



# Verbetering van Data-Export Functionaliteit voor Grootschalige Grids

Realisatie

Bachelor in de Informatica  
keuzerichting Application development

Boons Jens

Academiejaar 20xx-20xx

Campus Geel, Kleinhoefstraat 4, BE-2440 Geel

|            |  |           |
|------------|--|-----------|
| <b>1</b>   | <b>ABSTRACT .....</b>                                    | <b>4</b>  |
| <b>2</b>   | <b>DANKWOORD .....</b>                                   | <b>5</b>  |
| <b>3</b>   | <b>TERMINOLOGIE.....</b>                                 | <b>6</b>  |
| <b>4</b>   | <b>PROJECT .....</b>                                     | <b>7</b>  |
| <b>4.1</b> | <b>Inleiding .....</b>                                   | <b>7</b>  |
| <b>4.2</b> | <b>Wie is Persion Architects?.....</b>                   | <b>8</b>  |
| 4.2.1      | Innovatie en Groei .....                                 | 8         |
| 4.2.2      | Toewijding aan Klantgerichtheid.....                     | 8         |
| 4.2.3      | Toekomstvisie .....                                      | 8         |
| <b>4.3</b> | <b>Project Beschrijving .....</b>                        | <b>9</b>  |
| 4.3.1      | Vorbereiding en Planning: .....                          | 9         |
| 4.3.2      | Datamodeling en Review: .....                            | 9         |
| 4.3.3      | Ontwikkeling van Prototipe Frontend: .....               | 9         |
| 4.3.4      | Implementatie van Backend Functionaliteiten: .....       | 9         |
| 4.3.5      | Aanpassing van Dispatcher en Flow van het Project: ..... | 9         |
| 4.3.6      | Testen en Validatie: .....                               | 9         |
| <b>5</b>   | <b>GEBRUIKTE TECHNOLOGIEËN .....</b>                     | <b>10</b> |
| <b>5.1</b> | <b>Technologieën.....</b>                                | <b>10</b> |
| 5.1.1      | Java .....   | 10        |
| 5.1.2      | Apache Tapestry .....                                    | 10        |
| 5.1.3      | Maven .....  | 10        |
| 5.1.4      | API.....   | 10        |
| 5.1.5      | JavaScript .....   | 10        |
| 5.1.6      | Liquibase .....  | 10        |
| 5.1.7      | Hibernate.....   | 11        |
| 5.1.8      | MySQL.....   | 11        |
| 5.1.9      | Apache POI .....   | 11        |
| 5.1.10     | Unit Test.....   | 11        |
| 5.1.11     | Tomcat .....   | 11        |
| 5.1.12     | Jenkins .....  | 11        |
| <b>5.2</b> | <b>Programma's/ Platformen .....</b>                     | <b>12</b> |
| 5.2.1      | Figma .....  | 12        |
| 5.2.2      | StarUML.....   | 12        |
| 5.2.3      | Git .....  | 12        |
| 5.2.4      | Draw.io.....   | 12        |
| <b>5.3</b> | <b>Methodologieën .....</b>                              | <b>12</b> |
| 5.3.1      | 4.3.1 SCRUM .....  | 12        |
| 5.3.2      | Code Review .....  | 12        |
| <b>6</b>   | <b>REALISATIE .....</b>                                  | <b>14</b> |
| <b>6.1</b> | <b>Analyse van Bestaande Projecten en Software .....</b> | <b>14</b> |
| <b>6.2</b> | <b>Prototype en Datamodeling.....</b>                    | <b>14</b> |
| 6.2.1      | Prototype Ontwikkeling .....                             | 14        |
| 6.2.2      | Datamodeling .....                                       | 16        |
| <b>6.3</b> | <b>Backend en Front-end Ontwikkeling .....</b>           | <b>17</b> |
| 6.3.1      | Volledige Project Flow .....                             | 17        |
| 6.3.2      | Database Structuur .....                                 | 18        |
| 6.3.3      | Nieuwe Webpagina voor File Management .....              | 19        |
| 6.3.4      | Export Grid knop.....                                    | 20        |
| 6.3.5      | Tasks en Jobs Implementatie.....                         | 21        |
| 6.3.5.1    | Jobscheduler .....                                       | 22        |
| 6.3.5.2    | Gridtask.....  | 24        |

|            |   |           |
|------------|---|-----------|
| 6.3.6      | Dispatcher en Dispatch Messages.....    | 26        |
| 6.3.6.1    | Dispatcher .....                        | 27        |
| 6.3.7      | API.....                                | 28        |
| <b>6.4</b> | <b>Ophalen van output files .....</b>   | <b>29</b> |
| 6.4.1      | Ophalen .....                           | 29        |
| 6.4.2      | Notificaties .....                      | 30        |
| 6.4.3      | Download Functionaliteit .....          | 31        |
| <b>6.5</b> | <b>Integratie en Testen .....</b>       | <b>32</b> |
| 6.5.1      | Integratie met Bestaande Systemen ..... | 32        |
| 6.5.2      | Unit Testing.....                       | 32        |

# 1 **ABSTRACT**

Dit document is geschreven op basis van mijn stage bij Pension Architects en dient ter afronding van mijn bacheloropleiding Toegepaste Informatica aan de Thomas More Hogeschool in Geel, België. Het onderwerp van deze stage was het verbeteren van de exportfunctionaliteit van grote datasets uit het interne beheersysteem van Pension Architects naar Excel.

Het project begon met een analyse van de bestaande exporttoepassing, die vaak crashte bij het verwerken van grote grids vanwege het hoge verbruik van rekenkracht. Om een beter inzicht te krijgen in deze beperkingen, heb ik de bestaande code onderzocht en interessante projecten en toepassingen geïdentificeerd die vergelijkbare functionaliteiten bieden.

Na deze initiële analyse, richtte de tweede fase van het project zich op het identificeren van de kernoorzaken van de crashes en het formuleren van oplossingen. Door vragen te stellen zoals "Waarom verbruikt de huidige toepassing zoveel rekenkracht bij grote datasets?" en "Welke specifieke behoeften hebben eindgebruikers bij het exporteren van data naar Excel?", kon ik de belangrijkste technische en gebruikersgerichte obstakels vaststellen.

In de laatste fase heb ik verschillende technologische verbeteringen en optimalisaties geïmplementeerd en getest. Deze verbeteringen zorgden ervoor dat de toepassing stabiel en efficiënter werd, wat resulteerde in een betrouwbare exportfunctionaliteit voor grote datasets.

De resultaten van dit project bieden een robuust framework en praktische aanbevelingen voor toekomstige verbeteringen van de exportfunctionaliteit bij Pensionarchitects. Deze studie draagt bij aan het verhogen van de klanttevredenheid door een efficiënte en betrouwbare manier te bieden om grote datasets naar Excel te exporteren.

## 2 **DANKWOORD**

Ik wil graag mijn dank uitspreken aan iedereen die heeft bijgedragen aan mijn succesvolle stageperiode bij Pensionarchitects. Het was een geweldige ervaring om deel uit te maken van een klein maar dynamisch team dat zich inzet voor het ontwikkelen van pensioenplannen voor andere bedrijven.

Allereerst wil ik de CEO, Ivan Eulaers, bedanken voor het bieden van de mogelijkheid om bij Pension Architects stage te lopen en te werken aan interne projecten.

Mijn stagebegeleider, Patrick Dielens, verdient bijzondere dank voor zijn voortdurende ondersteuning en begeleiding gedurende mijn stage. Zijn inzichten en adviezen waren van onschatbare waarde.

Ik wil ook mijn stage mentoren bedanken. Yves Anthoni heeft me uitstekend op weg geholpen totdat hij vanwege een operatie afwezig moest zijn. Na zijn vertrek hebben Nigel Riemis en Ben Maes zijn rol overgenomen. Hun begeleiding en hulp waren cruciaal voor de voortgang en het succes van mijn opdracht.

Een speciale dank gaat uit naar Tim Janssens voor het opzetten van de MacBook en zijn hulp met alle cloud-gerelateerde zaken. Zijn technische ondersteuning heeft ervoor gezorgd dat ik mijn werk efficiënt kon uitvoeren.

Ik ben ook dankbaar voor alle andere werknemers van Pension Architects voor het opnemen van mij als stagiair en voor hun bereidheid om mij te helpen waar nodig. Hun vriendelijkheid en steun hebben mijn tijd bij het bedrijf zeer aangenaam gemaakt.

Daarnaast wil ik de docenten van Thomas More Hogeschool bedanken voor hun begeleiding en onderwijs gedurende de afgelopen drie jaren. Hun inspanningen hebben me goed voorbereid op deze stage en mijn toekomstige carrière.

Mijn opdracht bestond uit het werken met een intern beheersysteem waarin data vaak wordt weergegeven in "grids", dit zijn grote tabellen. Mijn taak was om deze data in een Excel-bestand te zetten en te downloaden. Er was al een toepassing die dit deed, maar voor grotere grids veroorzaakte dit dat de hele applicatie crashte. Mijn opdracht was om dit probleem te onderzoeken en een betere oplossing te ontwikkelen.

Tot slot, een groot dankwoord aan iedereen die ik mogelijk vergeten ben te noemen. Uw hulp en steun worden zeer gewaardeerd!

Bedankt aan iedereen die deze stage tot een onvergetelijke en leerzame ervaring heeft gemaakt.

### 3 TERMINOLOGIE

**API (Application Programming Interface):** Een set regels en protocollen voor het bouwen en communiceren van softwaretoepassingen. Het definieert hoe verschillende softwarecomponenten met elkaar moeten communiceren en samenwerken.

**CI (Continuous Integration):** Een softwareontwikkelmethode waarbij teamleden hun werk frequent integreren, vaak meerdere keren per dag. Iedere integratie wordt automatisch geverifieerd door middel van een build (inclusief tests) om fouten zo snel mogelijk op te sporen.

