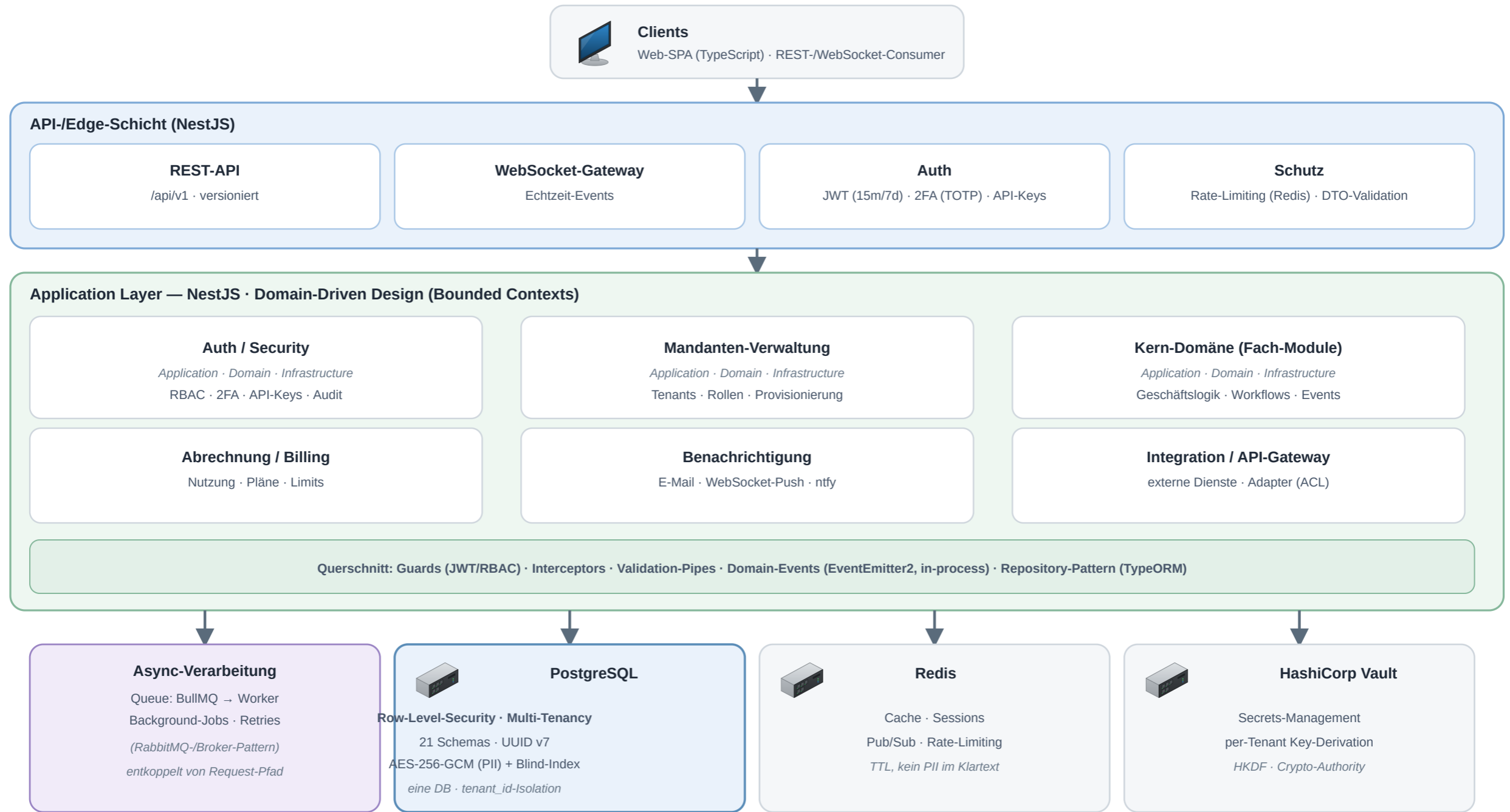


Multi-Tenant-SaaS · Software-/Backend-Architektur

NestJS (Domain-Driven Design) · PostgreSQL Row-Level-Security · REST + WebSocket · BullMQ-Worker · Redis · JWT/2FA · anonymisiert



★ Mandanten-Isolation (der Kern der Multi-Tenancy)

Jeder Request trägt eine tenant_id → PostgreSQL Row-Level-Security (FORCE) erzwingt Zeilen-Isolation pro Mandant — eine DB, harte Trennung.


PII pro Tenant AES-256-GCM verschlüsselt, Schlüssel via Vault (HKDF) abgeleitet. Domain-Schicht bleibt framework-frei (DDD), externe SDKs nur über Adapter (ACL).

 Datenspeicher (PostgreSQL · Redis · Vault)

 Client (Web-SPA / API-Consumer)

 Edge/API + Datenbank

 Application Layer (DDD)

 Async (Queue/Worker)

anonymisiertes Architektur-Muster der eigenen Multi-Tenant-SaaS-Plattform — generische Bounded Contexts, keine Produkt-Internas. Stack 1:1 wie betrieben.

Stack: NestJS · PostgreSQL/RLS · Redis · BullMQ · WebSocket · JWT/2FA · TypeORM · DDD · anonymisiert

