THOMAS MORE

Realisatie stage Continuum-Jidoka

AI Implementatie in de interne Knowledge Checker applicatie.

2023 - 2024

Voorwoord

Het afronden van mijn stage bij Continuum Consulting was een ongelooflijke reis vol groei en leren. Ik wil mijn oprechte dank uitspreken aan iedereen die deze ervaring onvergetelijk heeft gemaakt. Allereerst mijn mentor, Tibeau Vandenbroeck, voor zijn voortdurende begeleiding en ondersteuning bij de frontend-ontwikkeling. Ook wil ik Bas Serrachia bedanken voor zijn hulp met de backend en AI-implementatie. Verder bedank ik iedereen die meewerkte aan de stages voor hun algemene hulp, en natuurlijk alle collega's bij het bedrijf voor de leuke sfeer en de warme ontvangst. Dankzij jullie allen heb ik enorm veel geleerd en genoten van mijn tijd bij Continuum-Jidoka. Bedankt!

Inhoud

1. Inleiding

In dit bestand zal ik de realisaties van mijn stage tonen, in dit eerste hoofdstuk zal ik mijn bedrijf voorstellen en mijn opdracht, de “timeline” van mijn stage uitleggen.

1.1. Continuum-Jidoka

Ik heb mijn stage is bij het Consultancy bedrijf “Continuum-Jidoka” gedaan. Hun specialisatie is Java en Spring development en ze tellen een 200-tal werknemers. Ze hebben twee kantoren, een in Hasselt en een in Mechelen.

De hoofdwaarden van het bedrijf zijn kwaliteit, innovatie en lange termijn relaties. Bij Continuum gaat het niet alleen om wat ze doen, maar vooral om waarom ze dat doen. Hun missie, “Raising the bar for Java Software Development”, is het hogere doel dat alle beslissingen bij Continuum drijft.

Ook geloven ze sterk in het delen van kennis en het opbouwen van een ‘community of experts’. Dit is al direct te merken in mijn stage door de meerdere invites naar “Knowledge nights” in mijn inbox, hierbij geven interne / externe specialisten presentaties over hun kennis. Of hun “brown bags” waar kennis gedeeld word tijdens de middag lunch. Het bedrijf gelooft dat alleen door kennis te delen en actief deel te nemen aan een bredere ‘community’, iedereen kan blijven leren en groeien.

Bovendien is Continuum-Jidoka deel van de Allan Allman Groep, een ecosysteem van verschillende consultancy bedrijven. Het kantoor van Continuum is ook gedeeld met drie andere bedrijven uit deze groep (in Hasselt). Dit biedt de mogelijkheid voor referenties aan mogelijke klanten door andere bedrijven binnen deze groep. Ook zijn Continuum in 2022-23 gemerged met JArchitects en Jidoka. De officiële naam van Continuum is nu dus ook “Continuum-Jidoka”.

2. Stage Opdracht – Knowledge Checker

De opdracht voor mijn stage was voor AI te implementeren in een interne applicatie genaamd de “Knowledge Checker”. Deze applicatie dient als een tool voor interview vragen te centraliseren binnen het bedrijf want op dit moment heeft iedere interviewer zijn elke lijst van interview vragen.

Voor de AI-implementatie zijn er 2 gevraagde items: het genereren van nieuwe interview vragen en een verificatie systeem voor deze vragen. Ook zou ik voor de AI-implementatie een onderzoek moeten doen van welke technologien ik hiervoor zou gebruiken.

2.1.1. Plan van aanpak

Voor school moest ik in het begin van mijn stage een plan van aanpak schrijven, hierin heb ik de staat van het project bekeken om te kijken wat ik concreet allemaal zou kunnen maken in mijn stage. Na de huidige designs te bekijken en applicatie te bekijken had ik al een item extra gevonden voor mijn stage: het filter design implementeren. Ook zou ik in Figma nieuwe designs maken voordat ik begin met developen.

De lijst van items die ik wou realiseren waren na mijn plan van aanpak dus:

- Nieuwe designs maken
- Verificatie systeem implementeren
- AI-onderzoek
- Vragen generatie systeem implementeren
- Vragen hergeneratie systeem implementeren

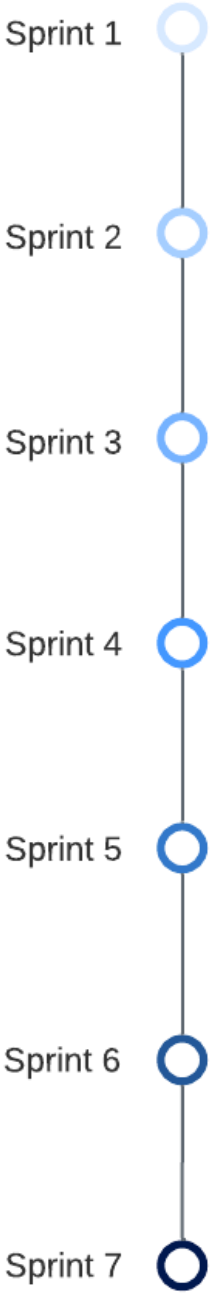
2.1.2. Business Case / Business Doel van de Opdracht

Op dit moment hanteert elke interviewer zijn eigen lijst met vragen, wat kan leiden tot inconsistentie en inefficiëntie. Het hoofddoel van deze stageopdracht is om AI-functionaliteiten te implementeren die het proces van vraaggeneratie en validatie automatiseren en optimaliseren. Zodat wanneer de applicatie online gezet word deze vragen sneller en consistenter ingegeven kunnen worden.

Door AI te integreren in de Knowledge Checker, kan het bedrijf dus een aantal belangrijke voordelen realiseren:

- Consistentie en Kwaliteit: De AI kan helpen bij het genereren van consistente en kwalitatieve interviewvragen, waardoor de variatie in vraagstellingen tussen verschillende interviewers vermindert en een uniforme standaard wordt gehandhaafd.
- Efficiëntie en Tijdsbesparing: Het gebruik van AI om vragen te genereren en te verifiëren bespaart tijd voor de interviewers, die anders handmatig vragen zouden moeten samenstellen en controleren.
- Innovatie: Het inzetten van geavanceerde technologieën zoals AI geeft het bedrijf een innovatief karakter, wat kan bijdragen aan een concurrentievoordeel in de markt door een modern en efficiënt wervingsproces.
- Kwaliteitscontrole: Het verificatiesysteem zorgt ervoor dat de gegenereerde vragen voldoen aan de gestelde eisen en relevant zijn voor de functieprofielen, wat de kwaliteit van de sollicitatiegesprekken verbetert.

3. Timeline stage



Sprint 1

De staat van het project bekijken op een grotere schaal en nieuwe designs, een plan van aanpak maken.

Sprint 2

Het verificatie systeem van vragen er (grotendeels) inzetten zodat ik bekend kon worden met het project.

Sprint 3

Verificatie systeem verder afwerken en het begin van het AI onderzoek, welke modellen kan ik gebruiken en welke frameworks zijn er op de markt?

Sprint 4

Een POC maken van de AI implementatie met Spring AI

Sprint 5

Een ADR en tweede POC met Langchain4j voor een RAG implementatie

Sprint 6

Het begin van de AI implementatie met Langchain4j en het afwerken van het verificatie systeem.

Sprint 7

Halve sprint waarin ik finishing touches heb gemaakt en mijn stage heb gedocumenteerd

4. Technology Stack

In dit deel zal ik de uiteindelijke technology stack van het project bespreken, maar ook de andere technologieën die ik aangeraakt hebt tijdens mijn stage.

4.1.1. Knowledge Checker Stack

Voor de knowledge checker heb ik aan beide de front-end en back-end gewerkt tijdens mijn stage.

Het front-end gedeelte was geschreven in Angular 14 met nx monorepos, ik ben niet veel in aanraking gekomen met het nx gedeelte maar heb veel code

TECHSTACK FRONTEND

- Angular
- Angular-Material
- SCSS
- (HTML / Typescript)
- Nx Monorepos (Weinig in aanraking meegekomen!)

De backend was geschreven in Java | Springboot. De versie van Java was v17 en springboot was initiaal 2.7.x maar deze heb ik geupdate naar springboot 3.2.4.

TECHSTACK BACKEND

- Sprint Boot
- Spring Framework
- Langchain4j (v0.29)
- DynamoDB
- Spotless

4.1.2. Spring training

Tijdens mijn stage heb ik ook meegedaan aan een 4-daagse spring training, hier hebben we van de vrij snel het Spring Framework geleerd van de basis tot de meer geavanceerde delen.

De technologien die ik hiervan heb meegenomen zijn:

- Spring Boot
- Spring Framework
- Spring + H2 DB
- Spring + MySQL DB
- Flyway (MySQL)
- ArchUnit

En natuurlijk de coding practices heb ik veel coding practices geleerd/overgenomen tijdens deze training.

4.1.3. POC Stacks

Voor het AI onderzoek deel van mijn stage heb ik 2 POCs gemaakt, een POC met Spring AI waar ik met OpenAI Question objects zou generen.

TECHSTACK POC SPRING AI

- Springboot
- Spring + DynamoDB
- SpringAI (v0.8.1)
- OpenAI, Ollama (mistral, llama2)

Ook heb ik een POC gemaakt met langchain4j, hier heb ik ook RAG in geïmplementeerd voor relevante categories; tags te vinden voor een vraag.

TECHSTACK POC LANGCHAIN4J

- Springboot
- Spring + DynamoDB
- Langchain4j (v0.29)
- Azure OpenAI
- ChromaDB
- Embeddings-all-minilm-l6-v2

4.1.4. Andere technologieën

Voor de designs heb ik tijdens mijn stage zoals eerder vermeld Figma gebruikt, maar op aanrading van mijn mentor heb ik dit later ook gebruikt voor mijn demos en presentaties te gebruiken. Waardoor ik heel goed de voor en na situatie kon laten zien d.m.v. de nieuwe en oude designs in de presentatie te zetten en gebruik te maken van Smart Animations.

Voor documentatie heb ik dit vooral in Markdown geschreven, met enkele diagrammen geschreven in MermaidJS.

Het project was opgeslaan op GitHub, met een gelinkt GitHub project waar het Kanban bord op stond. In het GitHub project stonden ook de projecten van andere stagiaires. Voor hen heb ik ook code reviews gedaan.

Het knowledge checker project werd ook gemigreerd van DynamoDB naar MongoDB in de laatste maand van mijn stage, aan deze code ben ik niet veel komen en is pas gemerged in de laatste week van mijn stage. Maar ik heb hier wel veel code reviews voor gedaan.s

4.1.5. Agile

Tijdens mijn stage hebben we een agile driven development proces gevolgd. Met sprints van 2 weken waarin we feature/ticket-driven development doen. Elke sprint begon op de eerste maandag met een retro van de vorige sprint en een planning voor de komende sprint. Elke dag hadden we in de middag een standup meeting voor alle stagiaire projecten. Met een stagiaire als Scrum master. En elke laatste vrijdag met een demo waar iedereen van het bedrijf bij mocht zijn.