**DB (Database):** Een gestructureerde verzameling gegevens die gemakkelijk kan worden geraadpleegd, beheerd en bijgewerkt. Het slaat informatie op die door de applicatie wordt gebruikt, zoals gebruikersgegevens en applicatie-instellingen.

**Dispatcher:** Een project dat wordt gebruikt door de beheerder om bestanden op te slaan. Het wordt gebruikt in combinatie met de juiste dispatchmessage om het bestand op de juiste plaats te plaatsen.

**Dispatchmessage:** Een tekstbestand dat wordt meegegeven met een Job waarin alle informatie staat die de Dispatcher nodig heeft om de Job uit te voeren, zoals het pad van het document, het documenttype en de datum.

**File Path:** De specifieke locatie van een bestand binnen het bestandssysteem of de cloudopslag. Het geeft aan waar een bestand is opgeslagen en hoe het kan worden benaderd.

**Grid:** Een tabel met gegevens vanuit de applicatie. In dit geval de data van de geëxporteerde Excel.

**Job:** Een proces dat wordt gestart om taken uit te voeren, zoals bijvoorbeeld een bestand te downloaden.

**Jobscheduler:** Een project dat Jobs en Taken uitvoert, of een wachtrij van Jobs die één voor één worden uitgevoerd.

**ORM (Object-Relational Mapping):** Een programmeertechniek voor het converteren van gegevens tussen incompatibele type-systemen in objectgeoriënteerde programmeertalen. Het creëert een "virtuele objectdatabase" die vanuit de programmeertaal kan worden gebruikt.

**SQS (Simple Queue Service):** Een volledig beheerde berichtwachtrijservice van AWS die je in staat stelt om microservices, gedistribueerde systemen en serverloze toepassingen los te koppelen en te schalen. Het helpt om Jobs asynchroon in de wachtrij te plaatsen en te verwerken.

**Task:** Een onderdeel van de Job dat een specifieke actie uitvoert. Meerdere taken vormen een Job, bijvoorbeeld een lege directory aanmaken, een bestand in die directory plaatsen, en de directory vervolgens verwijderen.

## **4 PROJECT**

### **4.1 Inleiding**

Tijdens mijn stage bij Pension Architects heb ik gewerkt aan een belangrijk project gericht op het verbeteren van de exportfunctionaliteit van grote datasets uit het interne beheersysteem van het bedrijf. In dit systeem worden gegevens vaak weergegeven in "grids" - grote tabellen met informatie. Hoewel dit systeem effectief is voor het beheren en analyseren van gegevens, geven veel klanten nog steeds de voorkeur aan het werken met Excel vanwege de bekendheid en de flexibiliteit die het biedt.

Het exporteren van grote grids naar Excel is echter niet zonder uitdagingen. De huidige exporttoepassing van Pension Architects kan grote datasets niet efficiënt verwerken, wat vaak resulteert in crashes en inefficiënte workflows. Dit vormt een probleem voor klanten die regelmatig met grote hoeveelheden data werken en deze willen exporteren voor verdere analyse en rapportage in Excel.

Mijn opdracht was om deze problemen te onderzoeken en een robuuste en efficiënte oplossing te ontwikkelen voor het exporteren van grote grids naar Excel. Dit betekende dat ik de beperkingen van de huidige toepassing ging analyseren, de oorzaken van de crashes identificeren en vervolgens een verbeterde oplossing heb ontworpen en geïmplementeerd.

Het project bestond uit drie hoofdfasen. De eerste fase omvatte het analyseren van de huidige beperkingen van de exporttoepassing door verder te bouwen op bestaande analyses. Het was al bekend dat de toepassing crashte bij grote grids vanwege het hoge verbruik aan rekenkracht. Om dit probleem beter te begrijpen, heb ik de bestaande code doorgelicht en interessante projecten en toepassingen genoteerd die ook een downloadfunctie hebben.

In de tweede fase lag de focus op het analyseren van mijn bevindingen uit de code-analyse om de belangrijkste oorzaken van de crashes te identificeren. Hierbij stelde ik vragen zoals "Waarom verbruikt de huidige toepassing zoveel rekenkracht bij grote datasets?" en "Welke specifieke behoeften hebben eindgebruikers bij het exporteren van data naar Excel?". Deze vragen hielpen bij het identificeren van de technische en gebruikersgerichte obstakels die de huidige exportfunctionaliteit belemmerden.

De laatste fase bestond uit het ontwikkelen en testen van oplossingen om de exportfunctionaliteit te verbeteren. Hierbij implementeerde ik technologische verbeteringen en optimalisaties die ervoor zorgden dat de toepassing stabiel en efficiënter werd. De effectiviteit van deze oplossingen werd vervolgens gemeten door middel van tests en feedback van gebruikers.

Hoewel dit project ambitieuze doelen had, is het belangrijk te erkennen dat de complexiteit van het exportprobleem een voortdurende inspanning vereist. Deze studie biedt daarom een algemeen framework en praktische aanbevelingen voor toekomstige verbeteringen en aanvullingen, zodat Pension Architects de exportfunctionaliteit kan afwerken en blijven optimaliseren.

## 4.2 Wie is Pension Architects?



*Figuur 4-1: Pension Architects*

Pension Architects werd opgericht door CEO Ivan Eulaers. Het begon allemaal bij Eulaers thuis, waar hij met veel Excel-sheets werkte en al snel inzag dat automatisering de efficiëntie aanzienlijk zou kunnen verbeteren. Dit inzicht leidde tot de oprichting van PensionArchitects, een bedrijf dat nu een team van ontwikkelaars en pensioenskundigen in dienst heeft die samen werken aan het optimaliseren van pensioenbeheer.

### 4.2.1 Innovatie en Groei

Sinds de oprichting heeft Pension Architects zich gevestigd als een innovatief bedrijf binnen de pensioenindustrie. Ze integreren technologie met diepgaande expertise in actuariële, fiscale, juridische en boekhoudkundige aspecten van pensioenen. Hun focus op automatisering en digitalisering maakt het mogelijk om administratieve processen te stroomlijnen, waardoor ze kosteneffectief en betrouwbaar zijn.

Ivan Eulaers en zijn team van ontwikkelaars hebben geavanceerde digitale tools en platforms ontwikkeld die grote hoeveelheden gegevens efficiënt beheren. Deze aanpak verhoogt niet alleen de nauwkeurigheid, maar ook de transparantie in pensioenbeheer, wat resulteert in betere dienstverlening voor hun klanten.

### 4.2.2 Toewijding aan Klantgerichtheid

Pension Architects staat bekend om hun sterke klantgerichtheid. Ze begrijpen dat pensioenen in de eerste plaats over mensen gaan en niet alleen over cijfers. Het bedrijf biedt gepersonaliseerde diensten aan, waarbij de vragen en zorgen van klanten snel en duidelijk worden beantwoord. Door voortdurend te investeren in technologie en innovatie, blijft Pension Architects een betrouwbare partner voor pensioenbeheer.

### 4.2.3 Toekomstvisie

Met het oog op de toekomst blijft Pension Architects streven naar verdere verbetering van hun diensten door geavanceerde technologieën te integreren en hun aanbod uit te breiden. Hun doel is om een leider te blijven in de pensioenadministratie, waarde te leveren aan hun klanten door innovatieve oplossingen en deskundige begeleiding.



## **4.3 Project Beschrijving**

### **4.3.1 Voorbereiding en Planning:**

Voorafgaand aan de ontwikkeling van het project heb ik een gedetailleerd plan opgesteld, inclusief een tijdschema en een overzicht van de benodigde taken. Dit zorgde voor een gestructureerde aanpak van het project.

### **4.3.2 Datamodeling en Review:**

Ik heb een datamodel ontworpen voor de database, waarin het bestandspad, jobs en taken werden opgeslagen. Dit model heb ik laten reviewen door mijn mentor om feedback te ontvangen en eventuele verbeteringen door te voeren.

### **4.3.3 Ontwikkeling van Prototipe Frontend:**

Ik heb een prototype van de frontend ontwikkeld voor de nieuwe webpagina binnen het systeem van Pensionarchitects. Dit prototype diende als visuele representatie van de project flow en gebruikersinterface, dit werd gebruikt om feedback te verzamelen voor verdere verbeteringen.

### **4.3.4 Implementatie van Backend Functionaliteiten:**

Verskillende backend functionaliteiten zijn ontwikkeld, waaronder het aanmaken van taken en jobs binnen het systeem, evenals het opzetten van de database structuur, API en het implementeren van de dispatchmessage-functionaliteit voor de dispatcher.

### **4.3.5 Aanpassing van Dispatcher en Flow van het Project:**

De dispatcher is aangepast om de ontvangen dispatchmessage te verwerken en de juiste opslag van bestanden te garanderen. Ook heb ik de workflow van het project gedefinieerd en geïmplementeerd, inclusief de interactie tussen de verschillende componenten.

### **4.3.6 Testen en Validatie:**

Tijdens de ontwikkeling heb ik tests uitgevoerd om de correcte werking van de functionaliteiten te verifiëren en eventuele bugs op te lossen. Dit zorgde voor een stabiel en betrouwbaar systeem. Ook zijn er altijd code reviews gedaan door mijn mentor die dan altijd goede en leerrijke feedback gaf.

## **5 GEBRUIKTE TECHNOLOGIEËN**

### **5.1 Technologieën**

#### **5.1.1 Java**

Java is een veelgebruikte, platformonafhankelijke programmeertaal die bekend staat om zijn stabiliteit en veelzijdigheid.

Reden: Java wordt al gebruikt in het bestaande systeem van Pensionarchitects, wat de integratie en compatibiliteit vergemakkelijkt.

