**01 - Gedetailleerd Specificatiedocument voor NEXA**

**Status:** IN PROGRESS - (Database Schema & RLS Basis Geïmplementeerd)

**Inhoudsopgave:**

1. Use Cases per Doelgroep
2. Functionele Requirements (TODO)
3. Non-Functionele Requirements (TODO)
4. Acceptatiecriteria (TODO)

**1. Use Cases per Doelgroep**

Dit beschrijft de primaire scenario's waarin de verschillende gebruikersrollen NEXA zullen gebruiken.

**1.1 Salesforce Developers**

* **Als Salesforce Developer wil ik** *alle afhankelijkheden* van een specifiek metadata component (veld, Apex class, Flow, LWC) visualiseren, **zodat ik** de impact van een wijziging of verwijdering kan inschatten voordat ik deze doorvoer.
  + *Scenario:* Voor het aanpassen van een Apex Trigger, wil de developer zien welke objecten, velden of andere classes deze trigger aanroepen of door deze trigger worden beïnvloed.
* **Als Salesforce Developer wil ik** een *AI-gegenereerde uitleg* krijgen *waarom* een voorgestelde wijziging (bijv. verwijderen custom field 'Legacy\_Status\_\_c') impact heeft op andere componenten (bijv. 3 Flows, 2 Reports, 1 Apex Class), **zodat ik** snel begrijp wat het risico is en waar ik moet kijken.
* **Als Salesforce Developer wil ik** *specifieke Flow errors* (veroorzaakt door metadata inconsistenties, zoals een verwijzing naar een niet-bestaand veld) snel kunnen identificeren en de *root cause* kunnen zien in de context van de metadata graph, **zodat ik** troubleshooting tijd kan verkorten.
* **Als Salesforce Developer wil ik** *voorafgaand aan een deployment* via CI/CD (bijv. Copado, Gearset) een automatische NEXA impact analyse kunnen triggeren, **zodat ik** een extra veiligheidscheck heb op onverwachte neveneffecten. (V1/V2 Feature)
* **Als Salesforce Developer wil ik** kunnen zien welke metadata componenten *niet (meer) gebruikt* lijken te worden (geen inkomende relaties in de graph), **zodat ik** kan helpen met het opschonen van de org (technical debt reductie).

**1.2 Salesforce Admins & Consultants**

* **Als Salesforce Admin wil ik** de impact simuleren van het *wijzigen van een veldtype* (bijv. van Text naar Picklist) of het *verwijderen van een veld*, **zodat ik** proactief kan zien welke page layouts, flows, validation rules, reports, etc. aangepast moeten worden.
* **Als Salesforce Consultant wil ik** snel een *overzicht krijgen van de complexiteit* en belangrijkste afhankelijkheden in een (voor mij nieuwe) Salesforce org, **zodat ik** sneller inzicht krijg in de bestaande configuratie en risicogebieden.
* **Als Salesforce Admin wil ik** een *AI-suggestie krijgen voor remediatie* wanneer een hoog-risico wijziging wordt gedetecteerd (bijv. "Update deze 3 Flows en verwijder het veld uit Rapport X voordat je het veld verwijdert"), **zodat ik** begeleiding krijg bij het veilig doorvoeren van de wijziging. (V1 Feature)
* **Als Salesforce Admin wil ik** de impact zien van het *deactiveren van een Flow* of *aanpassen van een Permission Set*, **zodat ik** begrijp welke processen of gebruikerstoegang hierdoor beïnvloed worden.
* **Als Salesforce Consultant wil ik** *metadata snapshots* kunnen vergelijken (bijv. voor en na een release), **zodat ik** kan valideren welke wijzigingen zijn doorgevoerd en of er onbedoelde aanpassingen zijn gemaakt. (V2 Feature)