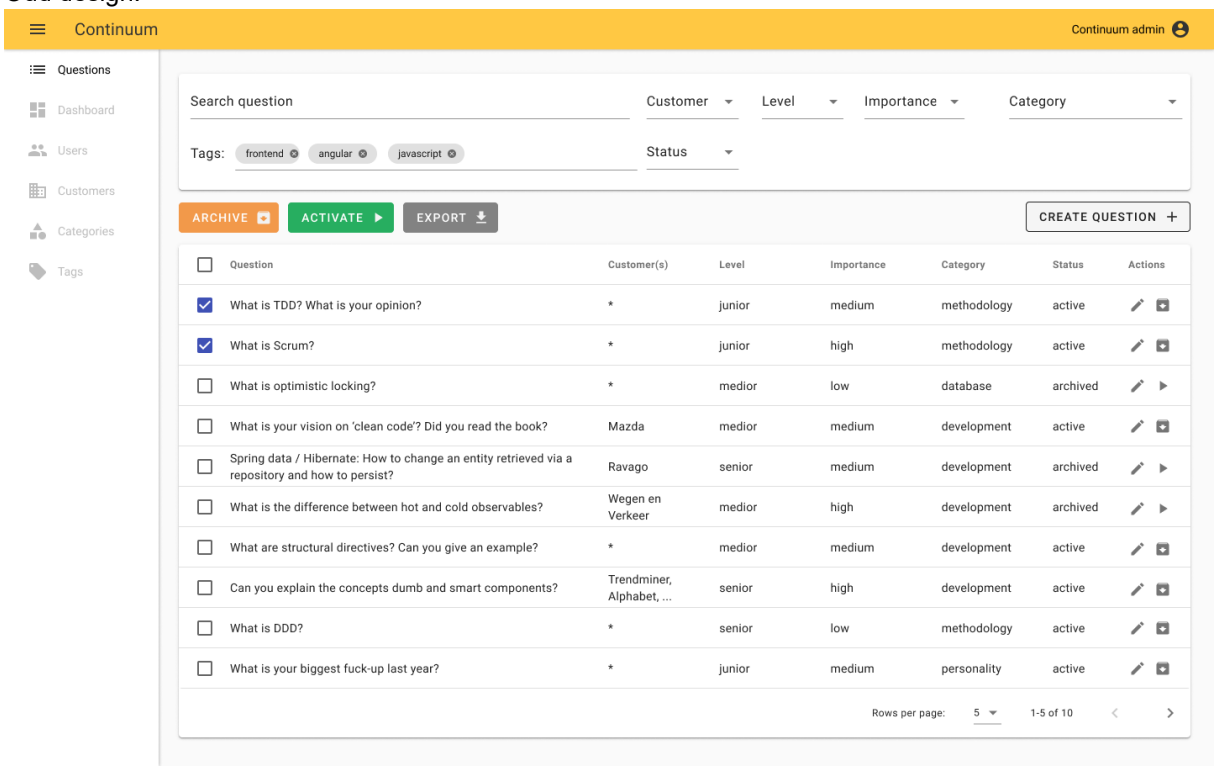
4.1.6. Concepten

Door het werken aan mijn stageopdracht heb ik ook me ook verdiept in nieuwe concepten. Enkel van de belangrijkste zijn:

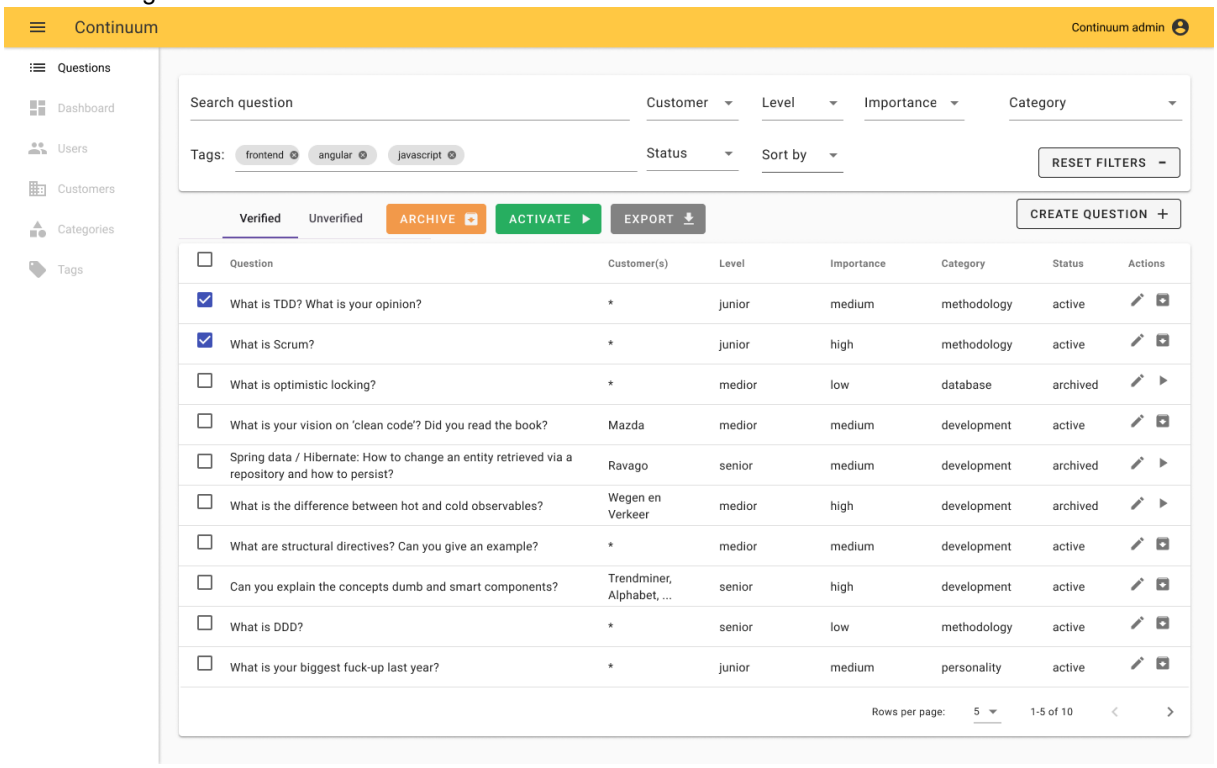
- Ticket-Driven Development
- Feature-Driven Development
- Test Driven Development
- Unit testing
- Behaviour testing
- Developing with LLMs
- RAG

5. Ontwerpen

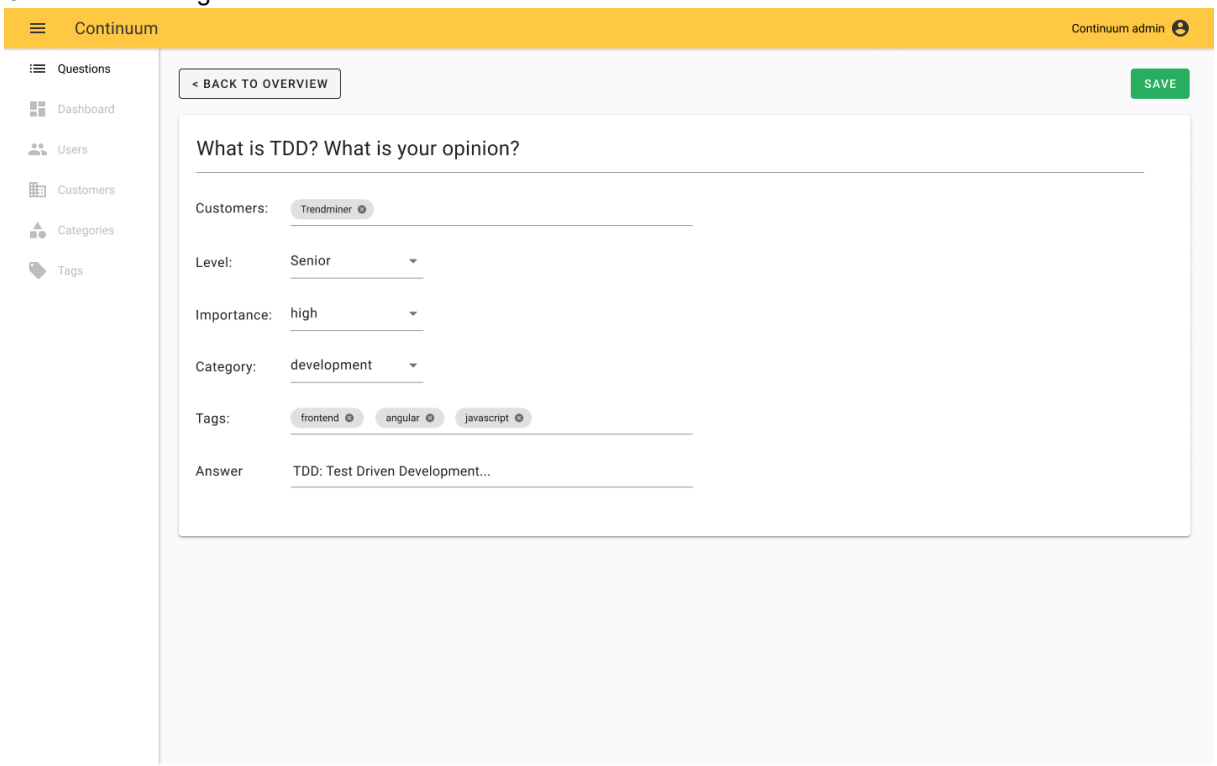
In de eerste Sprint van mijn stage heb ik mijn focus gelegd op het maken van nieuwe Figma designs voor de applicatie. Voor de Questions Page heb ik aan het design een tab toegevoegd en een reset filter knop:
Oud design:



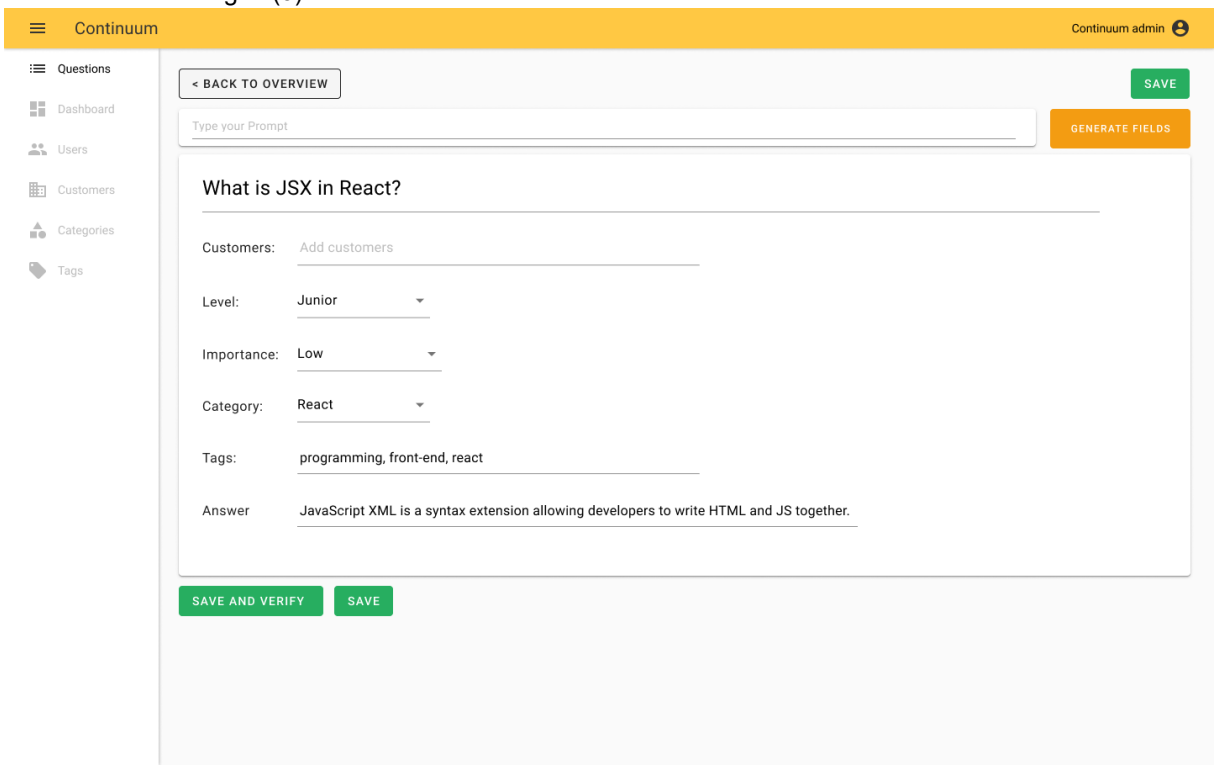
Nieuwe design

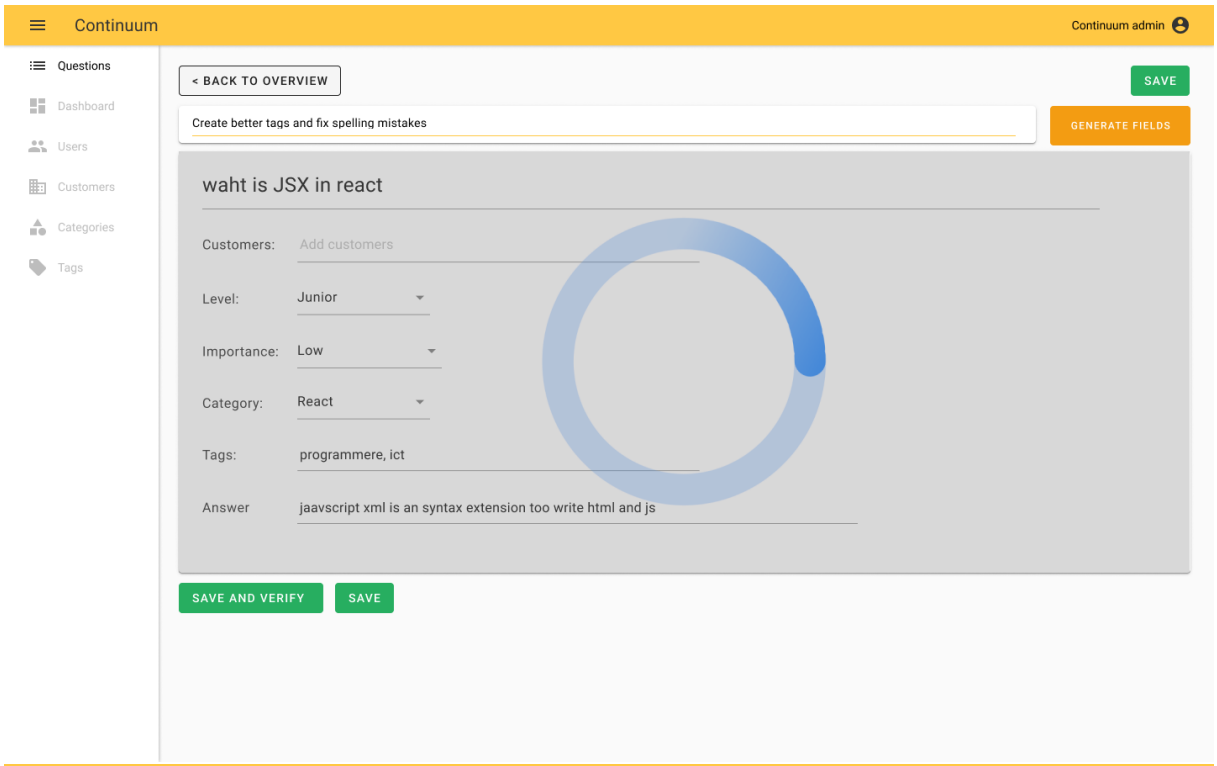


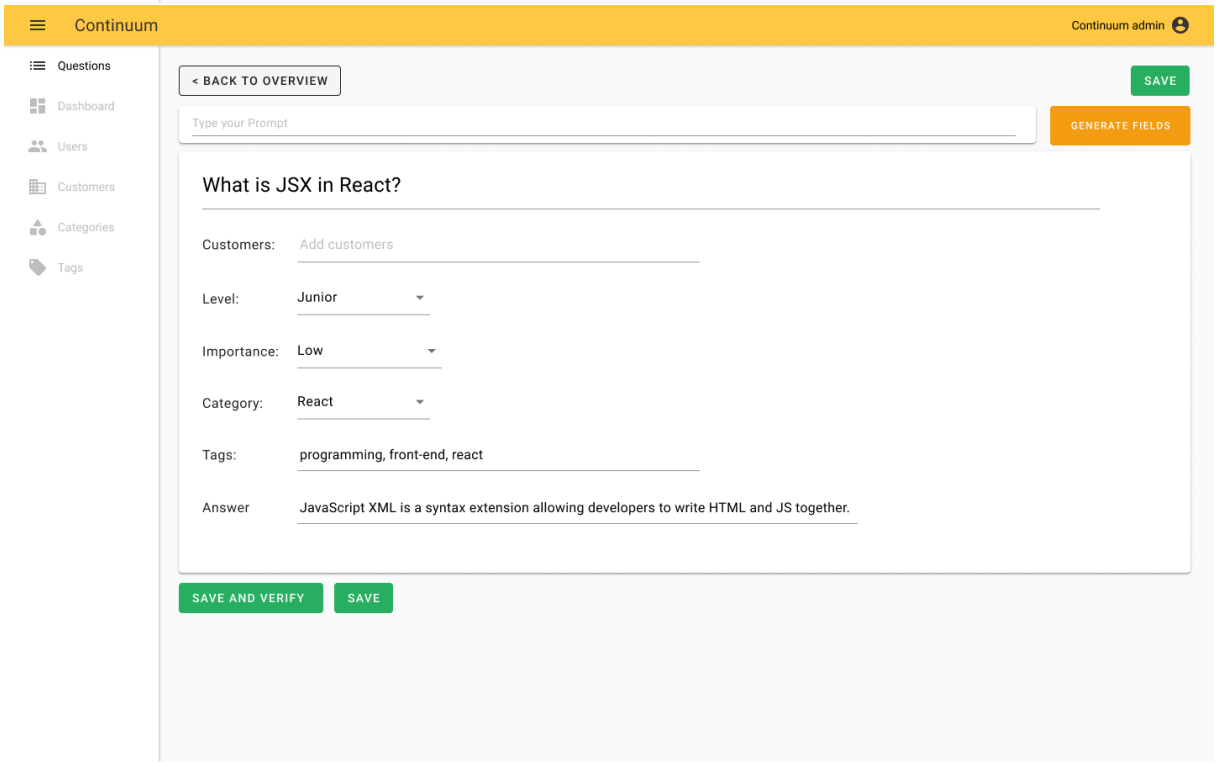
Oude create designs:



Nieuwe create Designs (3):