#### **5.1.2 Apache Tapestry**

Apache Tapestry is een open-source Java webapplicatie framework dat component-gebaseerde architectuur gebruikt om dynamische, schaalbare webapplicaties te bouwen.

Reden: Tapestry wordt al gebruikt binnen Pension Architects voor webontwikkeling, waardoor consistentie en expertise in het team behouden blijven.

#### **5.1.3 Maven**

Maven is een build automation tool die voornamelijk wordt gebruikt voor Java-projecten. Het biedt een uniforme manier om projecten te beheren, afhankelijkheden te beheren en builds te automatiseren.

Reden: Maven stroomlijnt het bouwproces en wordt al toegepast in het bestaande systeem van Pensionarchitects.

#### **5.1.4 API**

API's (Application Programming Interfaces) worden gebruikt om verschillende softwarecomponenten met elkaar te laten communiceren.

Reden: Hoewel de basis-API al bestond, moest ze worden uitgebreid en aangepast om te voldoen aan de specifieke eisen en behoeften van de nieuwe toepassing. Deze aanpassingen waren nodig om de integratie van verschillende softwarecomponenten mogelijk te maken en een naadloze werking van het systeem te garanderen.

#### **5.1.5 JavaScript**

JavaScript is een programmeertaal die wordt gebruikt voor het ontwikkelen van interactieve webpagina's. Het geeft ontwikkelaars de mogelijkheid om de inhoud van webpagina's te manipuleren.

Reden: JavaScript is essentieel voor het dynamische gedrag van het front-end, met name in combinatie met Apache Tapestry.

#### **5.1.6 Liquibase**

Liquibase is een database schema wijziging management tool die database wijzigingen bijhoudt in een gestructureerde en gecontroleerde manier.

Reden: Liquibase wordt al gebruikt bij Pension Architects en helpt om database wijzigingen te beheren en te synchroniseren.

#### **5.1.7 Hibernate**

Hibernate is een object-relational mapping (ORM) tool voor Java die de interactie tussen de applicatie en de database vereenvoudigt.

Reden: Hibernate wordt al gebruikt en zorgt voor een efficiënte en flexibele datatoegang.

#### **5.1.8 MySQL**

MySQL is een open-source relationele databasemanagementsysteem (RDBMS) dat gegevens opslaat en beheert.

Reden: MySQL is de bestaande database binnen Pensionarchitects, wat zorgt voor continuïteit en compatibiliteit.

#### **5.1.9 Apache POI**

Apache POI is een Java bibliotheek voor het manipuleren van Microsoft Office documentformaten zoals Excel.

Reden: Apache POI is gebruikt om de Excel export functionaliteit te implementeren, zodat grote grids zonder problemen gedownload kunnen worden.

#### **5.1.10 Unit Test**

Unit testen zijn geautomatiseerde tests die individuele eenheden van broncode testen om ervoor te zorgen dat ze correct functioneren.

Reden: Unit testen worden gebruikt om de betrouwbaarheid en stabiliteit van de code te waarborgen.

#### **5.1.11 Tomcat**

Tomcat is een open-source implementatie van de Java Servlet en JavaServer Pages technologieën, die fungeert als een webserver en servlet container.

Reden: Tomcat wordt gebruikt om het lokale beheersysteem te runnen tijdens de ontwikkeling en het testen.

#### **5.1.12 Jenkins**

Jenkins is een open-source automatiseringsserver die gebruikt wordt voor continu integratie (CI). Het controleert de codekwaliteit door middel van geautomatiseerde tests en build processen.

Reden: Jenkins zorgt voor geautomatiseerde codecontrole en integratie, wat helpt om de kwaliteit en betrouwbaarheid van de software te waarborgen en het wordt bij Pensionarchitects ook al standaard gebruikt.

## **5.2 Programma's/ Platformen**

### **5.2.1 Figma**

Figma is een ontwerptool die wordt gebruikt voor het maken van gebruikersinterfaces, wireframes, prototypes en andere ontwerpelementen.

Reden: Figma werd gebruikt om sitemaps, wireframes en prototypes van de frontend te maken en om feedback van de mentor te verwerken.

### **5.2.2 StarUML**

StarUML is een software modeling tool die gebruik maakt van UML (Unified Modeling Language) om software-architectuur te visualiseren.

Reden: StarUML werd gebruikt voor het ontwerpen van de database structuur en het modelleren van use cases, vanwege het uitgebreide gebruik tijdens schoolprojecten.

### **5.2.3 Git**

Git is een versiebeheersysteem dat ontwikkeld is om projecten te beheren en samenwerking te ondersteunen. Het biedt de mogelijkheid om code te delen en versiegeschiedenis bij te houden.

Reden: Git wordt gebruikt om de versiecontrole en samenwerking binnen het ontwikkelteam te verbeteren, en wordt lokaal op kantoor gehost via AWS.

### **5.2.4 Draw.io**

Draw.io is een tool voor het maken van flowcharts en diagrammen, waardoor een visuele representatie van projectinformatie mogelijk is.

Reden: Draw.io werd gebruikt om flowcharts te tekenen en zo een visuele representatie van de projectuitleg eenvoudiger te maken.

## **5.3 Methodologieën**

### **5.3.1 4.3.1 SCRUM**

SCRUM is een agile methodologie die werkt op basis van sprints, typisch van twee tot vier weken. Het omvat ook retrospectives waarin het team terugblijkt op de vorige sprint en elkaar feedback bezorgt over de positieve en negatieve punten van die sprint.

Reden: Bij Pension Architects wordt volgens de SCRUM-methodologie gewerkt en ik ben aangesloten bij deze werkwijze.

### **5.3.2 Code Review**

Code Review is een praktijk waarbij de code wordt gecontroleerd en beoordeeld door andere teamleden om de kwaliteit van de code te verbeteren en eventuele fouten op te sporen.

Reden: Na het voltooien van een taak werd mijn code gereviewd door mijn mentor, die feedback gaf en eventuele verbeteringen voorstelde. Dit proces droeg bij aan het verbeteren van de codekwaliteit en het delen van kennis binnen het team.

## **6 REALISATIE**

Tijdens mijn stage bij Pension Architects heb ik aan verschillende onderdelen van het project gewerkt, gericht op het verbeteren van de exportfunctionaliteit voor grote data grids en de bijbehorende backend-processen. Hieronder volgt een overzicht van mijn realisaties:

### **6.1 Analyse van Bestaande Projecten en Software**

Beschrijving: Als eerste stap in het project heb ik de bestaande analyse doorgenomen, waarin werd opgemerkt dat het systeem crasht bij het exporteren van grote data grids door het hoge rekenkrachtverbruik.

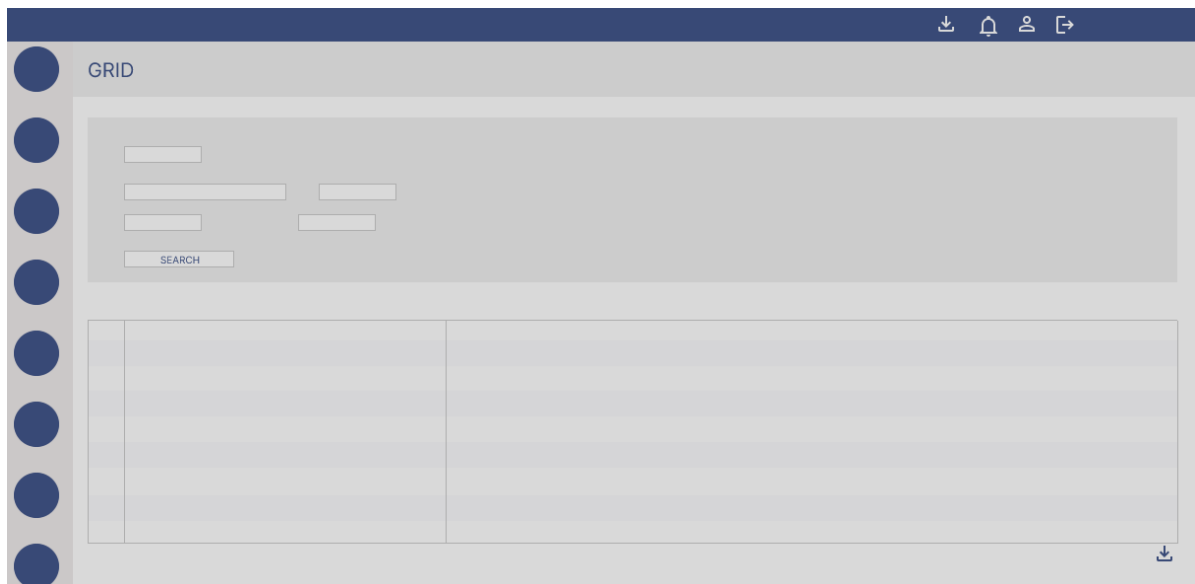
Acties: Ik heb de bestaande codebase uitgebreid onderzocht. Daarnaast heb ik interessante bestaande projecten en toepassingen met downloadfunctionaliteiten geïdentificeerd en geanalyseerd om beste mogelijke oplossingen te ontdekken. Dit legde een basis voor verdere optimalisaties en verbeteringen.

