**Appalti AI Platform**

Technisch overzicht en werking

2025-08-27

# **Inhoudsopgave**

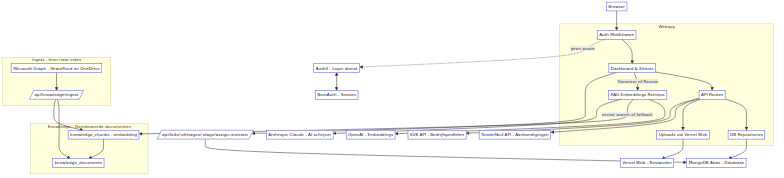
# **1. Overzicht**

Het Appalti AI Platform is een multi-tenant SaaS-applicatie (Next.js) die consultants en klanten helpt bij het vinden, schrijven en beoordelen van aanbestedingen.

Belangrijkste componenten: Webapp (Next.js), Auth0/NextAuth (login & sessies), MongoDB Atlas (database + vector index), Vercel Blob (bestanden), Upstash Redis (rate limiting), Sentry (monitoring), externe APIs (TenderNed, KVK), en AI (Anthropic voor genereren/review, OpenAI voor embeddings).

# **2. Architectuur in één oogopslag**

Onderstaande afbeelding toont de globale componenten en datastromen. RAG gebruikt geïndexeerde documenten uit MongoDB (knowledge\_\*), niet direct uit SharePoint/OneDrive; deze documenten komen via de ingestie-pipeline binnen.



# **3. Authenticatie en sessies**

Login verloopt via Auth0 (Universal Login). NextAuth verwerkt de callback en beheert sessies. In callbacks.signIn synchroniseren we de gebruiker en memberships naar MongoDB. De middleware beschermt routes en stuurt niet-ingelogde gebruikers naar /auth/signin.

# **4. Tenant context en rollen**

De actieve tenant/company wordt afgeleid uit memberships en cookies (activeCompanyId/activeTenantId). Platformrollen (Appalti) en company-rollen (client-tenant) bepalen wat een gebruiker mag doen.

# **5. Client Companies en team**

Client companies leven in de eigen tenant. Voor Enterprise-scenario’s worden teamleden beheerd en kunnen invites worden verstuurd en geaccepteerd. Eén client kan eigen tenders/bids beheren.

# **6. TenderNed integratie en koppelen**

De lijst met aanbestedingen komt via de TenderNed API. Na selectie wordt een Tender-document (source=tenderned, externalId) aangemaakt en – zo nodig – een Bid-proces gecreëerd. Hierdoor ontstaat een door Appalti te bewerken proces per tender.

# **7. Editor en AI (RAG)**

Bij “Genereer met AI (RAG)” maakt de app een embedding van de vraag, zoekt relevante passages in knowledge\_chunks (Atlas Vector Search of fallback cosine-ranking) en construeert een prompt voor Anthropic (Claude). Bij “Review per alinea” geeft de AI per alinea diagnose + verbeterde versie.

# **8. Ingestie van documenten (SharePoint/OneDrive)**

Via Microsoft Graph worden tekstachtige documenten opgehaald (vertical: SharePoint, horizontaal: OneDrive). De ingestie-route splitst tekst in chunks, maakt embeddings (OpenAI) en slaat alles op in knowledge\_documents en knowledge\_chunks. RAG zoekt vervolgens uitsluitend in deze geïndexeerde kopieën.

# **9. Uploads en bijlagen**

Bestanden die in de editor worden geüpload gaan via @vercel/blob. De URL en metadata worden opgeslagen in Bid.stages.$.attachments zodat ze per fase terug te vinden zijn.

# **10. Datamodel (kern)**

Kerncollecties: companies, users, memberships, clientCompanies, tenders, bids (stages). Voor RAG: knowledge\_documents en knowledge\_chunks (embedding).

# **11. Externe APIs**

• TenderNed – aanbestedingsaankondigingen.
• KVK – bedrijfsprofielen.
• Auth0 – authenticatie.
• Microsoft Graph – documentenbronnen.
• Anthropic – genereren/review.
• OpenAI – embeddings.

# **12. Observability, veiligheid en limiting**

Sentry logt errors. Upstash Redis beperkt verzoeken waar nodig (b.v. KVK, invites). Geheimen staan in Vercel Project Settings. Multi-tenant isolatie wordt afgedwongen in repositories en helpers.

# **13. Troubleshooting (bekende punten)**

Als Atlas Vector Search ontbreekt, zie je “Vector search unavailable, falling back to in-memory similarity”. De app werkt dan gewoon, maar trager en minder precies. Oplossing: maak index "vector\_index" op knowledge\_chunks.embedding (dimensie 1536).

# **14. Bijlagen**

• A4 architectuurdiagram: public/architecture-a4.png