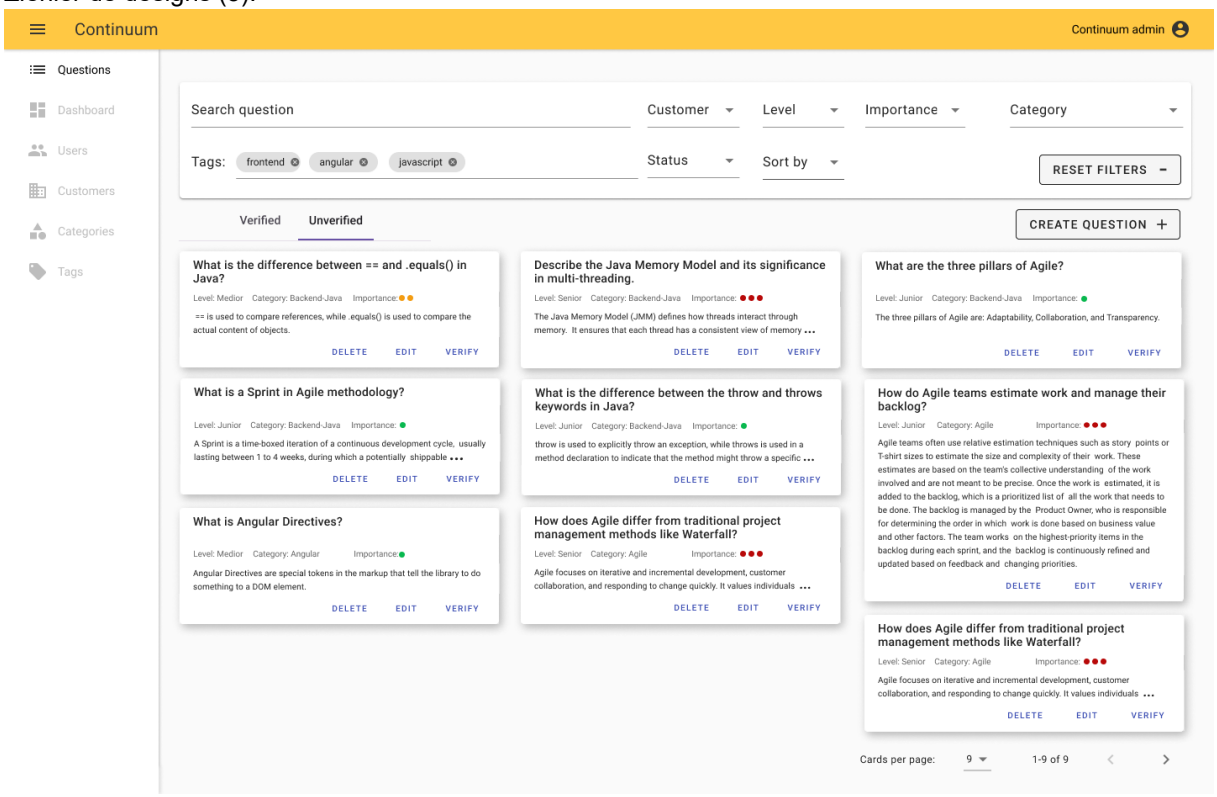


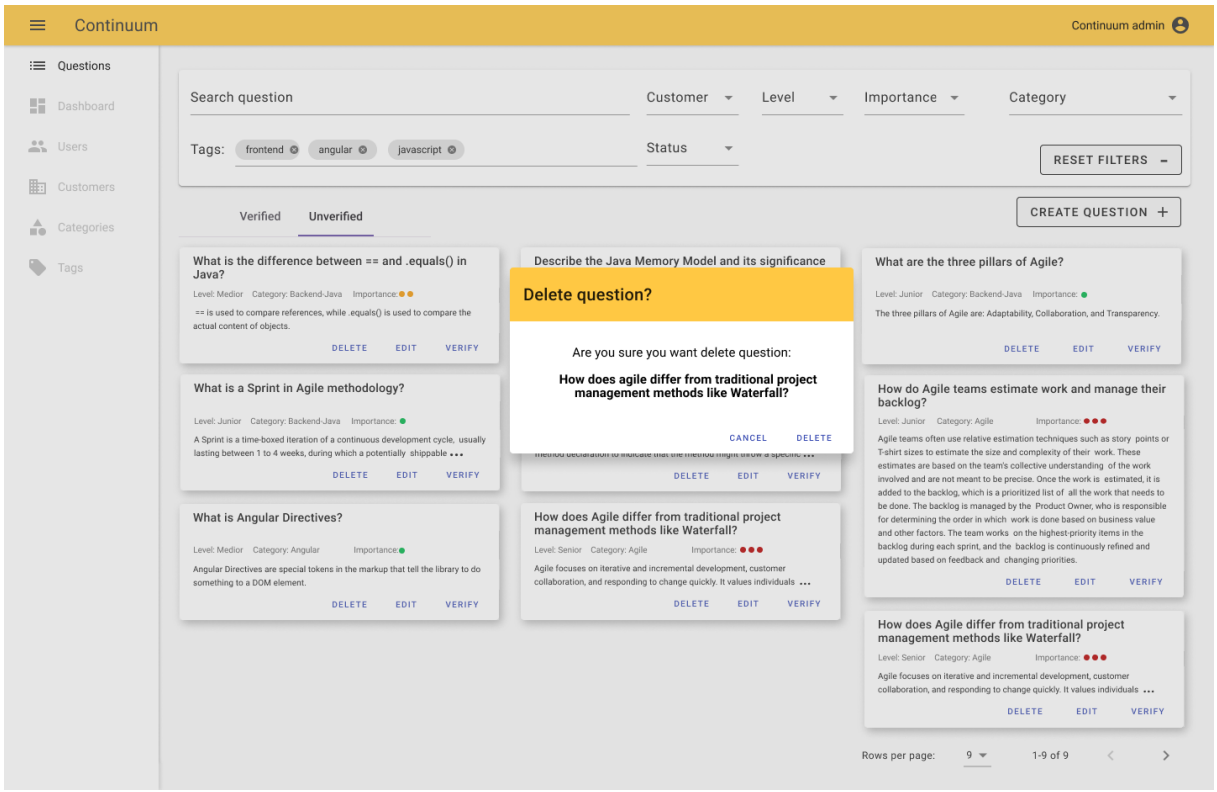

Continuum
Continuum admin
Questions
Dashboard
Users
Customers
Categories
Tags
< BACK TO OVERVIEW
SAVE
Create better tags and fix spelling mistakes
GENERATE FIELDS
waht is JSX in react
Customers:
Add customers
Level:
Junior
Importance:
Low
Category:
React
Tags:
programmere, ict
Answer
jaavscript xml is an syntax extension too write html and js
SAVE AND VERIFY
SAVE

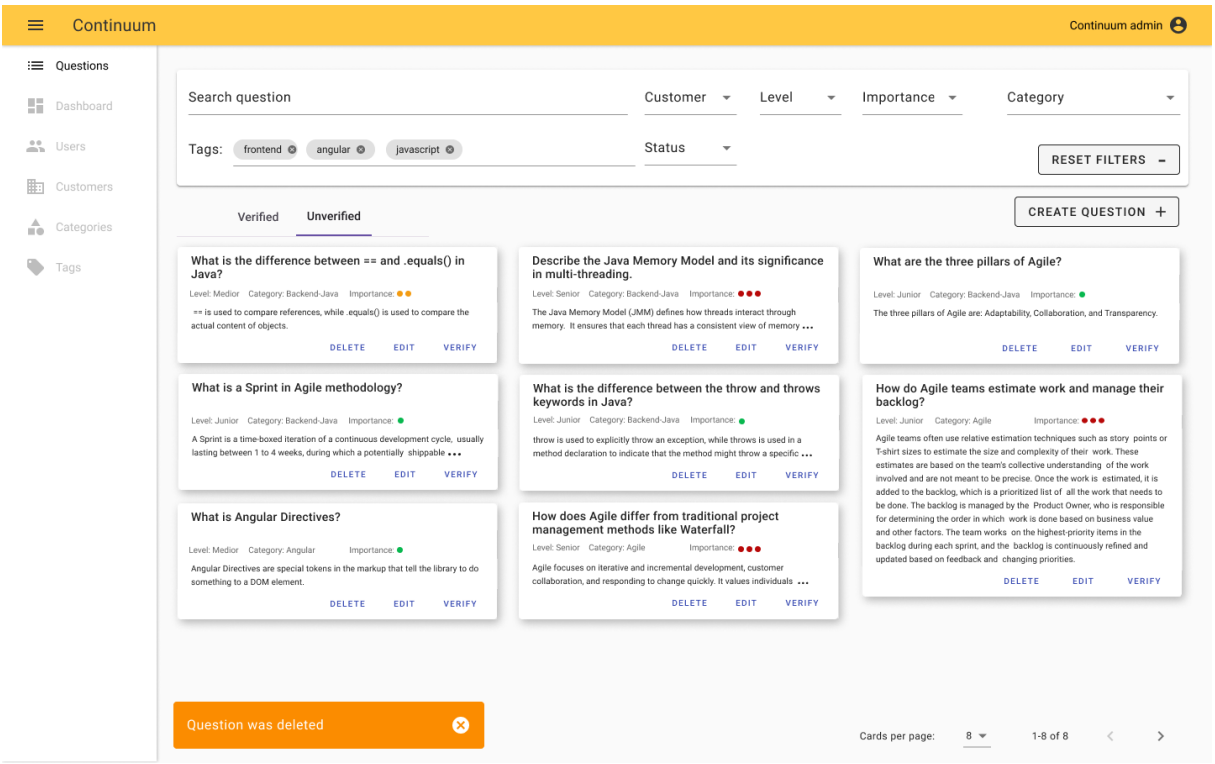

Continuum
Continuum admin
Questions
Dashboard
Users
Customers
Categories
Tags
< BACK TO OVERVIEW
SAVE
Type your Prompt
GENERATE FIELDS
What is JSX in React?
Customers:
Add customers
Level:
Junior
Importance:
Low
Category:
React
Tags:
programming, front-end, react
Answer
JavaScript XML is a syntax extension allowing developers to write HTML and JS together.
SAVE AND VERIFY
SAVE

Voor het unverified tab heb ik ook een nieuw design gemaakt. Hierin heb ik de cards zelf gedesigned.en niet die van Material gebruikt.

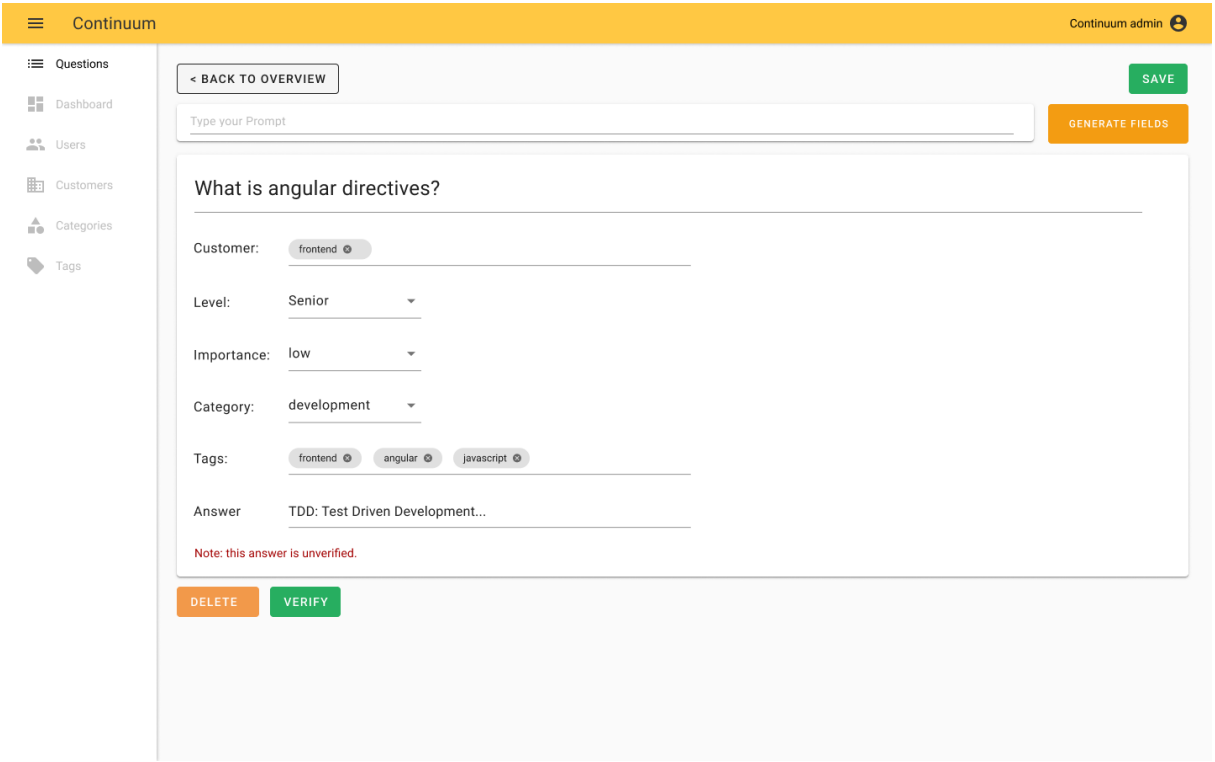
Ziehier de designs (3):