### **6.2 Prototype en Datamodeling**

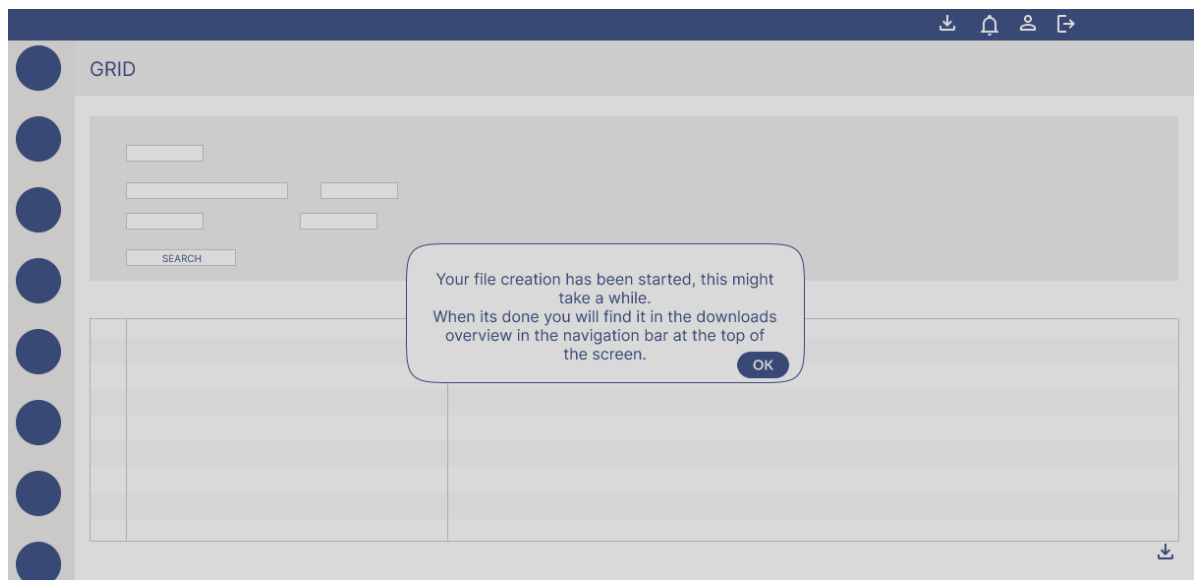
#### **6.2.1 Prototype Ontwikkeling**

Beschrijving: Als eerste stap heb ik een prototype van de frontend ontwikkeld, waarbij ik de gebruikersinterface en de flow van het project visualiseerde.

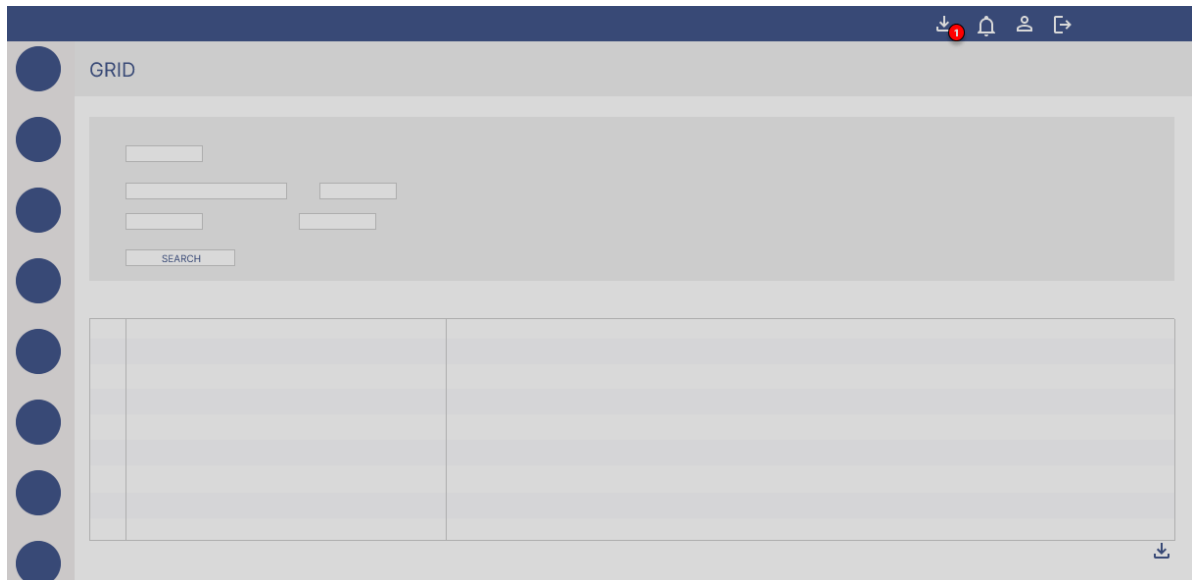
Acties: Ik gebruikte Figma om een representatie van het uiteindelijke project flow te maken. Na het voltooien van de eerste versie, presenteerde ik het prototype aan mijn mentor en een paar collega's voor feedback. Op basis van de ontvangen feedback werden verschillende iteraties uitgevoerd om de interface te verfijnen en te optimaliseren. Het eindresultaat was een duidelijk en gebruiksvriendelijk ontwerp dat klaar was voor implementatie.



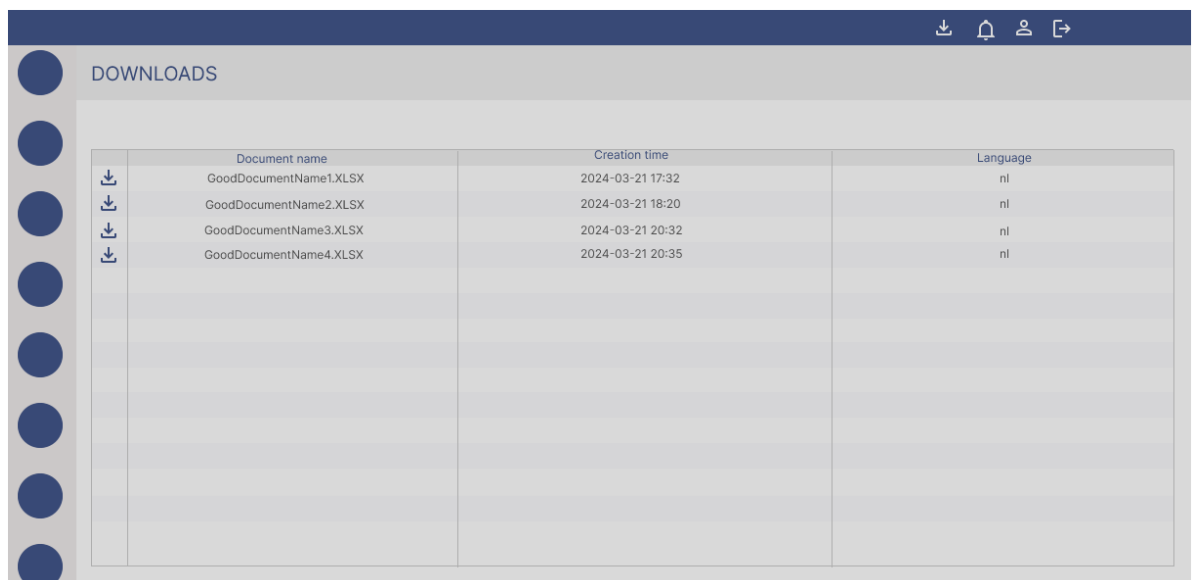
*Figuur 6-1: prototype grid om te exporteren*



*Figuur 6-2: prototype confirmatie pop-up*



*Figuur 6-3: prototype notificatie file is klaar*



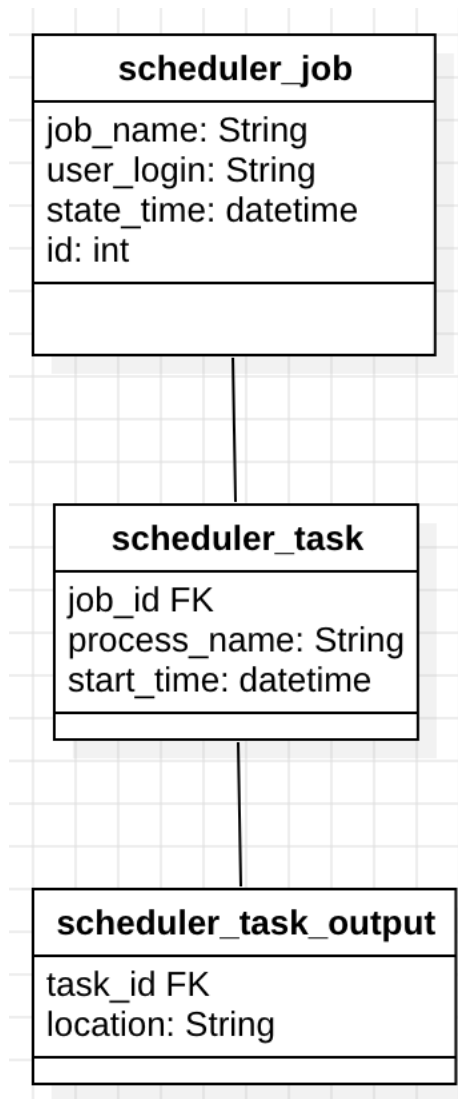
*Figuur 6-4: prototype pagina met alle beschikbare files*

### 6.2.2 Datamodeling

**Beschrijving:** Vervolgens heb ik het datamodel voor de benodigde database gemaakt, waarin de structuren voor het opslaan van file paths, jobs, en tasks werden gedefinieerd.

**Acties:** Ik ontwierp het datamodel in StarUML en zorgde voor regelmatige reviews door mijn mentor. Na enkele aanpassingen en goedkeuring werd de definitieve database structuur geïmplementeerd. Dit model vormde de basis voor het beheren van de data gerelateerd aan de exportfunctionaliteit en de bijbehorende processen.



