistence:** Restart `npm run dev` and confirm configuration persists.
4. **Admin access:** Add admin emails via Config UI (must match existing Supabase users).
5. **WhatsApp:** Select provider, complete QR auth, send/receive test messages.

---

7. WhatsApp Integration

ChatRAG ships with a dual-provider WhatsApp adapter powered by Baileys. It supports QR or pairing-code login, live webhooks, media delivery, session persistence, and optional MCP/web search access for mobile users.

7.1 Deploy the Baileys bridge

1. **Clone the Baileys service** (included with ChatRAG assets) or keep it in a separate repository. It exposes REST endpoints such as `/api/sessions/create`, `/api/sessions/:id/qr`, `/api/messages/send`, and signs outgoing webhook payloads.
2. **Choose a host:**
   - **Koyeb:** Push the Baileys project to Git, connect it in Koyeb, and deploy using the included `Procfile` (`web: node lib/server.js`). Attach a persistent volume if you want login sessions to survive restarts.
   - **Fly.io:** Run `fly launch` inside the Baileys folder (a sample `fly.toml` is provided). Fly builds the Dockerfile, deploys a global instance, and keeps ports HTTPS-ready.
3. **Set environment variables on the Baileys service:**
   - `WEBHOOK_URL=https://<your-chatrag-domain>/api/whatsapp/webhook` (public endpoint ChatRAG exposes)
   - `WEBHOOK_SECRET=<random-hex>` — must match `WHATSAPP_WEBHOOK_SECRET` in ChatRAG
   - Optional: `USE_PAIRING_CODE=true`, `PHONE_NUMBER=...` if you prefer pairing codes instead of QR, plus any provider-specific auth (Fly API key, Koyeb API key).
4. **Expose port 8080 or 3000** (depending on deployment) and confirm `/health` responds. The service stores auth credentials in `baileys_auth_*` folders—keep them on persistent storage.

7.2 Wire ChatRAG to your provider

1. Open the Config UI → **WhatsApp** tab.
2. Toggle **Enable WhatsApp Integration** (`NEXT_PUBLIC_WHATSAPP_ENABLED=true`).
3. Pick a **provider** (`WHATSAPP_PROVIDER=koyeb` or `flyio`).
4. Enter the appropriate base URL (`KOYEB_BAILEYS_URL` or `FLYIO_BAILEYS_URL`). Add optional API keys if your host requires authenticated requests.
5. Paste the **Webhook URL** you configured on the Baileys side (usually the public `/api/whatsapp/webhook` route of your ChatRAG deployment). Generate and store a matching `WHATSAPP_WEBHOOK_SECRET`.
6. Set the default WhatsApp model (`WHATSAPP_DEFAULT_MODEL`, e.g., `gpt-4.1-mini`), token limit (`WHATSAPP_MAX_TOKENS`), session cap (`WHATSAPP_MAX_SESSIONS_PER_USER`), and queue size (`WHATSAPP_MESSAGE_QUEUE_SIZE`).
7. Save the configuration; the dashboard writes everything to `.env.local`.

7.3 Enable advanced options

- **MCP in WhatsApp:** Toggle **Enable MCP Tools** to set `WHATSAPP_ENABLE_MCP=true`, letting mobile users invoke Zapier or custom MCP servers.
- **Web search in WhatsApp:** Toggle **Enable Web Search** (`WHATSAPP_ENABLE_WEB_SEARCH=true`) to expose Exa-powered search results.
- **Prompt tuning:** Adopt the "WhatsApp Conversational" system prompt template for concise, mobile-friendly responses.
- **Security:** Use webhook secrets and provider API tokens to secure the bridge. Rotate them if a device is lost.

7.4 Connect & monitor

1. In the web app, start a WhatsApp session; ChatRAG calls `/api/sessions/create` and displays the QR code returned by Baileys.
2. Scan the QR with the WhatsApp mobile app (linked devices). Once connected, ChatRAG receives webhook events for message delivery and status updates.
3. Use scripts under `scripts/whatsapp/` (`clear-whatsapp-sessions.js`, `test-whatsapp-flow.js`, `test-numbers.js`) to debug sessions, reset state, or confirm formatting.
4. Watch Baileys logs for disconnects. The ChatRAG client automatically retries (`ping`, `reconnect`, `refresh`) but redeploy or rescan if WhatsApp invalidates tokens.

Use cases include support inboxes, concierge bots, lead qualification, and any channel where users prefer WhatsApp over web chat.