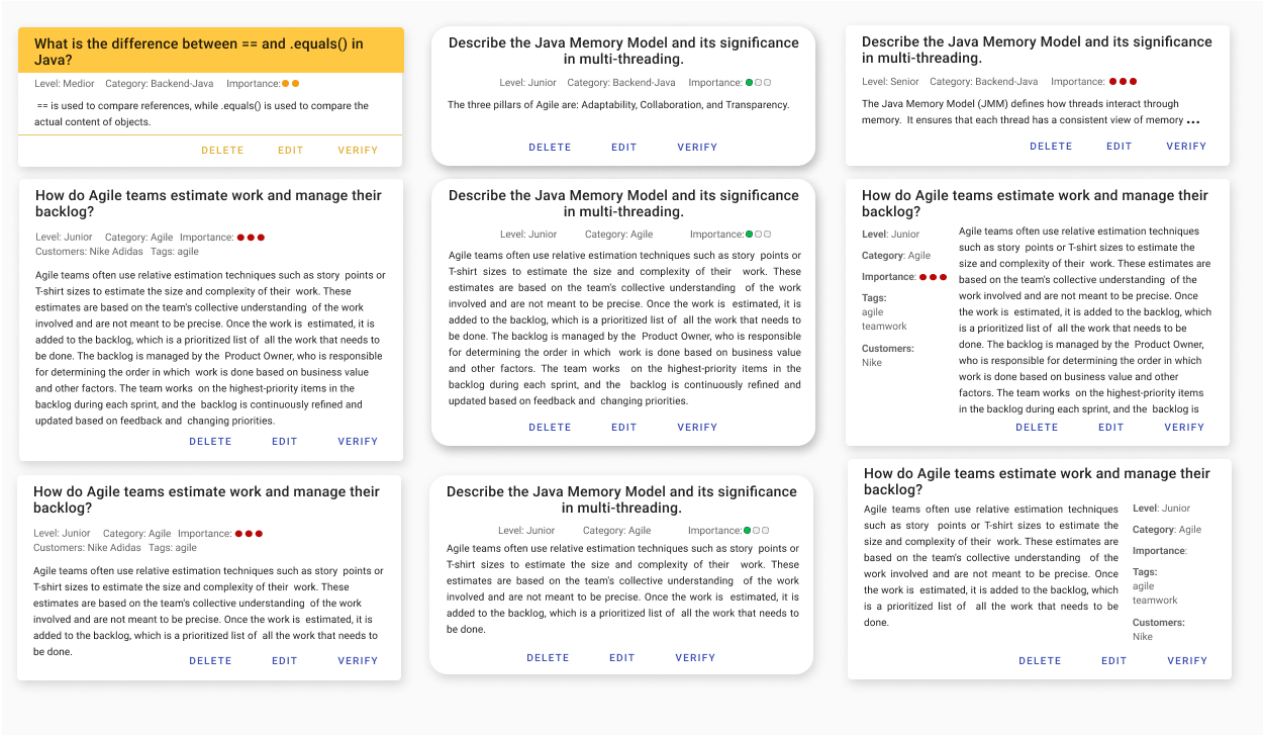




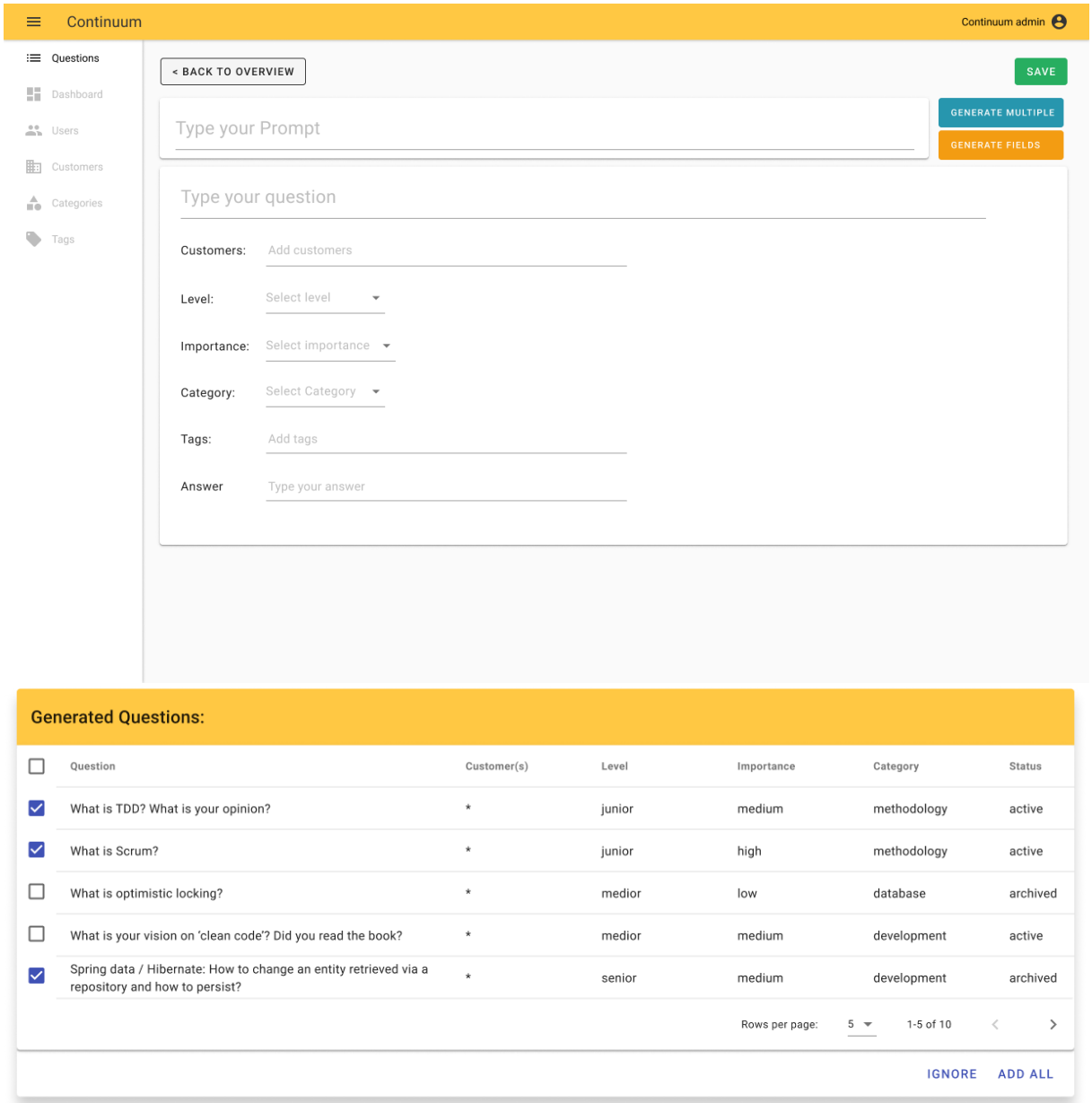
Ik heb ook voor het edit screen een nieuwe “variant” gemaakt als je een unverified card gaat editen (1):



Ik heb ook prototype card designs gemaakt (1):

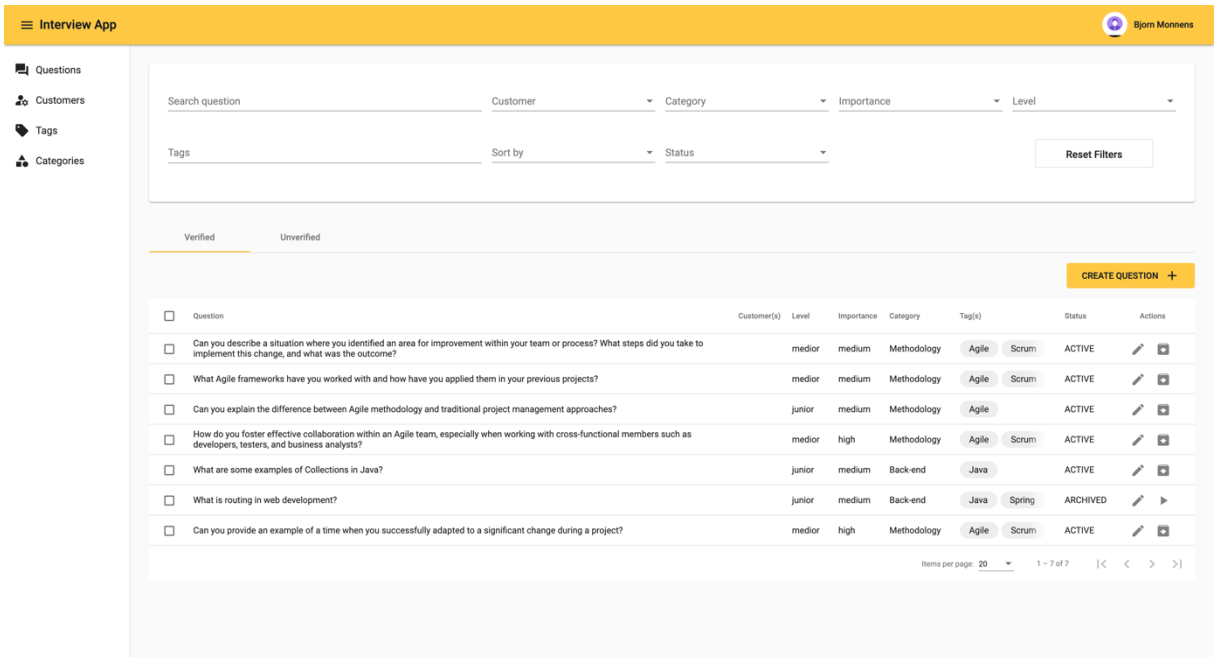

What is the difference between == and .equals() in Java?
Level: Medior Category: Backend-Java Importance:
== is used to compare references, while .equals() is used to compare the actual content of objects.
DELETE EDIT VERIFY
Describe the Java Memory Model and its significance in multi-threading.
Level: Junior Category: Backend-Java Importance:
The three pillars of Agile are: Adaptability, Collaboration, and Transparency.
DELETE EDIT VERIFY
Describe the Java Memory Model and its significance in multi-threading.
Level: Senior Category: Backend-Java Importance:
The Java Memory Model (JMM) defines how threads interact through memory. It ensures that each thread has a consistent view of memory ...
DELETE EDIT VERIFY
How do Agile teams estimate work and manage their backlog?
Level: Junior Category: Agile Importance:
Customers: Nike Adidas Tags: agile
Agile teams often use relative estimation techniques such as story points or T-shirt sizes to estimate the size and complexity of their work. These estimates are based on the team's collective understanding of the work involved and are not meant to be precise. Once the work is estimated, it is added to the backlog, which is a prioritized list of all the work that needs to be done. The backlog is managed by the Product Owner, who is responsible for determining the order in which work is done based on business value and other factors. The team works on the highest-priority items in the backlog during each sprint, and the backlog is continuously refined and updated based on feedback and changing priorities.
DELETE EDIT VERIFY
Describe the Java Memory Model and its significance in multi-threading.
Level: Junior Category: Agile Importance:
Agile teams often use relative estimation techniques such as story points or T-shirt sizes to estimate the size and complexity of their work. These estimates are based on the team's collective understanding of the work involved and are not meant to be precise. Once the work is estimated, it is added to the backlog, which is a prioritized list of all the work that needs to be done. The backlog is managed by the Product Owner, who is responsible for determining the order in which work is done based on business value and other factors. The team works on the highest-priority items in the backlog during each sprint, and the backlog is continuously refined and updated based on feedback and changing priorities.
DELETE EDIT VERIFY
How do Agile teams estimate work and manage their backlog?
Level: Junior
Category: Agile
Importance:
Tags: agile teamwork
Customers: Nike
Agile teams often use relative estimation techniques such as story points or T-shirt sizes to estimate the size and complexity of their work. These estimates are based on the team's collective understanding of the work involved and are not meant to be precise. Once the work is estimated, it is added to the backlog, which is a prioritized list of all the work that needs to be done. The backlog is managed by the Product Owner, who is responsible for determining the order in which work is done based on business value and other factors. The team works on the highest-priority items in the backlog is
DELETE EDIT VERIFY
How do Agile teams estimate work and manage their backlog?
Level: Junior Category: Agile Importance:
Customers: Nike Adidas Tags: agile
Agile teams often use relative estimation techniques such as story points or T-shirt sizes to estimate the size and complexity of their work. These estimates are based on the team's collective understanding of the work involved and are not meant to be precise. Once the work is estimated, it is added to the backlog, which is a prioritized list of all the work that needs to be done.
DELETE EDIT VERIFY
Describe the Java Memory Model and its significance in multi-threading.
Level: Junior Category: Agile Importance:
Agile teams often use relative estimation techniques such as story points or T-shirt sizes to estimate the size and complexity of their work. These estimates are based on the team's collective understanding of the work involved and are not meant to be precise. Once the work is estimated, it is added to the backlog, which is a prioritized list of all the work that needs to be done.
DELETE EDIT VERIFY
How do Agile teams estimate work and manage their backlog?
Agile teams often use relative estimation techniques such as story points or T-shirt sizes to estimate the size and complexity of their work. These estimates are based on the team's collective understanding of the work involved and are not meant to be precise. Once the work is estimated, it is added to the backlog, which is a prioritized list of all the work that needs to be done.
Level: Junior
Category: Agile
Importance:
Tags: agile teamwork
Customers: Nike
DELETE EDIT VERIFY