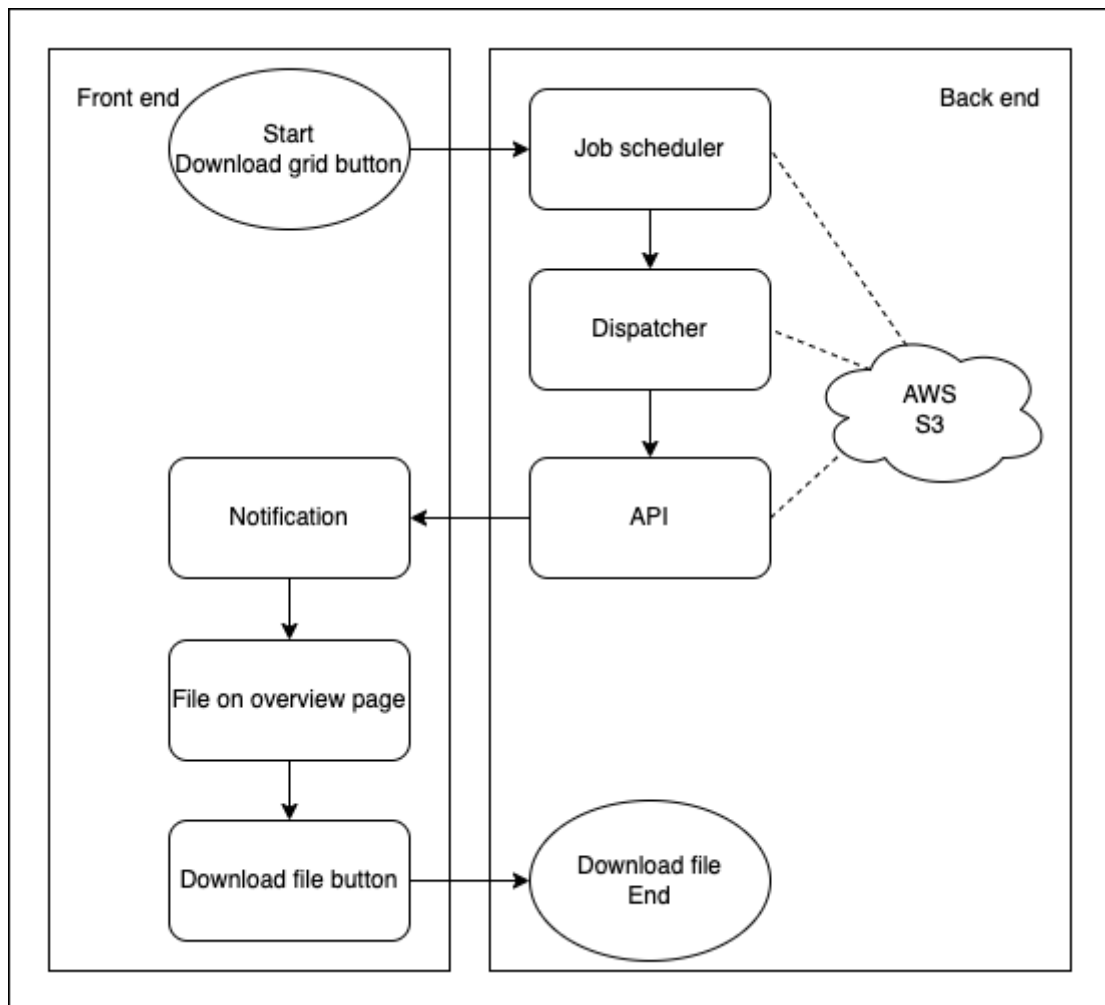


*Figuur 6-5: datamodel*

## 6.3 Backend en Front-end Ontwikkeling

### 6.3.1 Volledige Project Flow

De volledige projectflow begint met het idee van een gebruiker om een grid te exporteren. Dit proces omvat verschillende stappen en interacties tussen de verschillende componenten van het systeem. Hieronder wordt de flow opgesplitst in afzonderlijke delen, waarbij elk deel gedetailleerd wordt beschreven om een dieper inzicht te bieden in de technische aspecten van het proces.

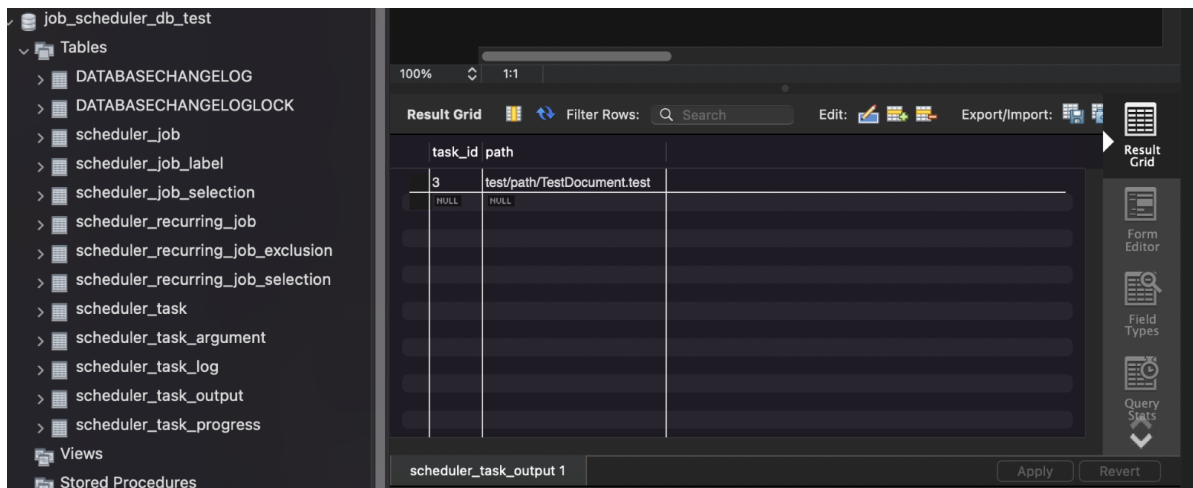


*Figuur 6-6: flowchart van de volledige project flow*

### 6.3.2 Database Structuur

**Beschrijving:** Opzetten van een database structuur voor het opslaan van file paths, jobs, en tasks.

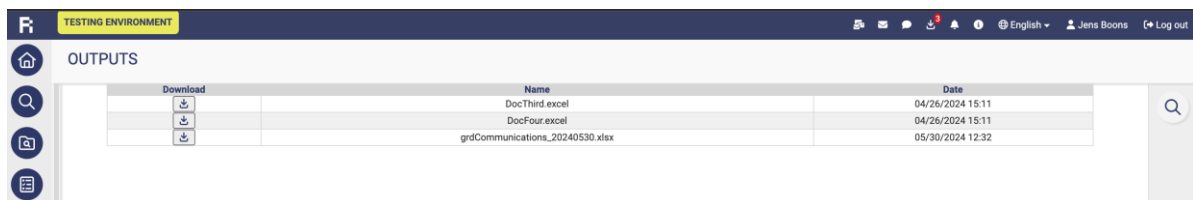
**Acties:** Ik implementeerde het datamodel in MySQL en zorgde voor de integratie met Hibernate voor ORM-functionaliteit. Dit omvatte het aanmaken van tabellen en het definiëren van de relaties tussen verschillende entiteiten, wat een efficiënte en schaalbare manier bood om de benodigde data te beheren.



*Figuur 6-7: database structuur*

### 6.3.3 Nieuwe Webpagina voor File Management

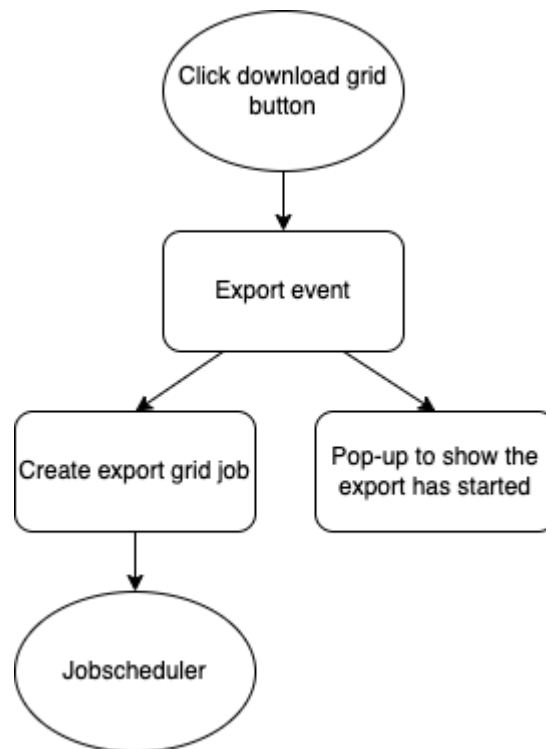
Beschrijving: Een nieuwe webpagina werd ontwikkeld binnen het bestaande systeem om de bestanden te beheren en weer te geven.



*Figuur 6-8: file-management pagina*

Acties: Ik integreerde deze pagina in de bestaande applicatie met behulp van Apache Tapestry. De pagina stelde gebruikers in staat eenvoudig door bestanden te navigeren, deze kunnen dan gedownload worden door op de knop bij de file te klikken.