**1.3 Change Managers**

* **Als Change Manager wil ik** een *geconsolideerd impact rapport* en *risicoscore* ontvangen voor een voorgestelde set van wijzigingen (change set), **zodat ik** een onderbouwde beslissing kan nemen over de goedkeuring ervan.
* **Als Change Manager wil ik** de *AI-gegenereerde uitleg* van de impact kunnen gebruiken als onderdeel van de documentatie in het change request, **zodat** de risico's duidelijk zijn voor alle stakeholders.
* **Als Change Manager wil ik** een *historisch overzicht* kunnen zien van eerdere analyses en gedetecteerde risico's, **zodat ik** trends kan identificeren en het change proces kan verbeteren. (Logging in Supabase)
* **Als Change Manager wil ik** notificaties ontvangen wanneer een geplande wijziging (via CI/CD hook) een *hoog risico* heeft volgens NEXA, **zodat ik** proactief kan ingrijpen. (V2 Feature)

**1.4 Enterprise Architecten**

* **Als Enterprise Architect wil ik** een *holistisch beeld* krijgen van de metadata structuur en de belangrijkste afhankelijkheden binnen de Salesforce org (en potentieel tussen orgs/clouds), **zodat ik** de algehele architectuur kan beoordelen en documenteren.
* **Als Enterprise Architect wil ik** *gebieden met hoge complexiteit* of *verouderde patronen* identificeren via de graph analyse, **zodat ik** input heb voor refactoring of moderniseringsinitiatieven.
* **Als Enterprise Architect wil ik** NEXA gebruiken om de *impact van het introduceren of verwijderen van Managed Packages* te analyseren, **zodat ik** de afhankelijkheden met custom metadata beter begrijp.
* **Als Enterprise Architect wil ik** de *evolutie van de metadata graph* over tijd kunnen volgen, **zodat ik** architecturale drift kan monitoren en de technische schuld kan beheersen. (Via snapshots)

**2. Functionele Requirements**

Deze sectie beschrijft de specifieke functionaliteiten die NEXA moet bieden.

2.1 Authenticatie en Autorisatie

* FR-AUTH-01: Gebruikers moeten zich kunnen registreren voor een NEXA account (bijv. met e-mail/wachtwoord of via OAuth provider zoals Google/GitHub).
* FR-AUTH-02: Geregistreerde gebruikers moeten kunnen inloggen op het NEXA platform.
* FR-AUTH-03: Gebruikers moeten hun NEXA account kunnen koppelen aan één of meerdere Salesforce orgs via een veilige OAuth 2.0 PKCE flow.
* FR-AUTH-04: Toegang tot metadata en analyses moet beperkt zijn tot de gebruiker(s) die toegang hebben tot de gekoppelde Salesforce org binnen NEXA (basis Role-Based Access Control - RBAC).
* FR-AUTH-05: Salesforce credentials (tokens) moeten veilig (encrypted) worden opgeslagen en beheerd.

2.2 Metadata Pipeline (Fetch, Parse, Store)

* FR-PIPE-01: Het systeem moet, op verzoek van de gebruiker, verbinding maken met een gekoppelde Salesforce org en metadata ophalen via de Salesforce Metadata API en/of Tooling API.
* FR-PIPE-02: Het systeem moet de volgende Salesforce metadata types kunnen parsen (MVP focus):
  + Custom Objects & Fields (inclusief relaties, formules)
  + Flows (Screen Flows, Autolaunched Flows, Record-Triggered Flows - inclusief elementen zoals Decisions, Assignments, Get/Update/Create/Delete Records)
  + Apex Triggers (identificatie van object en basis events - DML operations)
  + Reports & Report Types (gebruikte objecten en velden)
  + (V1+) Apex Classes (basis dependencies)
  + (V1+) Permission Sets / Profiles (veld- en objectpermissies)
  + (V1+) Page Layouts
  + (V1+) Validation Rules
* FR-PIPE-03: Het systeem moet de geparste metadata omzetten naar een vooraf gedefinieerde, consistente structuur per component type (bijv. vaste JSON-schema's of tabeldefinities), waarbij informatie zoals attributen en relaties uniform worden vastgelegd, geschikt voor opslag in Supabase en latere verwerking.
* FR-PIPE-04: Het systeem moet de gestructureerde, geparste metadata opslaan in de Supabase database, gekoppeld aan de specifieke Salesforce org en een uniek "snapshot" ID.
* FR-PIPE-05: Het systeem moet de gebruiker feedback geven over de voortgang en status (Gestart, Bezig met ophalen, Bezig met parsen, Opslaan voltooid, Mislukt) van de metadata pipeline run.
* FR-PIPE-06: Het systeem moet om kunnen gaan met time-outs of errors tijdens het ophalen/parsen en deze loggen.
* FR-PIPE-07: Gebruikers moeten een lijst kunnen zien van recent aangemaakte metadata snapshots voor hun gekoppelde orgs.
* FR-PIPE-08: Elk metadata snapshot vertegenwoordigt de staat van de verbonden Salesforce org op het moment van de 'Fetch' actie. Latere wijzigingen in Salesforce (inclusief verwijderingen) worden pas zichtbaar in NEXA nadat een nieuwe snapshot is gemaakt.
* FR-RSLT-01: De resultaten van alle uitgevoerde analyses (inclusief gedetecteerde inconsistenties, risicoscores, etc.) moeten worden opgeslagen in Supabase, gekoppeld aan de analyse run en de betreffende metadata snapshot.