---

8. Monetization & Authentication Setup

1. **Stripe:** Enable in Config UI, supply secret/public keys, configure products for chatbot access.

2. **Polar:** Alternative for one-time payments or memberships.
3. **Supabase Auth:** Enable providers (email/password out of the box; configure GitHub/Google as needed).
4. **Access control:** Combine auth and subscription state to gate features or limit usage per tenant.
5. **Billing narratives for chatbots:** Emphasize clients can launch paid chat portals quickly, with usage tracked per Supabase tenant.

---

9. Media Generation & Advanced Features

- **Images:** `IMAGE_GENERATION_PROVIDER=openai|fal|replicate`. Models include `gpt-image-1`, Fal `flux` variants.
- **Video:** Fal video text-to-video models by default; set `USE_REPLICATE_PROVIDER=true` for Replicate fallback.
- **3D:** Fal Trellis for 3D assets; Replicate optional backup.
- **Speech:** OpenAI Whisper for STT, ElevenLabs for TTS/STT.
- **Localization:** UI localized in 14+ languages; prompts auto-translated with AI translation system.
- **Automation:** Zapier MCP integration ready; connect workflows or trigger external automations from conversations.

---

10. Deployment Guidance

- **Local development:** `npm run dev` + `npm run config`.
- **Production hosting:** Deploy Next.js app to Vercel or preferred host; ensure environmental variables mirror `.env.local`.
- **Database:** Use hosted Supabase (recommended) or self-hosted Postgres with pgvector + HNSW.
- **Edge considerations:** Keep environment secrets safe; use Vercel/hosted environment variables dashboards.
- **Scaling:** HNSW indexes handle large corpora (100k+ documents) with <200 ms retrieval. Utilize Supabase autoscaling and caching strategies for heavy workloads.

---

11. Competitive Comparison (ChatRAG vs Chatbase)

ChatRAG's landing page features a detailed side-by-side with Chatbase. Keep these highlights ready so assistants can answer "Why choose ChatRAG instead?" questions instantly.

Capability	ChatRAG	Chatbase
Licensing	✓ Pay once, own forever	✗ $40–500/mo subscription
Unlimited chats	✓ Included	✓ Included

Capability	ChatRAG	Chatbase
Unlimited messages	✓ No caps	✗ Tiered limits (100–40K/mo)
Ownership & hosting	✓ Self-hosted; you own stack	✗ SaaS-only; vendor lock-in
Lose access if vendor shuts down	✗ You keep code & data	✓ Access ends with vendor
Interface	✓ Full ChatGPT/ Claude-style web app + widget	✗ Widget only
WhatsApp (no Business account)	✓ Built-in dual providers	✗ Not offered
Monetization	✓ Stripe & Polar ready	✗ Not included
Media generation	✓ Images, video, 3D (Fal.ai, Replicate)	✗ Not available
MCP & automation	✓ Zapier, n8n, custom MCP tools	✗ No MCP support
Web search + RAG	✓ Combined retrieval	✗ RAG only
Model flexibility	✓ 200+ models via OpenRouter API	✗ Limited provider list
Claude Artifacts	✓ Generate shareable artifacts	✗ Not supported
Shareable chat links	✓ Yes	✗ No
Export chats as PDF	✓ Yes	✗ No
Manual memory updates	✓ Add AI replies to KB	✗ Not offered
Speech features	✓ STT & TTS built-in	✗ Not supported
Incognito chats	✓ Private sessions	✗ Not supported
Data control	✓ You own everything end-to-end	✗ Hosted entirely by Chatbase

Key takeaway: ChatRAG eliminates recurring license costs, grants full data ownership, and includes advanced production features (WhatsApp, monetization, media generation, MCP, artifacts) that Chatbase users cannot access without custom builds.

---

12. Troubleshooting Quick Reference

- **RAG not grounding answers:** Confirm `{{context}}` in prompt; ensure embeddings generated; verify `document_chunks` table populated.
- **Slow search:** Check HNSW index exists (`m=64`, `ef=200`); confirm using `text-embedding-3-small`.
- **Configuration not saving:** Ensure `SUPABASE_SERVICE_ROLE_KEY` set; `admin_settings` table exists; restart dev server after saving.
- **API key errors:** Remove trailing spaces; confirm scopes; re-save in Config UI.
- **WhatsApp disconnects:** Regenerate QR, verify server reachable, check provider logs.

