001 — ASSISTENTE CONVERSAZIONALE RAG

Chatbot RAG su misura per knowledge base editoriali.

Architettura riutilizzabile su qualsiasi dominio verticale.

STACK

BACKEND	Python 3.11 · FastAPI · Uvicorn · Pydantic v2 · httpx async
LLM	Google Gemini 2.5 Flash Lite — generazione streaming SSE
EMBEDDINGS	gemini-embedding-001 — 3072 dim, normalized
VECTOR DB	sqlite-vec — embedded, file singolo, zero infra
FRONTEND	JS vanilla · zero build · CSS scoped · fetch + ReadableStream
INFRA	VPS Linux · systemd · nginx + Let's Encrypt · SSE-friendly proxy

ARCHITETTURA

01

INGEST

Vault Obsidian come fonte autorevole. Frontmatter YAML per tema, tipo, priorita. Chunk per heading, 300–500 token target. Embedding batch via API Gemini.

02

RETRIEVAL

Query embedded con stesso modello. k-NN cosine su sqlite-vec. Top-k chunk + metadati iniettati nel prompt. System prompt definisce tono e fuori-scope.

03

DELIVERY

Endpoint POST /chat in SSE streaming. Widget JS consuma token-by-token. Markdown reso inline, sanitizzato lato client. Mobile fullscreen, safe-area insets.

STRUTTURA REPO

```
backend/app/api/          endpoint HTTP (chat SSE, health)
backend/app/rag/          chunker, embeddings, retriever, store, ingest CLI
backend/app/llm/          client Gemini con generazione streaming
backend/app/prompts/      system prompt caricato dalla KB
backend/data/             vectors.db (sqlite-vec, gitignored)
frontend/widget/          widget.js + widget.css
deploy/                   unit systemd + config nginx + runbook
```

RIUSABILITA — VARIABILI DI PROGETTO

KB	Obsidian, cartella markdown, export Notion, CMS headless
LLM	Gemini, OpenAI, Anthropic, modelli on-prem (Ollama, vLLM)
VECTOR	sqlite-vec (small/mid) · Qdrant / pgvector (scale)
PERSONA	Tono, lingua, regole di rifiuto via system prompt editoriale
WIDGET	Variabili CSS — colori, posizione, copy, modalità mobile
DEPLOY	VPS · container · serverless edge (Cloud Run, Fly.io)

→ ARCHITETTURA AGNOSTICA · COSTI MARGINALI · DATI SOTTO CONTROLLO · ADATTABILE A QUALSIASI DOMINIO VERTICALE