2.3 Knowledge Graph (Population & Querying)

* FR-GRAPH-01: Het systeem moet, op verzoek van de gebruiker voor een specifieke snapshot, de opgeslagen gestructureerde metadata uit Supabase lezen.
* FR-GRAPH-02: Het systeem moet de gelezen metadata omzetten naar een graph-structuur bestaande uit nodes (componenten zoals Object, Field, Flow, Report) en relaties (zoals USED\_IN, REFERENCES, BELONGS\_TO, TRIGGERS).
* FR-GRAPH-03: Het systeem moet deze graph-structuur populeren (aanmaken/updaten) in de Neo4j database, gekoppeld aan het snapshot ID.
* FR-GRAPH-04: Het systeem moet Neo4j queries kunnen uitvoeren om afhankelijkheden en impactpaden te traceren (bijv. "vind alle componenten die Field X gebruiken", "vind alle componenten die beïnvloed worden door Flow Y").
* FR-GRAPH-05: Het systeem moet basisstatistieken over de graph kunnen leveren (aantal nodes/relaties per type) voor een snapshot.
* FR-GRAPH-06: Het systeem moet specifieke analyse-queries op de Neo4j graph kunnen uitvoeren om potentiële problemen te detecteren, zoals metadata inconsistenties in Flows (bijv. verwijzingen naar niet-bestaande velden, gebruik van inactieve componenten) of andere gedefinieerde risicopatronen.

2.4 Frontend & Visualisatie

* FR-UI-01: De gebruiker moet een dashboard zien met een overzicht van gekoppelde orgs en recente snapshots/analyses.
* FR-UI-02: De gebruiker moet de metadata pipeline (Fetch/Store, Build Graph, Analyze) kunnen starten vanuit de UI.
* FR-UI-03: Het systeem moet een interactieve visualisatie van de Neo4j knowledge graph tonen voor een geselecteerde snapshot.
* FR-UI-04: De graph visualisatie moet basisinteracties ondersteunen: zoomen, pannen, selecteren van nodes/relaties.
* FR-UI-05: Bij selectie van een node in de graph moet de gebruiker basisinformatie over dat metadata component kunnen zien (naam, type, etc.) en de directe relaties ervan.
* FR-UI-06: (V1+) De gebruiker moet kunnen filteren in de graph visualisatie (bijv. op component type, relatie type).
* FR-UI-07: (V1+) De gebruiker moet aantekeningen kunnen maken bij specifieke nodes of relaties.
* FR-UI-08: De gebruiker moet de resultaten van uitgevoerde analyses (zoals gedetecteerde Flow inconsistenties of andere geïdentificeerde risico's) op een duidelijke manier kunnen inzien in de UI, met details over het betreffende component en de aard van het probleem/risico.