### 6.3.4 Export Grid knop



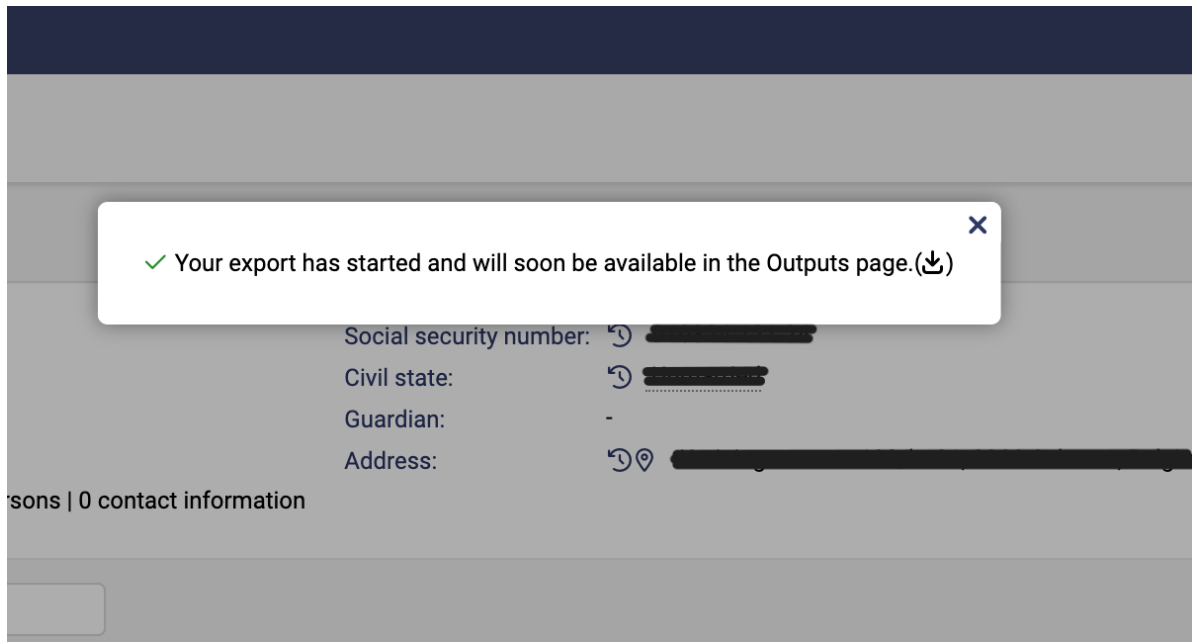
Figuur 6-9: flowchart van de knop waarmee je de export start

| COMMUNICATIONS |               |           |               |        |                   |   |  |  |  | SIGEDIS DECLARATIONS |  |  |  |
|----------------|---------------|-----------|---------------|--------|-------------------|---|--|--|--|----------------------|--|--|--|
| Date           | From          | To        | Administrator | Medium | Category          | Document type   |  |  |  |                      |  |  |  |
| 09/20/2023     | Administrator | Affiliate | System        | E-box  | Benefit statement | Annual benefit statement  |  |  |  |                      |  |  |  |
| 08/30/2022     | Administrator | Affiliate | System        | E-box  | Benefit statement | Annual benefit statement  |  |  |  |                      |  |  |  |
| 09/21/2021     | Administrator | Affiliate | System        | E-box  | Benefit statement | Annual benefit statement  |  |  |  |                      |  |  |  |
| 12/18/2020     | Administrator | Affiliate | System        | E-box  | Exit light        | Formulier aan de actieve aangesloten om te melden dat de aansluiting aan het plan gestopt is. |  |  |  |                      |  |  |  |
| 09/16/2020     | Administrator | Affiliate | System        | E-box  | Benefit statement | Annual benefit statement  |  |  |  |                      |  |  |  |

Figuur 6-10: voorbeeld van een grid dat kan geëxploreerd worden



Figuur 6-11: uitvergrote foto van de grid utility-bar met de nieuwe export knop



*Figuur 6-12: pop-up zodat de gebruiker weet dat de export is gestart*

**Beschrijving:** Implementatie van een knop voor het exporteren van een grid in de webapplicatie. De eerste stap in het proces is vrij eenvoudig: een gebruiker klikt op de knop "Export". Deze knop is wat hen in staat stelt om het exporteren van de grid gegevens uit het systeem te halen en elders te gebruiken. Het klikken op deze knop triggert een signaal dat het systeem een exportjob laat starten.

**Acties:** De gebruiker begint het exportproces door op de knop "Export" te klikken. Deze knop start van het exporteren mogelijk maakt. Door op de knop te klikken, wordt er een signaal gegenereerd dat het systeem vertelt om een exportjob te starten. Ik heb een HTML-element gemaakt dat fungeert als de exportknop. Met behulp van jQuery heb ik een click event listener toegevoegd aan dit element, zodat het systeem weet wanneer de knop is ingedrukt. Wanneer de knop wordt ingedrukt, wordt er een asynchrone exportactie gestart. Tijdens het exportproces wordt de knop tijdelijk vervangen door een spinner om de gebruiker te laten weten dat er een actie plaatsvindt en ook wordt er een pop-up getoond die uitleg geeft waar u file zal komen staan wanneer deze aangemaakt is. Deze wordt ook vertaald naar de taal van de gebruiker

**Beschrijving:** Aanmaken van een taak in de Jobscheduler voor het exporteren van een grid.

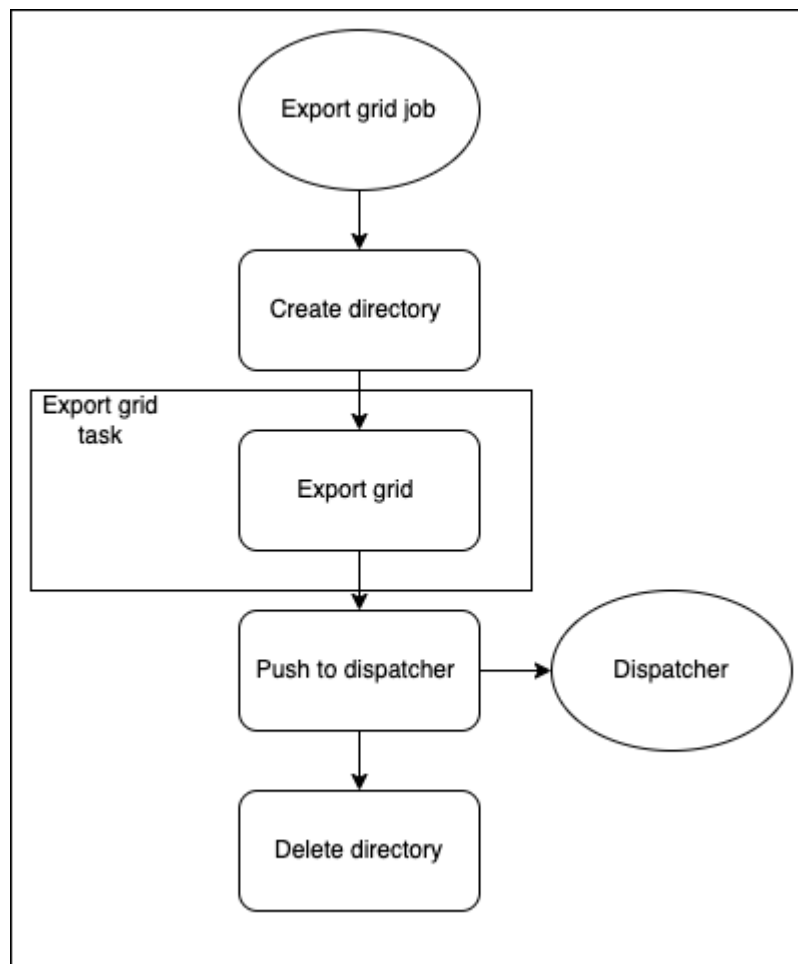
**Acties:** Zodra de exportactie is gestart, wordt er een taak gecreëerd in de Jobscheduler om het exportproces te verwerken. Dit wordt bereikt door een methode in de backend van de applicatie aan te roepen, genaamd exportGridAsync. Binnen deze methode wordt de taak voor de export ingepland en toegevoegd aan de wachtrij van de Jobscheduler voor verdere verwerking.

### **6.3.5 Tasks en Jobs Implementatie**

**Beschrijving:** Het aanmaken van tasks en jobs, die door de jobscheduler uitgevoerd kunnen worden.

Acties: Ik ontwikkelde de nodige Java-klassen en methoden voor het aanmaken, beheren en uitvoeren van tasks en jobs. Een job werd opgebouwd uit een reeks tasks zoals het aanmaken van een lege directory, het opslaan van een file, en het verwijderen van tijdelijke bestanden.

#### 6.3.5.1 Jobscheduler



*Figuur 6-13: flowchart van de jobscheduler*

Beschrijving: Werking van de Jobscheduler voor het verwerken van de exporttaak met verschillende taken.

Acties: Nadat de export job is aangemaakt is het aan de Jobscheduler, deze verwerkt verschillende taken die in de job zitten en worden meegegeven met de exportGridAsync. Deze taken omvatten het maken van een lege directory, het uitvoeren van de grid task waarin de file echt wordt in aangemaakt, het verzenden van de bestanden naar de dispatcher en uiteindelijk het opruimen van de directory nadat de bestanden zijn verzonden.

Beschrijving van de taken:

**Maken van een lege directory:** Voordat het exportproces begint, wordt er een lege directory aangemaakt op de server van de Jobscheduler. Dit is een tijdelijke

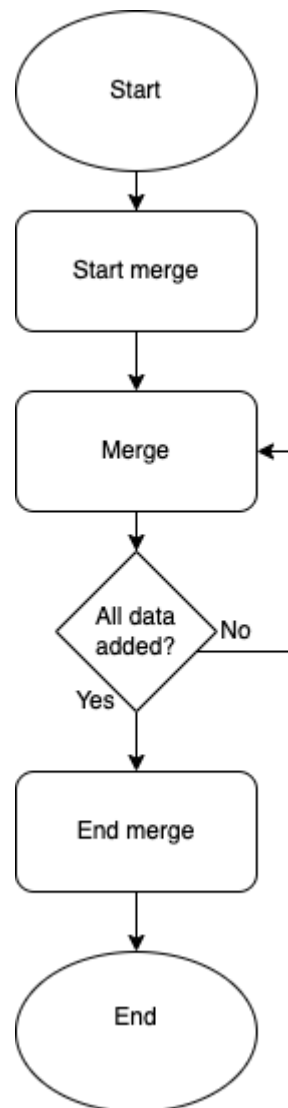
opslaglocatie waar de bestanden kunnen worden opgeslagen voordat ze worden verzonden naar de dispatcher.

**Uitvoeren van de gridtask:** De gridtaak is verantwoordelijk voor het genereren van een dispatchmessage en het bestand dat moet worden geëxporteerd. Deze taak wordt uitgevoerd nadat de lege directory is gemaakt. Het genereert het benodigde bestanden op basis van de geselecteerde grid en slaat het op in de tijdelijke directory.