En een prototype design voor het massa genereren van vragen. (2)

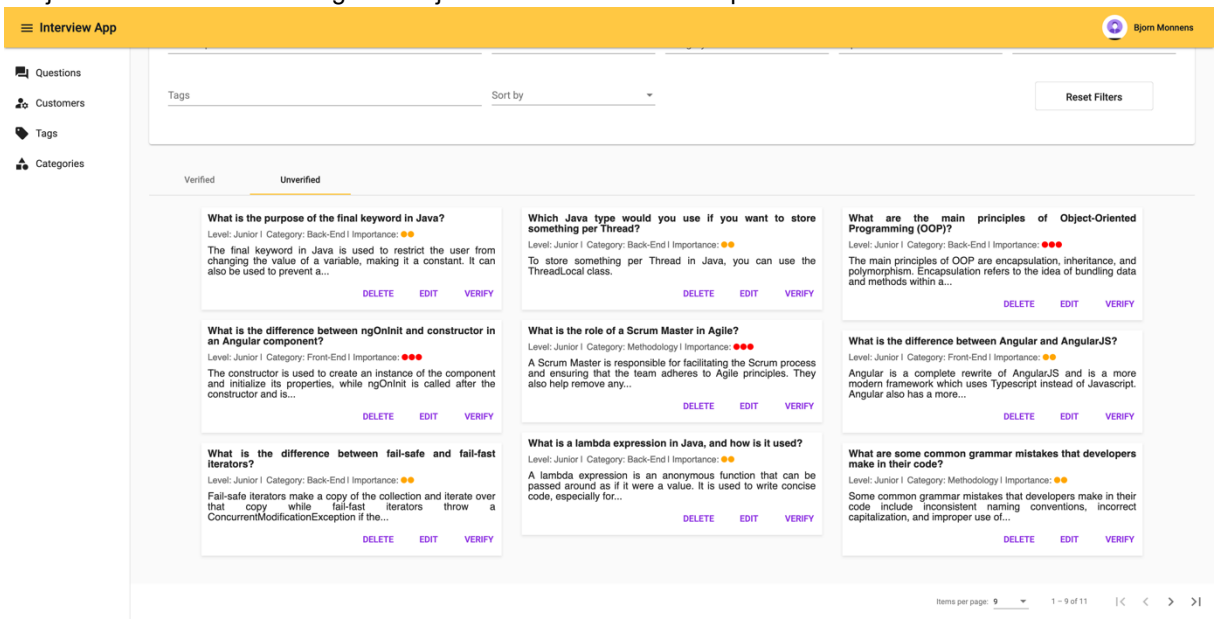


6. Verificatie Systeem (uitleg)

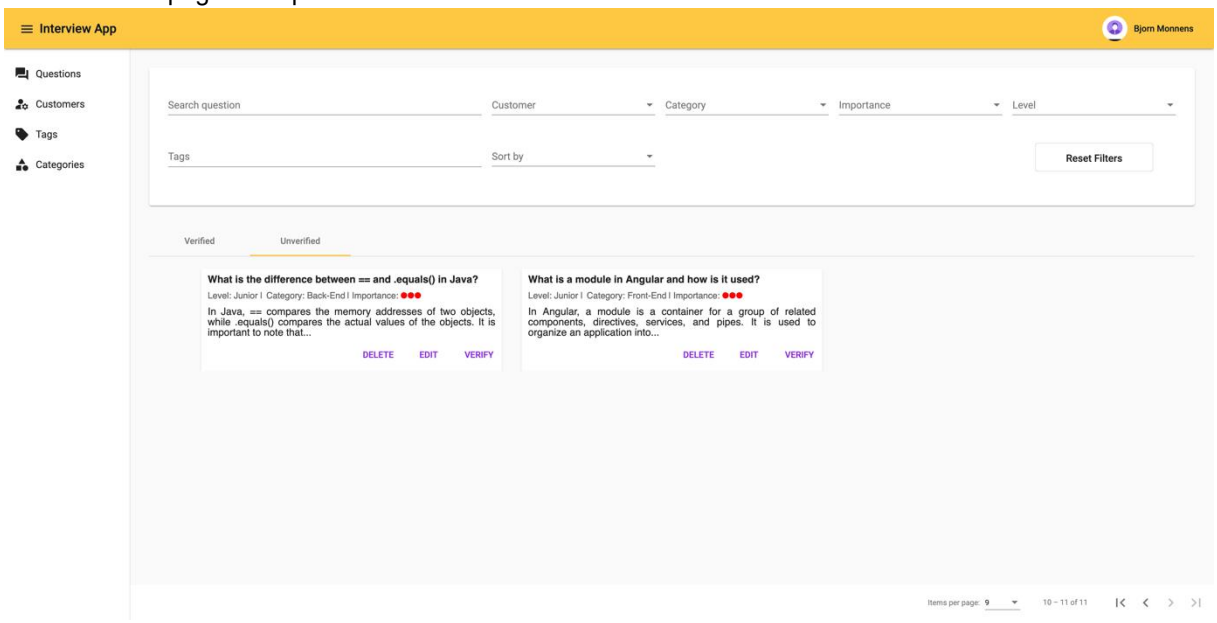
Het verificatie systeem heb ik als volgt geimplementeerd, eerst heb je 2 tabs bovenaan.



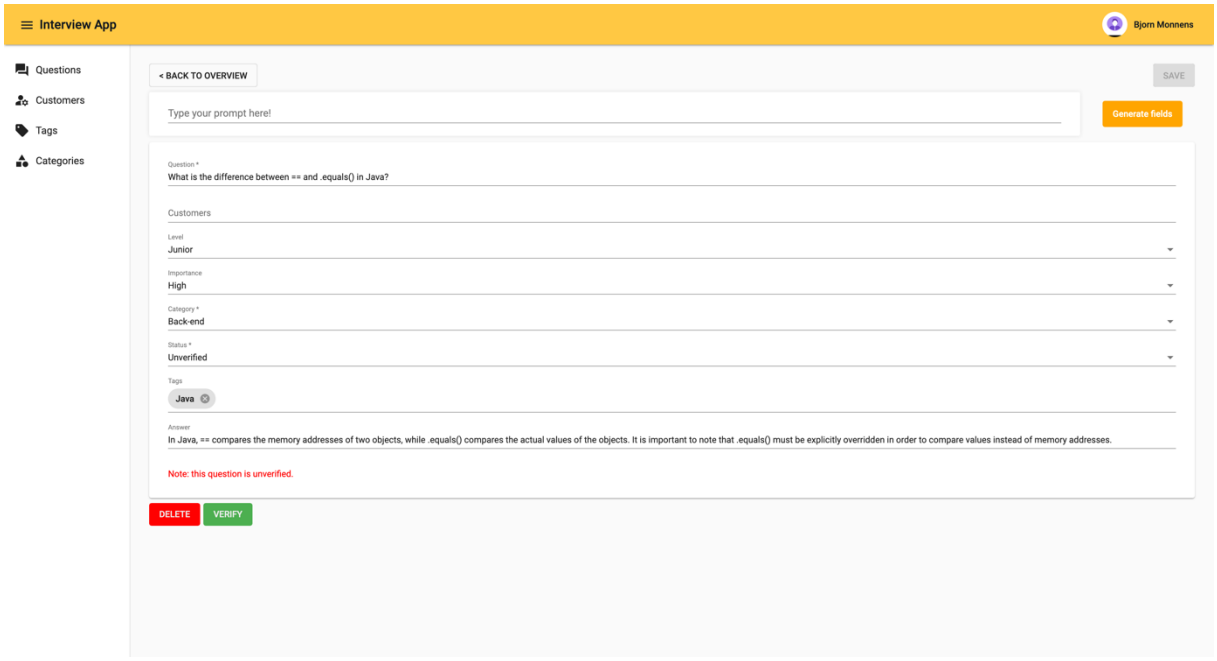
Als je naar het unverified tab gaat kan je de unverified interview questions zien:



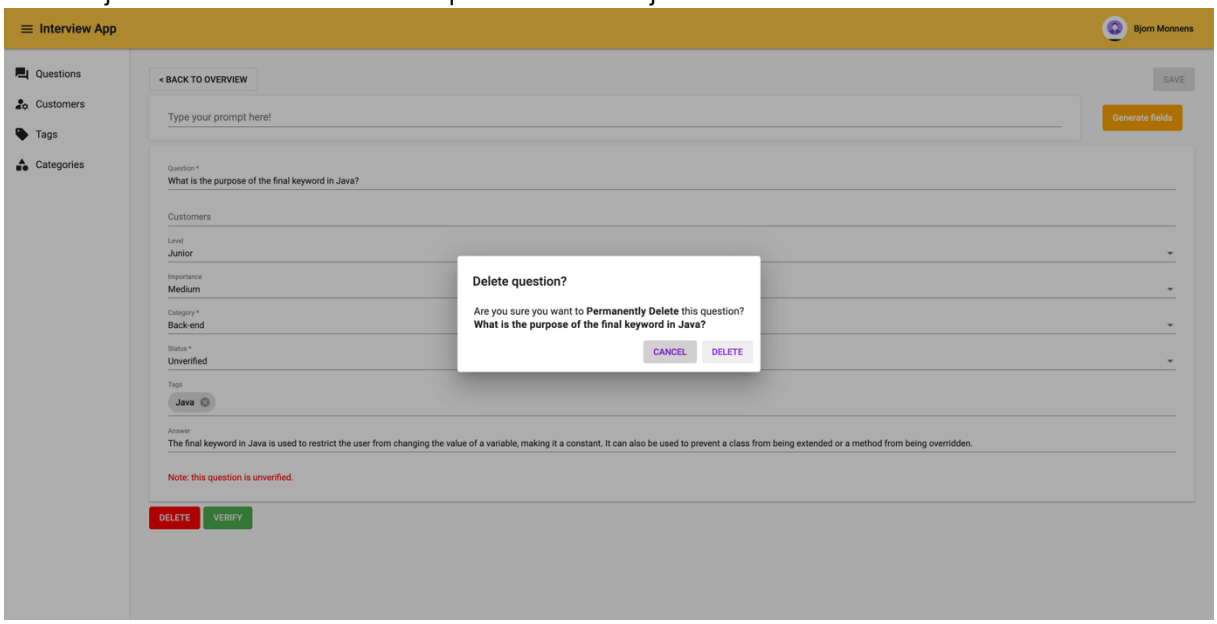
Hier staat ook paginatie op:



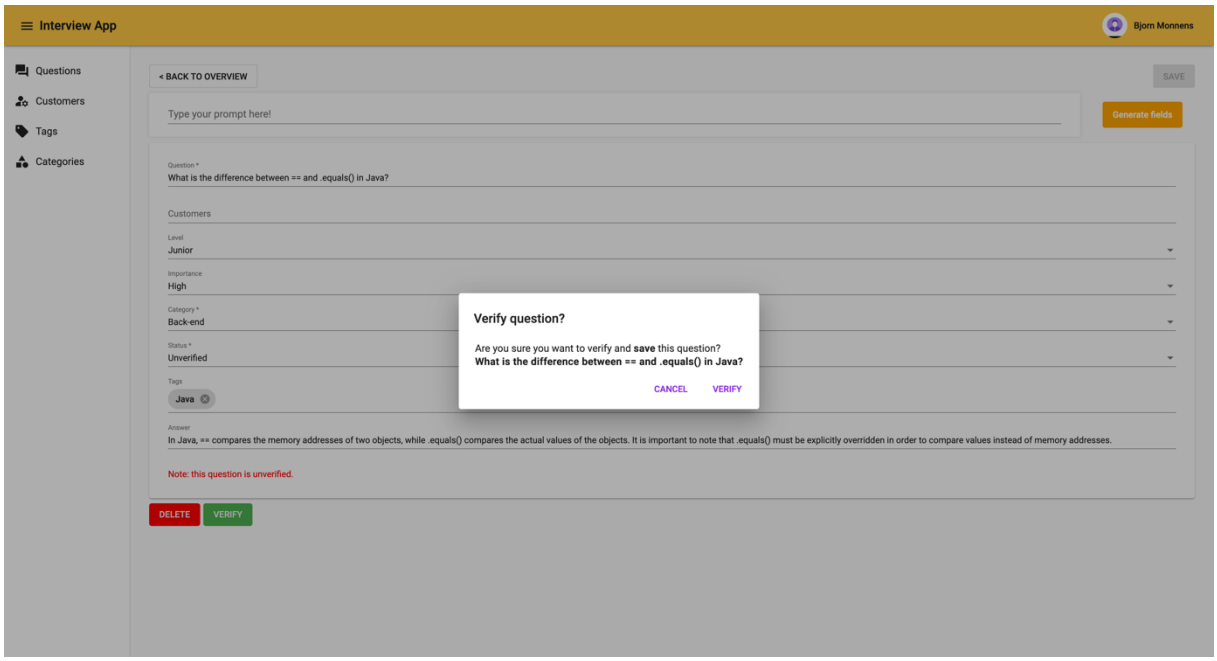
Je kan via de kaart de question deleten, verifieren of editten. In dit geval klikken we op edit en krijgen we het unverified edit schermt te zien:



Hier kan je dan de unverified interview question bvb verwijderen.



Of verifieren.



7. AI Onderzoek (uitleg)

In het AI-onderzoek, heb ik een Proof of Concept (POC) ontwikkeld om met Spring AI te experimenteren. Mijn focus lag op het genereren, hergenereren en massaal genereren van interviewvraag objecten. Na het succes van deze fase, rees er interesse in Retrieval Augmented Generation (RAG). Echter, vanwege twijfels over de haalbaarheid hebben we besloten opnieuw onderzoek te doen, wat resulteerde in een Architectuur-Decision Record (ADR).

7.1. POC Spring AI

De initiële POC was gebaseerd op een "barebones" kopie van het originele project, waarbij alleen de benodigde code en dependencies werden overgenomen. Dit betekende bijvoorbeeld het weglaten van delen zoals Spring Security en de CustomerController, aangezien deze niet relevant waren voor de AI-toepassing. Deze “barebones” kopie leidde tot de ontdekking dat het project moest worden gemigreerd naar de volgende grote release van Spring (van 2 naar 3). Deze POC was een groot success, het liet zien dat ons concept van interviewvragen generatie zou werken, en nadat de “barebones” kopie gemigreerd werd naar Spring versie 3.2.4 werd ook het hoofdproject gemigreerd naar deze nieuwere verie.

Hierna heb ik een ADR opgesteld waarin ik de frameworks Spring AI, Langchain4j en Langchain heb vergeleken, samen met de belangrijkste factoren voor het maken van de AI-implementatie in ons project met RAG (meer info over deze term in de volgende titel). Het resultaat werd samengevat in een ADR-matrix, waarbij Langchain4j op dat moment aanzienlijk beter scoorde voor RAG dan Spring AI.

Criterias	Spring AI	Langchain4j	Langchain	Weights
RAG Capabilities	3	5	5	3
Output parsing Capabilities	4	3	5	1
Ease of implementation	4	3	0	3
(Community) Support	0	5	5	3
Market share	3	4	5	1
Experience with the framework	3	1	0	1
Documentation	2	4	5	2
Integration with Spring (/ Java)	5	2	0	2
Total	24	27	25	
Total with weights	45	59	52	

ADR: 

Architecture Decision Record - AI Framework.pdf (Command Line)

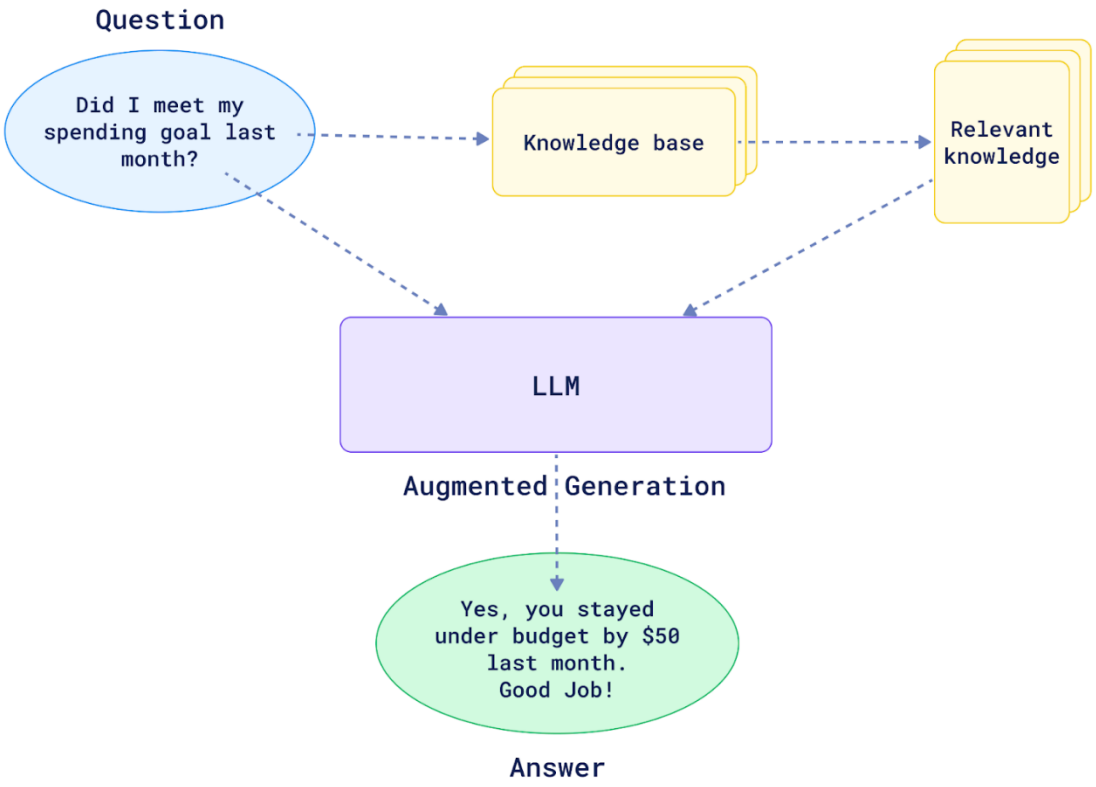
7.2. POC Langchain4j + RAG

Retrieval Augmented Generation (RAG) combineert het ophalen van relevante informatie uit een dataset met het genereren van nieuwe tekst op basis van die informatie. Het haalt eerst contextuele informatie op en gebruikt die vervolgens om nauwkeurige en samenhangende tekst te produceren. Dit resulteert in betere gegenereerde inhoud.

Een eenvoudig voorbeeld van RAG, dat ik tijdens mijn stage heb ontwikkeld, is een Spring AI API waarin ik de 'Survival Guide' PDF van Continuum-Jidoka heb opgeslagen in een in-memory vector database. Telkens wanneer er een verzoek naar de AI wordt gestuurd, wordt relevante informatie uit de vector database gehaald en meegestuurd naar de AI. Hierdoor kan men met de AI communiceren over de Survival Guide, waarbij de AI de juiste context heeft.

Voor de POC met RAG wilde ik het concept van Retrieval Augmented Generation toepassen door alle vraagobjecten op te slaan in een vector database, waarna ik de relevante tags en categorieën kan ophalen voor elke nieuwe intervraag die gegenereerd moet worden. Helaas bleek dit minder succesvol te zijn omdat ik RAG toepaste op JSON in plaats van normale tekst waarin meer context aanwezig zou zijn, iets wat cruciaal is voor RAG.

Desondanks was de tweede Proof of Concept met Langchain4j een succes. Ik ben voor deze POC opnieuw gestart met de "barebones" kopie van het originele project en kon deze POC bijna rechtstreeks implementeren in de main-codebase van het project.