2.6 AI Reasoning & Explainability

* FR-AI-01: De gebruiker moet via een "AI Assistant" interface vragen kunnen stellen over metadata relaties of de impact van (gesimuleerde) wijzigingen (bijv. "Wat gebeurt er als ik veld X verwijder?", "Welke flows gebruiken object Y?").
* FR-AI-02: Het systeem moet de gebruikersvraag interpreteren en vertalen naar relevante Neo4j queries om het impactpad of de relaties te vinden.
* FR-AI-03: Het systeem moet de resultaten van de Neo4j query (het impactpad/relevante subgraph) samen met context naar een LLM (GPT/Claude) sturen via een gestructureerde prompt.
* FR-AI-04: Het systeem moet het antwoord van de LLM (de uitleg van de impact/relaties) verwerken en presenteren aan de gebruiker in de AI Assistant interface.
* FR-AI-05: (V1) Het systeem moet een basis risicoscore kunnen berekenen voor een component of een gesimuleerde wijziging, gebaseerd op het aantal en type afhankelijkheden. Deze score moet zichtbaar zijn in de UI.
* FR-AI-06: (V1) Gebruikers moeten feedback kunnen geven (bijv. 👍/👎, commentaar) op de kwaliteit en correctheid van de AI-gegenereerde uitleg.
* FR-AI-07: (V1) De gebruikersfeedback moet worden opgeslagen in Supabase, gekoppeld aan de interactie, voor latere analyse en verbetering van de AI-prompts/modellen.
* FR-AI-08: (V1+) Het systeem moet op basis van gedetecteerde risico's of errors AI-gegenereerde suggesties voor remediatie kunnen geven.

2.7 Integraties & API

* FR-API-01: Er moet een interne API (gehost op Supabase Functions) beschikbaar zijn die de frontend gebruikt om te communiceren met de backend (Supabase DB, Neo4j, LLM).
* FR-API-02: (V2+) Er moeten externe API endpoints (REST/GraphQL) beschikbaar komen voor integratie met externe tools (bijv. CI/CD pipelines).
* FR-API-03: (V2+) Het systeem moet via webhooks kunnen reageren op events uit CI/CD tools (bijv. start impact analyse bij nieuwe Pull Request).

2.8 Analyse Resultaten

* FR-RSLT-01: De resultaten van alle uitgevoerde analyses (inclusief gedetecteerde inconsistenties, risicoscores, etc.) moeten worden opgeslagen in Supabase, gekoppeld aan de analyse run en de betreffende metadata snapshot.
* FR-RSLT-02: Alle analyses (graph queries, AI reasoning, inconsistentie detectie) worden uitgevoerd binnen de context van één specifieke, door de gebruiker geselecteerde, metadata snapshot. De resultaten reflecteren de metadata zoals vastgelegd in die snapshot.

2.9 Snapshot beheer en vergelijken

* FR-COMP-01 (V2 Feature): Het systeem moet gebruikers in staat stellen om twee metadata snapshots met elkaar te vergelijken om verschillen te identificeren, inclusief toegevoegde, gewijzigde en verwijderde componenten.

**3. Non-Functionele Requirements**

Deze sectie beschrijft de kwaliteitseisen waaraan het NEXA-platform moet voldoen.

**3.1 Performance & Schaalbaarheid**

* **NFR-PERF-01 (Metadata Fetch):** Het ophalen van metadata van een gemiddelde Salesforce org (bijv. <500 custom objects, <1000 flows/processes) moet binnen een acceptabele tijd voltooid zijn (streefwaarde: < 15 minuten). Gebruiker moet feedback krijgen tijdens lange runs.
* **NFR-PERF-02 (Parsing & Storage):** Het parsen van de opgehaalde metadata en opslaan in Supabase moet efficiënt gebeuren (streefwaarde: < 10 minuten na fetch voltooid).
* **NFR-PERF-03 (Graph Population):** Het populeren van de Neo4j graph vanuit de Supabase data moet binnen een redelijke tijd plaatsvinden (streefwaarde: < 5 minuten voor een gemiddelde org snapshot).
* **NFR-PERF-04 (Graph Query):** Standaard impactanalyse queries op de Neo4j graph (bijv. directe afhankelijkheden opvragen) moeten doorgaans binnen enkele seconden (< 5s) resultaat geven in de UI. Complexe, diepe analyses kunnen langer duren.
* **NFR-PERF-05 (AI Response):** De reactietijd van de AI Assistant (van vraag tot antwoord, inclusief Neo4j query en LLM call) moet acceptabel zijn voor interactief gebruik (streefwaarde: < 10-15 seconden voor de meeste vragen).
* **NFR-SCALE-01 (Gebruikers):** Het platform moet ontworpen zijn om gelijktijdig gebruik door tientallen tot honderden gebruikers te ondersteunen (V1). Supabase en Neo4j AuraDB bieden hiervoor schaalbaarheidsopties.
* **NFR-SCALE-02 (Orgs):** Het systeem moet kunnen omgaan met data van honderden gekoppelde Salesforce orgs, met duidelijke scheiding van data per tenant (org/gebruiker).
* **NFR-SCALE-03 (Metadata Grootte):** Hoewel initiële focus op gemiddelde orgs ligt, moet de architectuur rekening houden met de mogelijkheid om in de toekomst zeer grote en complexe Salesforce orgs te ondersteunen (bijv. door optimalisaties in parsing, storage, graph model).