**Push to dispatcher:** Nadat het bestand is gegenereerd wordt deze door de jobscheduer naar een gedeelde bucket op AWS S3 gezet waar hij beschikbaar wordt gesteld voor verdere verwerking door de dispatcher 6.4.6. Dit wordt gedaan met behulp van een SQS-que en de dispatchmessage waar ik in 6.4.5 meer uitleg over geef.



**Opruimen van de directory:** Zodra de bestanden succesvol zijn verzonden naar de dispatcher, is de tijdelijke directory niet langer nodig. Daarom wordt deze directory verwijderd om schijfruimte vrij te maken en resources te optimaliseren.

### 6.3.5.2 Gridtask



*Figuur 6-14: flowchart van de grid task*



| PLANNED JOBS                          |   |                 |         |                  |           |   |
|---------------------------------------|---|-----------------|---------|------------------|-----------|---|
| Planned jobs                          |   |                 |         |                  |           |   |
| Scheduled time                        | Status time   | Selection       | User    | Job name         | State     |   |
| 05/30/2024 12:20                      | 05/30/2024 12:20  |                 | jens    | export grids     | Scheduled |   |
| Showing 1-1 of 1 rows << < 1 > >> new |   |                 |         |                  |           |   |
| Tasks                                 |   |                 |         |                  |           |   |
| Task name                             | Arguments   | Start time      | Status  | Status time      |           |   |
| 0 createEmptyDirectory                | -dir /Users/thomax/output/jobscheduler-local/export-grids/shared/   | not yet started | Created | 05/30/2024 12:20 |           |   |
| 1 export-grids                        | -gridReference gridCommunications<br>-output /Users/thomax/output/jobscheduler-local/export-grids/shared/   | not yet started | Created | 05/30/2024 12:20 |           |   |
| 2 pushToDispatcher                    | -file /Users/thomax/output/jobscheduler-local/export-grids/shared/<br>-id ed68788d-3599-44f8-8af3-8515f1e06464<br>-type EXPORT_GRIDS -scope UNKNOWN | not yet started | Created | 05/30/2024 12:20 |           |   |
| 3 dispatching                         | -id ed68788d-3599-44f8-8af3-8515f1e06464<br>-type EXPORT_GRIDS -scope UNKNOWN   | not yet started | Created | 05/30/2024 12:20 |           |   |
| 4 deleteFileOrDirectory               | -cldir /Users/thomax/output/jobscheduler-local/export-grids/shared/   | not yet started | Created | 05/30/2024 12:20 |           |   |
| Showing 1-5 of 5 rows << < 1 > >> new |   |                 |         |                  |           |   |

Figuur 6-15: jobscheduler overview van de export-grid job

De Grid Task is verantwoordelijk voor het genereren van een Excel-bestand met gridgegevens en het aanmaken van een dispatchmessage.

Acties van de Grid Task:

**Start merge functie:** De Grid Task begint met het opzetten van een nieuw Excel-workbook en een waarop de gridgegevens zullen worden geschreven.

**Merge functie:** Vervolgens worden de gridgegevens samengesteld en ingevoegd in het werkblad. Dit omvat het formatteren van de gegevens en het plaatsen ervan in de juiste cellen van het werkblad.

**End merge functie:** Na het invullen van het werkblad met de gridgegevens wordt het Excel-bestand geschreven naar een tijdelijke locatie op de server van de Jobscheduler. Tegelijkertijd wordt een dispatchmessage aangemaakt met relevante metadata, zoals het pad naar het Excel-bestand en de gebruiker die de export heeft geïnitieerd.

```
{
  "id": "5847410f-0b41-46d9-9efa-aa2184d52abc",
  "dispatchDate": "2024-05-29",
  "documentPath": "/Users/kevinx/output/jobscheduler-local/export-grids/shared/grdJobs_20240529.xlsx",
  "documentType": "EXPORT_GRID",
  "scope": "USER",
  "scopeValues": [
    "03081123370"
  ],
  "discriminator": "DM",
  "taskId": 467305
}
```

*Figuur 6-16: voorbeeld van een dispatch message*

Het resulterende Excel-bestand en de dispatchmessage worden vervolgens opgeslagen in de tijdelijke directory en zijn klaar voor verdere verwerking door de dispatcher.

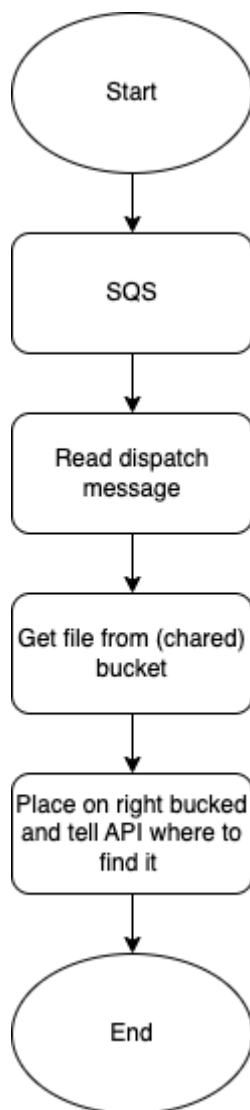
Het was de bedoeling om gedetailleerde gridgegevens te extraheren en in het exportbestand te plaatsen, maar dit proces bleek ingewikkelder dan daarvoor gedacht vanwege verschillende dependenties die niet mogen voor beveiligingsvereisten. Om vertraging van het project te voorkomen, is een tijdelijke oplossing geïmplementeerd waarbij telkens hetzelfde dummy-bestand met vooraf ingestelde gegevens wordt gegenereerd. Dit maakte het mogelijk om andere aspecten van het project te ontwikkelen zonder te wachten op de volledige implementatie van de gegevensextractie.

### **6.3.6 Dispatcher en Dispatch Messages**

**Beschrijving:** Aanpassingen aan de dispatcher om de nieuwe functionaliteit te ondersteunen.

**Acties:** Ik implementeerde dispatch message voor de grid export, waarbij alle informatie die de dispatcher nodig heeft om een job uit te voeren, zoals het pad van het document, documenttype en datum, werd meegegeven. Vervolgens werden de nodige wijzigingen in de dispatcher aangebracht om deze nieuwe berichten correct te verwerken. Dit zorgde voor een naadloze integratie van de nieuwe functies binnen het bestaande systeem.

#### 6.3.6.1 Dispatcher

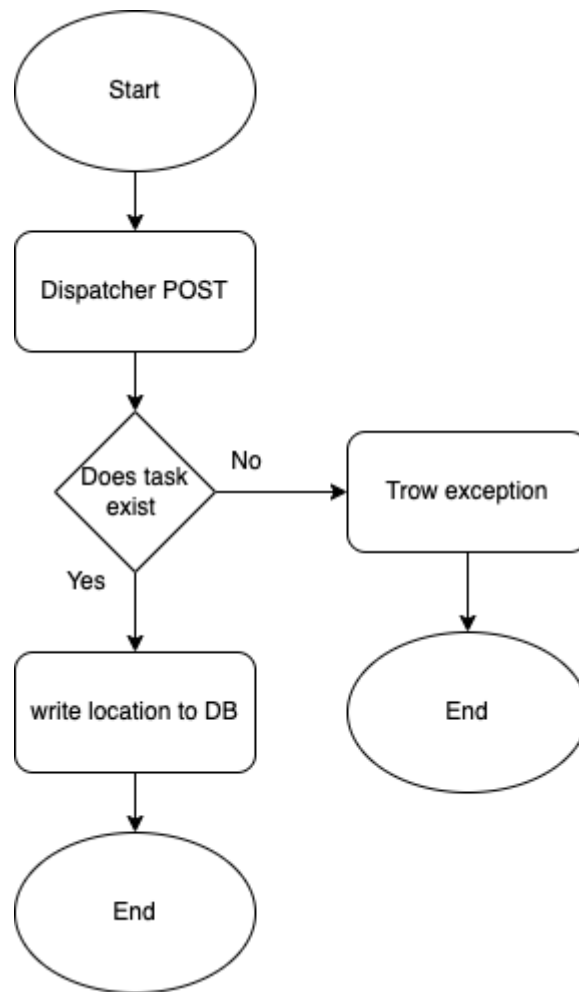


*Figuur 6-17: flowchart van de dispatcher*

Beschrijving: Functie van de Dispatcher voor het verwerken van dispatchberichten en het verplaatsen van bestanden naar een andere bucket via een API.

Acties: Zodra de dispatchmessage is gegenereerd door de Jobscheduler, wordt deze naar een SQS-queue gestuurd. De Dispatcher leest deze berichten uit de wachtrij en verkrijgt zo de locatie van het bestand via het meegeleverde pad. Vervolgens verplaatst het de bestanden naar een andere bucket, waar ze permanent worden opgeslagen. Dit gebeurt door een POST-request te sturen naar de Pension Architects API, waardoor de bestanden veilig kunnen worden opgeslagen op de nieuwe locatie.

