SaaS Project Blueprint: Multi-Tenant Knowledge Vault

---

1. Vision & Differentiation

The goal is to build a **multi-tenant SaaS knowledge assistant** that empowers organizations to securely store, search, and interact with their internal documents. Unlike generic RAG apps, this platform differentiates itself by:

- **Tenant-specific fine-tuning & embeddings** for customized accuracy.
- **Gap analytics**: dashboard insights into unanswered queries, frequent topics, and missing knowledge.
- **Proactive recommendations**: suggestions on which documents to add or update.
- **Transparent AI**: always citing document sources and maintaining audit trails.
- **Multi-modal ingestion**: handling PDFs, DOCX, images (OCR), and structured tables.
- **Integration-ready**: API and plug-ins for Slack, Teams, LMS, or customer portals.

---

2. Naming Candidates (To Be Finalized)

- **KnowVault** – Knowledge vault, clear and descriptive.
- **Vaultary** – Vault + Sanctuary, unique and brand-like.
- **Vaultoria** – Vault + Historia, story-driven.
- **CereVault** – Brain + Vault, intelligence angle.
- **Vaultium** – Vault + Element, modern techy feel.

(Next step: check domains, social handles, and trademarks.)

---

3. Core Features

User Experience

- Secure multi-tenant login and workspace isolation.
- Document upload (PDF, DOCX, TXT, CSV, images via OCR).
- Search & Chat assistant with contextual, cited responses.
- Dashboard for query history, usage metrics, and recommendations.
- Exportable insights for admins.

AI Features

- **Embeddings**: Document chunks embedded using open-source embedding models (e.g., Instructor, MiniLM, OpenAI embeddings if budget allows).
- **RAG (Retrieval-Augmented Generation)**: Query routed through vector store → relevant chunks → LLM response.
- **Fine-tuning**: Per-tenant adapter models (LoRA/PEFT) for specialized vocabulary and domain use.

- **Gap Analysis**: Identify unanswerable queries, cluster recurring questions, suggest new document uploads.

Analytics & Insights

- Top queries per week.
- Confidence scoring (low vs. high certainty answers).
- Knowledge gaps: documents or sections missing.
- Usage by user role (employee, customer, admin).

Integrations

- Slack, Teams, or LMS chatbot connectors.
- REST API for embedding/search as a service.
- Export reports (CSV, PDF).

---

4. System Architecture (Conceptual)

```
User / Tenant → Auth & Multi-Tenancy Layer
       │
       ▼
File Upload Module (PDF/DOCX/TXT/CSV/Image)
       │
       ▼
Document Processing:
   - Text extraction & OCR
   - Chunking & metadata tagging
       │
       ▼
Embeddings Module:
   - Embed chunks → Vector Store (pgvector, Qdrant, or Neon)
       │
       ▼
RAG Query Pipeline:
   - Query → Retrieve top-K docs → LLM inference
   - Response with citations
       │
       ▼
Chat UI / API Output

Parallel Paths:
- **Analytics Engine** → Dashboards & Insights
- **Fine-Tuning Module** → Per-tenant model adapters
- **Security & Compliance** → Isolation, encryption, audit logs
```

---

5. Security & Compliance

- Row-level tenant isolation in database.
- Encrypted data storage (AES-256 at rest, TLS in transit).
- Role-based access control.
- Audit logs for queries, responses, and data usage.
- GDPR-aligned user data handling.

---

6. Tech Stack

Frontend

- Next.js + Tailwind (clean, modern dashboard).
- tRPC for type-safe API calls.

Backend

- Node.js/TypeScript (via T3 stack).
- Prisma ORM with PostgreSQL/Neon.
- Vector DB: pgvector, Qdrant, or Weaviate.

AI/ML

- Embeddings: Instructor-XL / MiniLM / OpenAI embeddings (configurable).
- LLM Inference: Open-source (LLaMA-2/3, Mistral) + hosted inference APIs (Groq, OpenRouter) for speed.
- Fine-tuning: LoRA adapters stored per tenant.

Deployment

- Vercel (frontend).
- Supabase/Neon (database + storage).
- Render/Fly.io for backend.
- Open-source vector DB self-hosted if scaling.

Monitoring

- Basic metrics (latency, accuracy, error rates).
- Logging queries + feedback for improvements.

---

7. Free-Tier Development Path

- **Hosting**: Vercel (hobby), Supabase (free tier), Neon (free tier).
- **Vector Store**: pgvector on free PostgreSQL instance.
- **LLM**: OpenRouter (free credits) / HuggingFace Inference API.
- **OCR**: Tesseract (open-source).
- **Embeddings**: Instructor/MiniLM (free open-source) or OpenAI trial credits.

---

8. Roadmap

Phase 1 – MVP (4–6 Weeks)

- Multi-tenant login + workspaces.
- Document upload + processing.
- Embedding + retrieval with vector store.
- Basic chat assistant with citations.
- Minimal dashboard (query history).

Phase 2 – Analytics (4 Weeks)

- Usage statistics & top queries.
- Low-confidence query identification.
- Gap analysis & recommendations.
- Exportable CSV/PDF reports.

Phase 3 – Personalization (6 Weeks)

- Per-tenant LoRA fine-tuning.
- Support for multiple models per tenant.
- Switch between default & tenant-trained model.

Phase 4 – Integrations (6 Weeks)

- Slack/Teams/LMS chatbot connectors.
- REST API for external use.
- Webhooks for notifications.

Phase 5 – Scaling & Security (Ongoing)

- Audit logs & compliance features.
- Multi-region deployment.
- Custom billing & subscription plans.

---

9. Differentiators for Clients

- **Tailored AI**: Each client gets their own fine-tuned assistant.
- **Actionable Analytics**: Clear insights into knowledge usage & gaps.
- **Transparency**: All responses cite sources and maintain audit logs.
- **Cost-Efficiency**: Open-source first, scaling only when usage grows.
- **Integration-Friendly**: APIs and connectors for workplace adoption.

---

10. Next Steps

1. Finalize unique name + domain.
2. Create brand identity (logo, tagline, theme).
3. Set up repo + skeleton (create-t3-app).
4. Build MVP (Phase 1).

5. Start onboarding pilot users.