**3.2 Betrouwbaarheid & Beschikbaarheid**

* **NFR-REL-01:** De kernfunctionaliteiten (metadata pipeline, graph analyse, AI uitleg) moeten een hoge mate van betrouwbaarheid hebben. Fouten tijdens processen moeten correct worden afgehandeld en gelogd.
* **NFR-REL-02:** Het systeem moet robuust zijn tegen fouten in de Salesforce API (bijv. tijdelijke onbeschikbaarheid, API limieten) en hier correct op reageren (bijv. retry-mechanisme, duidelijke foutmelding).
* **NFR-AVAIL-01:** Het NEXA platform (frontend en API's gehost via Supabase) moet streven naar een hoge beschikbaarheid (bijv. 99.5% uptime), afhankelijk van de SLA's van de onderliggende cloud providers (Supabase, Neo4j AuraDB, LLM provider). Gepland onderhoud wordt gecommuniceerd.

**3.3 Beveiliging**

* **NFR-SEC-01 (Authenticatie):** Alle toegang tot het platform en API's moet beveiligd zijn via sterke authenticatiemechanismen (Supabase Auth).
* **NFR-SEC-02 (Autorisatie):** Toegang tot data moet gebaseerd zijn op rollen en eigendom (RBAC). Gebruikers mogen alleen data zien van de orgs waartoe zij geautoriseerd zijn (Supabase Row Level Security).
* **NFR-SEC-03 (Data Encryptie):** Gevoelige data, met name Salesforce access/refresh tokens, moeten zowel 'at rest' (in Supabase DB) als 'in transit' (API calls) versleuteld zijn.
* **NFR-SEC-04 (API Security):** API endpoints moeten beschermd zijn tegen veelvoorkomende aanvallen (OWASP Top 10), inclusief input validatie en rate limiting.
* **NFR-SEC-05 (Salesforce Connectie):** De OAuth 2.0 PKCE flow moet correct geïmplementeerd worden voor veilige Salesforce authenticatie. Tokens mogen nooit worden blootgesteld aan de frontend.
* **NFR-SEC-06 (Dependency Scanning):** Software dependencies (NPM packages, etc.) moeten regelmatig worden gescand op bekende kwetsbaarheden.
* **NFR-SEC-07 (Logging & Audit):** Belangrijke security events (logins, Salesforce connecties, toegang tot data) moeten worden gelogd voor audit doeleinden.

**3.4 Bruikbaarheid (Usability)**

* **NFR-USE-01:** De gebruikersinterface moet intuïtief en makkelijk te navigeren zijn voor de gedefinieerde doelgroepen.
* **NFR-USE-02:** Foutmeldingen en systeemberichten moeten duidelijk, begrijpelijk en (waar mogelijk) actiegericht zijn.
* **NFR-USE-03:** De graph visualisatie moet helder zijn en gebruikers helpen om complexe afhankelijkheden te begrijpen, niet te overweldigen.
* **NFR-USE-04:** De AI Assistant moet een natuurlijke interactie mogelijk maken en duidelijke, beknopte antwoorden geven.
* **NFR-USE-05:** Het moet voor de gebruiker duidelijk zijn welke data actueel is (status van snapshots en analyses).

**3.5 Onderhoudbaarheid & Uitbreidbaarheid**

* **NFR-MAINT-01:** De codebase moet modulair, goed gedocumenteerd en voorzien van tests (unit, integratie) zijn om toekomstig onderhoud en uitbreiding te vergemakkelijken.
* **NFR-MAINT-02:** Het moet relatief eenvoudig zijn om ondersteuning voor nieuwe Salesforce metadata types toe te voegen aan de parser en graph engine.
* **NFR-MAINT-03:** Het systeem moet configureerbaar zijn (bijv. API keys, LLM endpoint) via environment variabelen of een configuratieservice.
* **NFR-EXT-01:** De architectuur moet het mogelijk maken om in de toekomst nieuwe analysemodules of AI-capaciteiten toe te voegen zonder grote impact op bestaande functionaliteit.