### 6.3.7 API

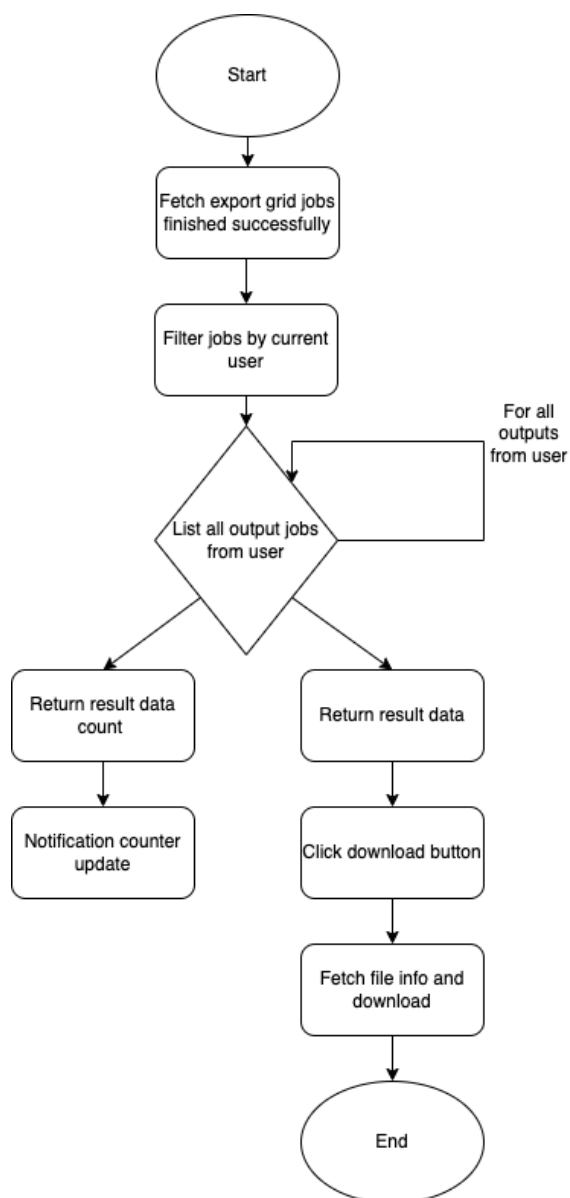


*Figuur 6-18: flowchart van de API*

Beschrijving: Werking van de API voor het verwerken van export grid taken en het opslaan van bestanden in de database.

Acties: Wanneer de API een POST-request ontvangt, bevat dit het pad en de ID van de grid taak. De API gebruikt deze informatie om te controleren of het bestand bestaat. Als het bestand niet gevonden wordt, wordt er een uitzondering gegooid. Indien het bestand wel gevonden wordt, slaat de API het bestand op in de juiste locatie binnen de database. Dit gebeurt in de task-outputs-tabel, die ik heb aangemaakt tmet hybernate en liquibase. Hierdoor kan de front-end eenvoudig toegang krijgen tot de geëxporteerde bestanden en deze aan de gebruikers presenteren.

## 6.4 Ophalen van output files



Figuur 6-19: flowchart van ophalen van files en notificaties

### 6.4.1 Ophalen



Figuur 6-20: view van de geëxporteerde file met notificatie in het systeem

Beschrijving: Het ophalen en weergeven van geëxporteerde gridbestanden op een nieuwe pagina in de webapplicatie.

**Acties:** Wanneer een gebruiker de nieuwe pagina bezoekt waarop de geëxporteerde gridbestanden worden weergegeven, zorgt de applicatie ervoor dat de bestanden correct worden opgehaald en in een overzichtelijke lijst worden weergegeven.

Ophalen van de gegevens:

Bij het renderen van de pagina wordt een lijst met geëxporteerde gridbestanden opgehaald van de server. Deze gegevens bevatten details zoals de naam van het document en de datum waarop het is aangemaakt. Deze informatie wordt opgehaald via een service die met de database communiceert.

Weergave van de bestanden in een tabel:

De opgehaalde gegevens worden weergegeven in een tabel op de webpagina. Elke rij in de tabel toont de naam van het document en de aanmaakdatum, samen met een downloadknop voor elk bestand.

Bestanden downloaden:

Wanneer een gebruiker op de downloadknop klikt, wordt er een verzoek gestuurd naar de server om het geselecteerde bestand te downloaden.

De server zoekt het bestand op in de S3-bucket aan de hand van het pad dat bij het verzoek is meegegeven.

Als het bestand wordt gevonden, wordt een downloadlink gegenereerd en aan de gebruiker gepresenteerd, zodat ze het bestand kunnen downloaden en lokaal opslaan.

Dit proces zorgt ervoor dat gebruikers eenvoudig toegang hebben tot de geëxporteerde gridbestanden, deze kunnen bekijken en downloaden voor verder gebruik.

#### 6.4.2 Notificaties



*Figuur 6-21: notificatie-bar met een notificatie voor geëxporteerde files*

**Beschrijving:** Implementatie van notificaties voor nieuwe geëxporteerde gridbestanden

**Acties:** Het systeem toont notificaties aan gebruikers wanneer er nieuwe geëxporteerde gridbestanden beschikbaar zijn. Dit gebeurt via een notificatie-icoon in de navigatiebalk, dat het aantal nieuwe bestanden weergeeft.

**Vorbereiding van de notificaties:**

De OutputView-component controleert op nieuwe geëxporteerde gridbestanden met behulp van de GridsExportFrontService.

Het aantal nieuwe bestanden wordt opgeslagen en bijgewerkt wanneer er nieuwe gegevens beschikbaar zijn.

**Tonen van de notificaties:**

In de navigatiebalk wordt een notificatie-icoon weergegeven met het aantal nieuwe bestanden.

Als er nieuwe bestanden zijn, verschijnt er een cijfer naast het icoon dat het aantal nieuwe bestanden aangeeft.

Navigeren naar de bestanden:

Wanneer een gebruiker op het notificatie-icoon klikt, wordt hij naar de outputs-pagina gebracht.

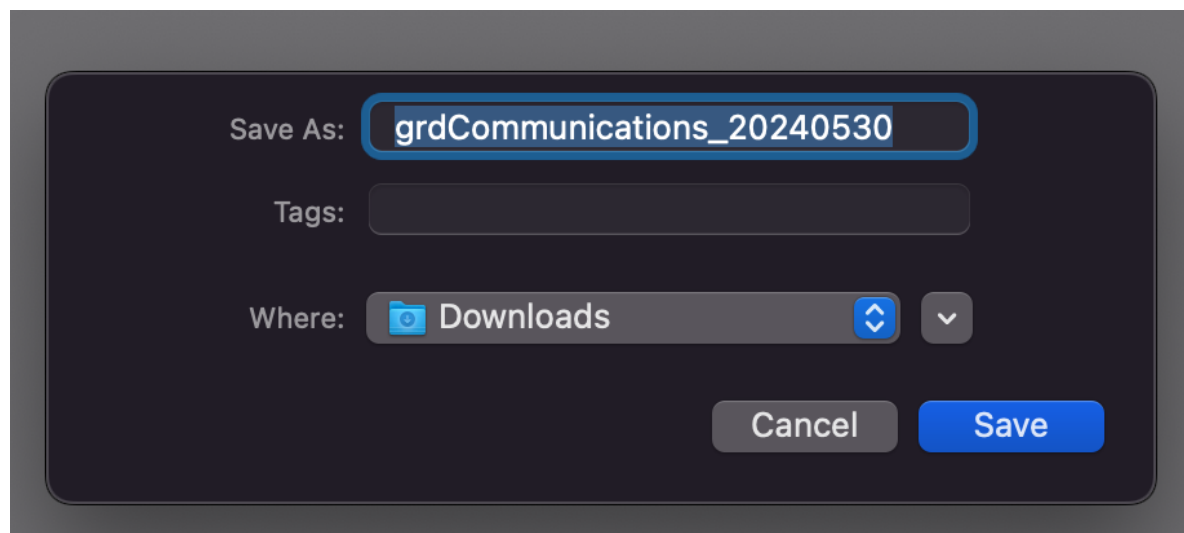
Op deze pagina kan de gebruiker de nieuwe geëxporteerde gridbestanden bekijken en downloaden.

Deze aanpak zorgt ervoor dat gebruikers snel op de hoogte zijn van nieuwe geëxporteerde gridbestanden en eenvoudig toegang hebben tot deze bestanden.

### 6.4.3 Download Functionaliteit



Figuur 6-22: file met download knop



Figuur 6-23: conformatie de file wordt gedownload

Beschrijving: Het uiteindelijk downloaden van de geëxporteerde grid om lokaal te bewaren of gebruiken.

Acties: Gebruikers kunnen de geëxporteerde gridbestanden downloaden door op de downloadknop naast elk bestand in de tabel te klikken.

Klikken op de downloadknop: Wanneer een gebruiker op de downloadknop naast een bestand in de tabel klikt, wordt een downloadverzoek naar de server gestuurd. Dit verzoek bevat het pad van het geselecteerde bestand.

Bestand ophalen van de server: De server ontvangt het downloadverzoek en zoekt het bestand op in de S3-bucket met behulp van het opgegeven pad. Als het bestand bestaat, genereert de server een downloadlink en stuurt deze terug naar de gebruiker.

Downloaden van het bestand: De gebruiker ontvangt de downloadlink en kan het bestand downloaden en lokaal opslaan.

Deze downloadfunctionaliteit zorgt ervoor dat gebruikers eenvoudig toegang hebben tot de geëxporteerde gridbestanden en deze kunnen downloaden voor verdere verwerking of opslag.

## **6.5 Integratie en Testen**

### **6.5.1 Integratie met Bestaande Systemen**

Beschrijving: Integratie van de nieuwe functies met de bestaande systemen en processen binnen Pensionarchitects.

Acties: Ik zorgde voor de integratie van de nieuwe database, jobscheduler, dispatcher, en frontend elementen binnen het bestaande systeem. Dit omvatte samenwerking met teamleden om ervoor te zorgen dat alle onderdelen naadloos samenwerkten en uitgebreide tests om te bevestigen dat de integratie succesvol was. Dit verzekerde dat de nieuwe functies betrouwbaar en effectief waren in de productieomgeving.

### **6.5.2 Unit Testing**

Beschrijving: Opzetten van unit tests doorheen de development van het project.

Acties: Ik ontwikkelde uitgebreide unit tests om de nieuwe functionaliteiten grondig te testen. Dit zorgde voor een continue controle van de codekwaliteit, wat bijdroeg aan een stabielere en betrouwbaardere codebase.