- **Performance profiling:** Temporary flags (`NEXT_PUBLIC_DISABLE_DEBUG_LOGS`, `NEXT_PUBLIC_REDUCED_MOTION`) reduce UI noise during debugging.

---

13. Frequently Asked Questions (Landing Page)

1. **What is RAG (Retrieval Augmented Generation)?**
   RAG combines language models with a knowledge base so responses stay contextual and accurate. It is effectively giving your AI assistant long-term memory—you upload docs, ChatRAG retrieves the right passages, and the model explains them naturally.

2. **How many chatbots can I build with ChatRAG?**
   Unlimited. Build for one client or one thousand—there are no limits on the number of spaces, datasets, or deployments you can ship.

3. **Do I need technical experience?**
   Only basic skills (copy/paste API keys) to launch the default chatbot. Cursor "vibe coding" or light TypeScript knowledge lets you customize deeply, but the visual Config UI covers everyday tasks.

4. **What's included in the boilerplate?**
   A complete Next.js app with AI integrations, Supabase database schema, authentication, Stripe & Polar payments, document dashboard, WhatsApp adapters, and a polished chat UI ready for production.

5. **Can I customize the chatbot's appearance?**
   Yes. Upload logos, tweak colors, alter prompts, and restyle widgets from the dashboard. Developers can go further with Cursor/TypeScript to match any brand.

6. **Are there subscriptions? Do I really own the chatbots I create?**
   Yes—you fully own them. ChatRAG is a one-time purchase, and you decide how to monetize via Stripe or Polar. Your code, your infrastructure.

7. **How often is ChatRAG updated?**
   Continuously. Carlos ships the same build he uses for client projects, so new AI features are incorporated as soon as they're proven in production.

8. **What's on the roadmap?**
   Upcoming drops include realtime voice conversations, persistent memory layered on top of RAG, and browser automation so bots can take action on the web. ChatRAG Complete buyers get lifetime access to these upgrades.

9. **What's the best reason to buy ChatRAG?**
   It was built for real client work. You focus on winning and serving customers; ChatRAG handles the hard engineering—RAG optimization, model updates, infrastructure—so you stay ahead without building everything from scratch.

---

14. Support & Resources

- **Documentation:** `AGENTS.md`, `CLAUDE.md`, `WARP.md`, and the repository README for in-depth developer rules.
- **Community:** Discord server — https://discord.gg/QYESpZvU
- **Email:** hello@chatrag.ai
- **Videos:** Watch walkthroughs stored in `ChatRAG Videos` (local) and embedded demo at https://www.youtube.com/embed/CRUlv97HDPI.
- **Landing page:** https://chatrag.ai for live marketing assets.

---

15. About the Creator

- **Founder:** Carlos Marcial
- **Role:** Independent engineer & designer behind ChatRAG.
- **Primary handle:** x.com/carlosmarcialt
- **Mission:** Help developers, agencies, and founders launch production-grade AI copilots without monthly licensing lock-in.

Encourage visitors to reach out directly via X or email for enterprise inquiries, custom integrations, or partnership opportunities.

---

16. FAQ Prompts for Chatbots

1. **“What is ChatRAG?”** — Production-ready Next.js RAG platform with visual config, 200+ models, and pay-once licensing.
2. **“Who is ChatRAG for?”** — Developers, AI agencies, founders, enterprises needing custom knowledge assistants.
3. **“How much does it cost?”** — Starter $199, Complete $269, one-time payments with lifetime ownership.
4. **“How do I set it up?”** — Clone repo, install dependencies, run dev server + config UI, execute Supabase setup SQL, add API keys.
5. **“Does it support WhatsApp?”** — Yes, dual-provider architecture (Koyeb/Fly), no business account required.
6. **“Can I monetize my chatbot?”** — Yes, Stripe & Polar integrations for subscriptions or one-time access.
7. **“What models can I use?”** — 100+ via OpenRouter plus direct OpenAI, Anthropic, Google, DeepSeek.
8. **“Is the license unlimited?”** — Commercial license with unlimited deployments for you and your clients.
9. **“Where can I get help?”** — Discord community, hello@chatrag.ai, or contact Carlos on X.
10. **“Who built ChatRAG?”** — Carlos Marcial (x.com/carlosmarcialt).

---

17. Revision History

- **2025-10-31:** Documented Config UI coverage (multi-modal toggles, Exa web search, MCP/Zapier setup) and expanded WhatsApp deployment guidance.
- **2025-10-31:** Initial consolidated knowledge base prepared for ChatRAG landing page chatbots.