**4. Acceptatiecriteria**

Deze sectie definieert de specifieke voorwaarden waaronder een Functionele Requirement als succesvol geïmplementeerd wordt beschouwd. Hieronder volgen voorbeelden voor enkele kern-FRs.

**FR-AUTH-03 (Salesforce Org Koppelen):**

* **AC-AUTH-03-01 (Start Koppeling):** Gegeven dat de gebruiker is ingelogd in NEXA, Wanneer de gebruiker klikt op "Koppel Salesforce Org", Dan wordt de gebruiker doorverwezen naar de Salesforce OAuth 2.0 login/autorisatie pagina.
* **AC-AUTH-03-02 (Succesvolle Autorisatie):** Gegeven dat de gebruiker succesvol autoriseert in Salesforce, Wanneer de gebruiker teruggeleid wordt naar NEXA, Dan wordt de connectie succesvol opgeslagen en ziet de gebruiker de gekoppelde org in zijn overzicht.
* **AC-AUTH-03-03 (Veilige Opslag):** Gegeven dat een Salesforce org succesvol is gekoppeld, Dan worden de verkregen access en refresh tokens veilig (versleuteld) opgeslagen in de Supabase database en zijn ze niet direct zichtbaar voor de gebruiker of in de frontend code.
* **AC-AUTH-03-04 (Foutafhandeling):** Gegeven dat de gebruiker de Salesforce autorisatie annuleert of dat er een fout optreedt, Wanneer de gebruiker teruggeleid wordt naar NEXA, Dan wordt er een duidelijke foutmelding getoond en wordt er geen koppeling opgeslagen.

**FR-PIPE-04 (Metadata Opslag in Supabase):**

* **AC-PIPE-04-01 (Data Aanwezig):** Gegeven dat de metadata fetch en parse stappen succesvol zijn voltooid voor snapshot XYZ, Wanneer een query wordt uitgevoerd op de Supabase parsed\_components tabel met snapshot ID XYZ, Dan worden de correcte geparste componenten (zoals Objecten, Velden, Flows) teruggevonden.
* **AC-PIPE-04-02 (Structuur Correct):** Gegeven dat een Custom Field component is opgeslagen voor snapshot XYZ, Dan bevat het opgeslagen record minimaal de verwachte attributen (bijv. api\_name, label, data\_type, object\_api\_name, snapshot\_id) in het correcte dataformaat.
* **AC-PIPE-04-03 (Relatie Consistentie):** Gegeven dat een Flow component verwijst naar een specifiek Custom Field, Dan is deze verwijzing correct vastgelegd in de opgeslagen data voor de Flow in Supabase (bijv. via een referenced\_field\_ids array of gerelateerde tabel).

**FR-GRAPH-03 (Graph Populatie in Neo4j):**

* **AC-GRAPH-03-01 (Nodes Aangemaakt):** Gegeven dat de gestructureerde metadata voor snapshot XYZ in Supabase staat en de graph populatie is gestart, Wanneer de populatie succesvol is voltooid, Dan bestaan er in Neo4j nodes met het label Field, Object, Flow etc., voorzien van een snapshotId property gelijk aan XYZ.
* **AC-GRAPH-03-02 (Relaties Aangemaakt):** Gegeven dat een Flow MyFlow het Veld MyField\_\_c op Object MyObject\_\_c gebruikt volgens de Supabase data voor snapshot XYZ, Wanneer de graph populatie succesvol is, Dan bestaat er in Neo4j een pad: (:Flow {name: 'MyFlow', snapshotId: 'XYZ'})-[:USES\_FIELD]->(:Field {name: 'MyField\_\_c', snapshotId: 'XYZ'})-[:BELONGS\_TO]->(:Object {name: 'MyObject\_\_c', snapshotId: 'XYZ'}). (Specifieke relatietypes en properties kunnen variëren).
* **AC-GRAPH-03-03 (Idempotentie):** Gegeven dat de graph populatie voor snapshot XYZ al eens succesvol is uitgevoerd, Wanneer de populatie opnieuw wordt gestart voor dezelfde snapshot XYZ, Dan leidt dit niet tot duplicate nodes of relaties voor die snapshot.