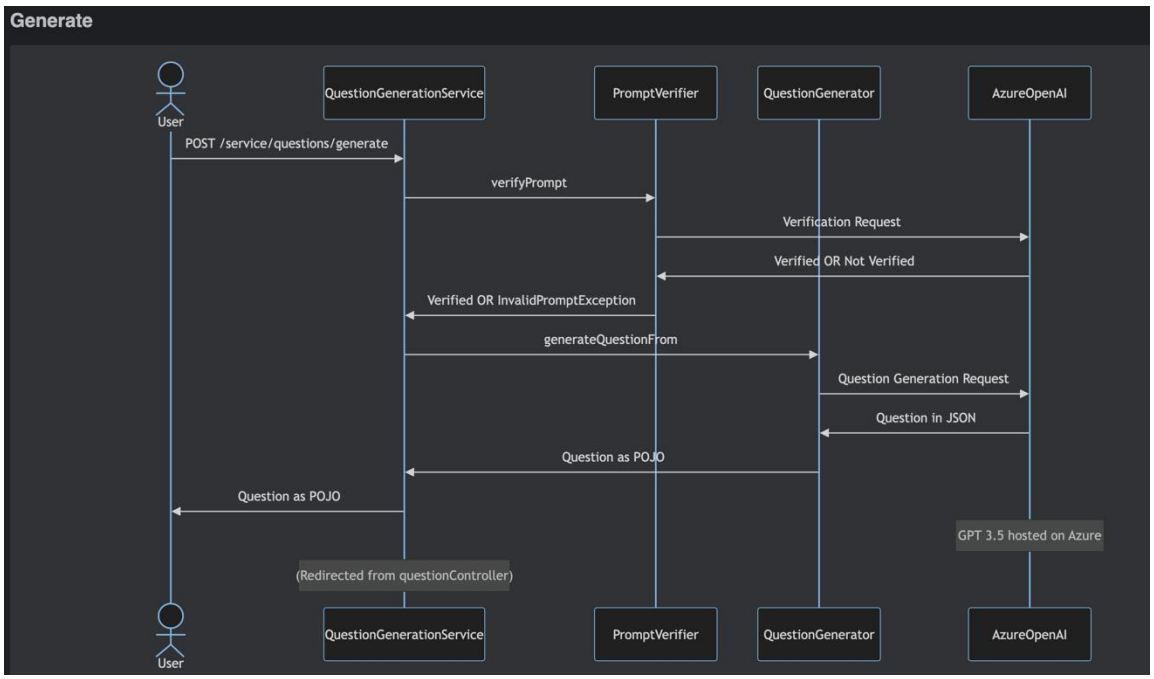
*Deze diagram heb ik niet zelf gemaakt en heb ik erbij gezet enkel voor uitleg.

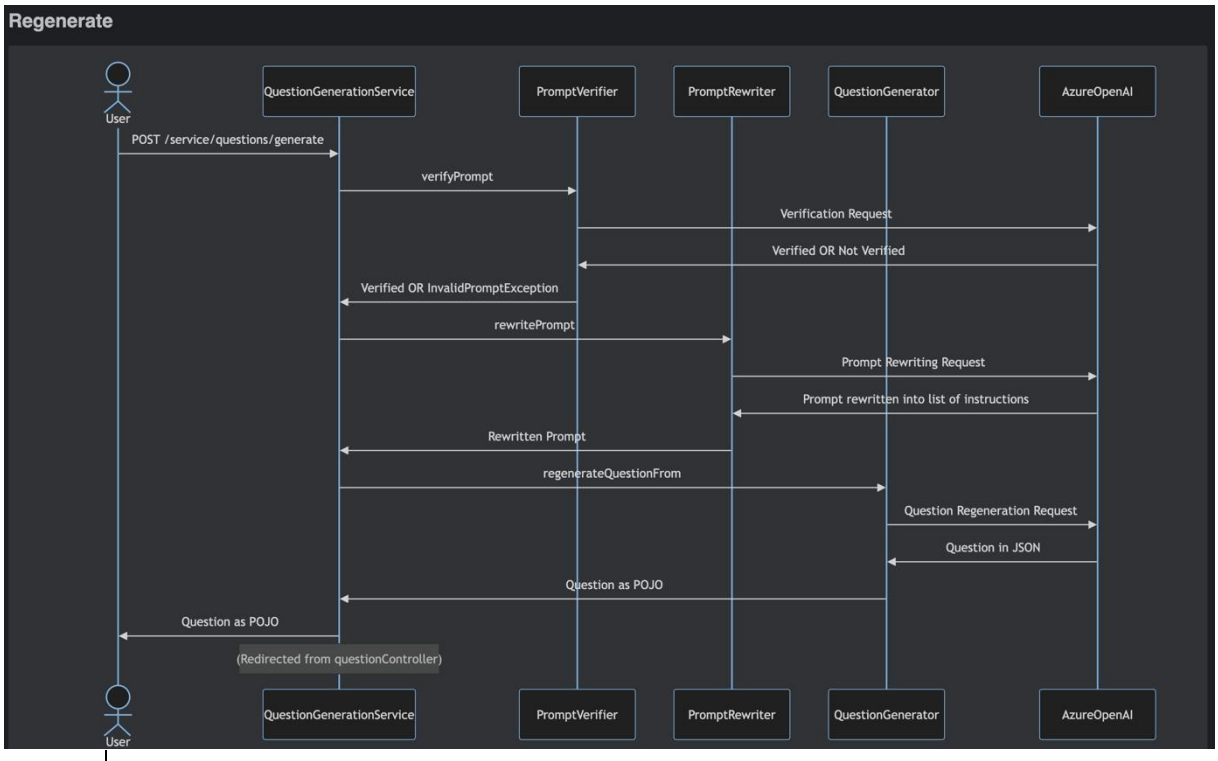
8. AI Implementatie | Vragen generatie

Omdat ik ervoor heb gekozen om voor de uiteindelijke AI-implementatie met Langchain4j te werken, heb ik gebruik moeten maken van Langchain4j's AiServices. Deze AiServices zijn een soort chatbots die je kunt aanmaken als Spring Beans met Langchain4j en vervolgens kunt gebruiken om calls naar de AI te maken.

Voor het genereren van vragen heb ik twee AiServices gemaakt: de PromptVerifier en de QuestionGenerator. De PromptVerifier controleert of de gebruiker een veilige prompt instuurt, en de QuestionGenerator genereert een interviewquestion object op basis van de prompt van de gebruiker.

Voor het herschrijven heb ik hier nog een derde AiService tussen geplaatst: de PromptRewriter. Deze zorgt ervoor dat de prompt wordt omgezet in een lijst met instructies, wat bijdraagt aan een beter hergeneratieresultaat.

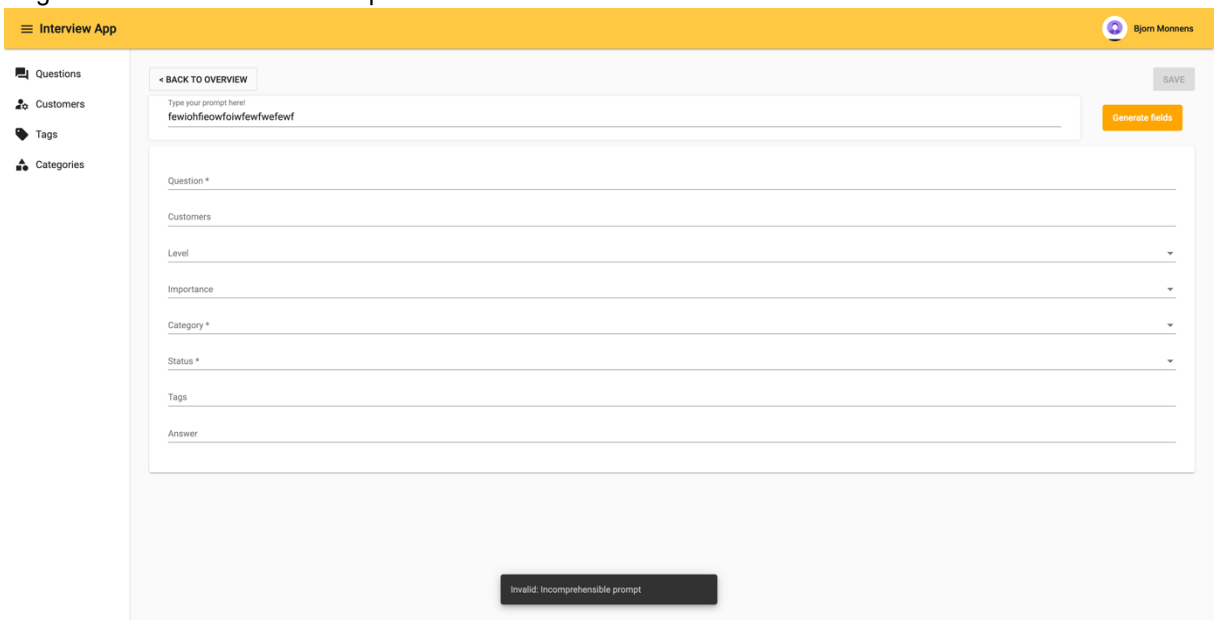




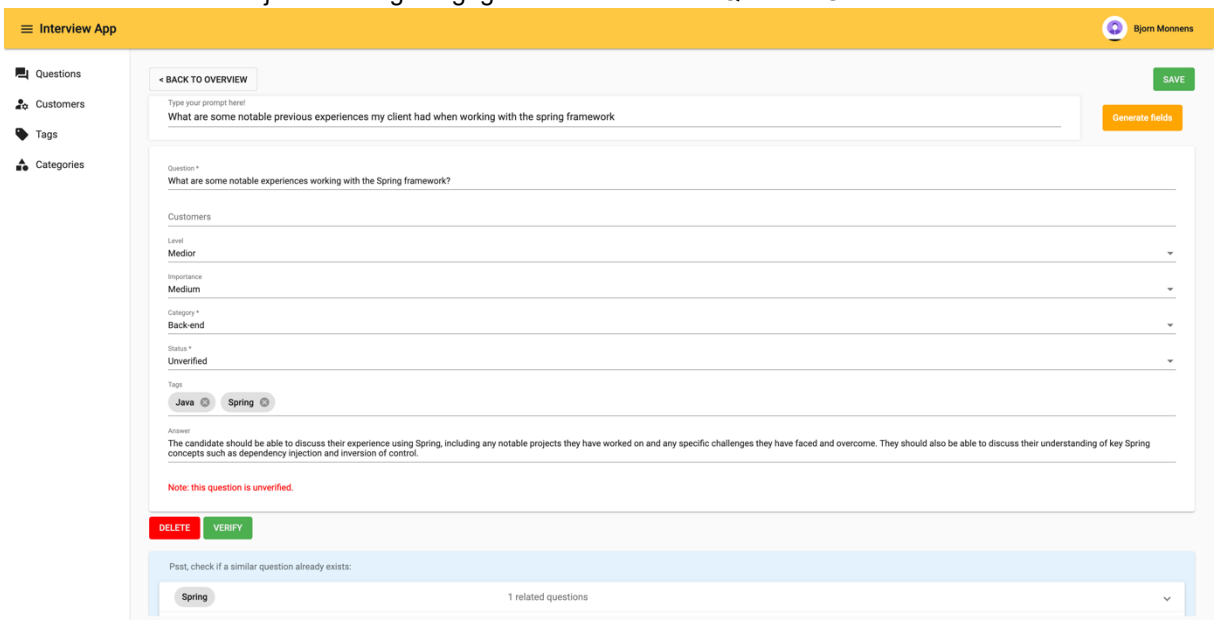
De uiteindelijke AI-implementatie zag er dus zo uit:
In deze screenshot kan je het design geïmplementeerd in de front-end zien:

In deze screenshot kan je de PromptVerifier in actie zien:

Nog een voorbeeld van de PromptVerifier:



In deze screenshot zie je een vraag die gegenereerd is door de QuestionGenerator:



CONTACT

Brent Simons
brentsimons1@gmail.com

VOLG ONS

https://brentsimons.github.io/bachelor-portfolio/
https://www.linkedin.com/in/brent-simons-5b3867252/

THOMAS MORE