**FR-UI-03 & FR-UI-04 (Basis Graph Visualisatie):**

* **AC-UI-03-01 (Weergave na Build):** Gegeven dat de graph populatie voor snapshot XYZ succesvol is, Wanneer de gebruiker snapshot XYZ selecteert in het dashboard, Dan wordt de graph visualisatie component geladen en toont deze een set nodes en relaties uit de Neo4j database voor die snapshot.
* **AC-UI-04-01 (Zoom/Pan):** Gegeven dat de graph visualisatie is geladen, Wanneer de gebruiker scrollt of sleept binnen het visualisatiegebied, Dan zoomt de graph respectievelijk in/uit of verschuift de weergave (panning).
* **AC-UI-04-02 (Node Selectie):** Gegeven dat de graph visualisatie is geladen, Wanneer de gebruiker op een node klikt, Dan wordt deze node visueel gemarkeerd (bijv. andere kleur/grootte) en verschijnt er ergens basisinformatie over die node (bijv. in een zijpaneel).

**FR-AI-01 & FR-AI-04 (AI Vraag & Antwoord):**

* **AC-AI-01-01 (Vraag Stellen):** Gegeven dat de gebruiker in de AI Assistant interface is, Wanneer de gebruiker een vraag typt (bijv. "Welke flows gebruiken Account.Name?") en verstuurt, Dan wordt de vraag zichtbaar in de chatgeschiedenis en wordt er een backend call getriggerd.
* **AC-AI-04-01 (Antwoord Ontvangen):** Gegeven dat de gebruiker een vraag heeft gesteld, Wanneer de backend analyse (Neo4j + LLM) succesvol is voltooid, Dan verschijnt er een AI-gegenereerd antwoord in de chatgeschiedenis dat relevant is voor de gestelde vraag.
* **AC-AI-04-02 (Loading Indicator):** Gegeven dat de gebruiker een vraag heeft gesteld, Wanneer de backend bezig is met het verwerken van het antwoord, Dan wordt er een visuele indicator (bijv. spinner, "denken..." bericht) getoond aan de gebruiker.

**FR-GRAPH-06 (Detectie Problemen - bijv. Flow Inconsistenties):**

* **AC-GRAPH-06-01 (Start Analyse):** Gegeven dat er een succesvol gepopuleerde graph bestaat voor snapshot XYZ, Wanneer de gebruiker de "Analyze" stap start voor snapshot XYZ, Dan wordt een set van vooraf gedefinieerde analyse-queries (inclusief queries voor flow inconsistenties) uitgevoerd op de Neo4j database voor die snapshot.
* **AC-GRAPH-06-02 (Detectie - Positief Scenario):** Gegeven dat snapshot XYZ een Flow bevat die verwijst naar een veld dat *niet* bestaat in dezelfde snapshot, Wanneer de analyse-queries worden uitgevoerd, Dan identificeert de relevante query deze inconsistentie.
* **AC-GRAPH-06-03 (Detectie - Negatief Scenario):** Gegeven dat snapshot XYZ alleen Flows bevat die naar bestaande componenten verwijzen, Wanneer de analyse-queries worden uitgevoerd, Dan identificeert de flow-inconsistentie query *geen* problemen voor dit type fout.
* **AC-GRAPH-06-04 (Resultaat Structuur):** Gegeven dat een inconsistentie is gedetecteerd, Dan levert de analyse-query een resultaat op dat minimaal identificeert: de betreffende Flow (bijv. naam/ID), het type inconsistentie (bijv. 'Missing Field Reference'), en het betreffende niet-bestaande component (bijv. veld API naam).

**FR-RSLT-01 (Opslag Analyse Resultaten):**

* **AC-RSLT-01-01 (Opslag na Analyse):** Gegeven dat de "Analyze" stap is voltooid voor snapshot XYZ, Wanneer er inconsistenties zijn gedetecteerd door FR-GRAPH-06, Dan worden deze resultaten opgeslagen in een daarvoor bestemde tabel (bijv. analysis\_results) in Supabase.
* **AC-RSLT-01-02 (Koppeling Data):** Gegeven dat een analyseresultaat is opgeslagen, Dan bevat het opgeslagen record minimaal een verwijzing naar het snapshot\_id (XYZ), het type analyse (bijv. 'Flow Inconsistency Check'), een ID/naam van het problematische component (bijv. Flow ID), een beschrijving van het probleem, en een timestamp.
* **AC-RSLT-01-03 (Geen Resultaten):** Gegeven dat de "Analyze" stap is voltooid voor snapshot XYZ en er *geen* inconsistenties zijn gedetecteerd, Dan worden er geen fout-gerelateerde records aangemaakt in de analysis\_results tabel voor die analyse run (of er wordt een record opgeslagen met status 'No issues found').

**FR-UI-08 (Weergave Analyse Resultaten):**

* **AC-UI-08-01 (Beschikbaarheid Resultaten):** Gegeven dat de "Analyze" stap is voltooid voor snapshot XYZ en de resultaten zijn opgeslagen (FR-RSLT-01), Wanneer de gebruiker de resultaten voor snapshot XYZ opvraagt in de UI (bijv. via een 'Analyse Resultaten' tab/sectie), Dan worden de opgeslagen resultaten opgehaald en weergegeven.
* **AC-UI-08-02 (Weergave Lijst):** Gegeven dat er gedetecteerde inconsistenties zijn voor snapshot XYZ, Dan toont de UI een lijst van deze problemen, met per probleem minimaal de naam/ID van het betreffende component (bijv. Flow naam) en de beschrijving van het probleem.
* **AC-UI-08-03 (Duidelijkheid):** Gegeven dat de lijst met problemen wordt getoond, Dan is deze lijst overzichtelijk en begrijpelijk voor de gebruiker.
* **AC-UI-08-04 (Geen Resultaten Weergave):** Gegeven dat er *geen* inconsistenties zijn gedetecteerd voor snapshot XYZ, Dan toont de UI een duidelijke melding dat er geen problemen zijn gevonden voor deze analyse.
* **AC-UI-08-05 (Navigatie - Optioneel V1+):** Gegeven dat een probleem wordt getoond in de lijst, Wanneer de gebruiker op het probleem klikt, Dan navigeert de UI (indien mogelijk) naar het betreffende component in de graph visualisatie of toont meer details.

**FR-AI-02 FR-AI-03 FR-AI-04 (AI Assistant testen)**

* **AC-AI-02-01 (Query Vertaling - Impact Vraag):** Gegeven dat de gebruiker vraagt "Wat gebeurt er als ik veld MyObject\_\_c.MyField\_\_c verwijder?", Wanneer het systeem de vraag interpreteert (FR-AI-02), Dan genereert het een Neo4j Cypher query die zoekt naar alle componenten die direct of indirect afhankelijk zijn van (:Field {name: 'MyField\_\_c'})-[:BELONGS\_TO]->(:Object {name: 'MyObject\_\_c'}).
* **AC-AI-03-01 (LLM Input):** Gegeven dat de Neo4j query voor een impactanalyse een set afhankelijke componenten retourneert, Wanneer het systeem de LLM aanroept (FR-AI-03), Dan bevat de prompt voor de LLM minimaal: de oorspronkelijke vraag, de gesimuleerde wijziging (verwijderen veld X), en een gestructureerde lijst van de gevonden afhankelijke componenten en hun types.
* **AC-AI-04-03 (Geen Impact Gevonden):** Gegeven dat de gebruiker een impactvraag stelt voor een component waarvoor de Neo4j query geen afhankelijkheden vindt, Wanneer het systeem het antwoord presenteert (FR-AI-04), Dan bevat het antwoord een duidelijke melding dat er (volgens de huidige graph) geen impact is gevonden.
* **AC-AI-04-04 (Foutafhandeling LLM):** Gegeven dat de LLM API een fout retourneert of timed out tijdens de verwerking van een vraag, Wanneer het systeem het antwoord probeert te presenteren, Dan toont de UI een duidelijke foutmelding aan de gebruiker (bijv. "Kon geen antwoord genereren, probeer het later opnieuw") en logt de fout intern.