



# Draaiboek - stage

**Elevate-IT: Bauwens Logistics**

**Thiemo Cumps**



## 1 Inhoudsopgave

1	Inhoudsopgave .....	2
2	Algemene afspraken.....	4
2.1	Documentatie .....	4
2.2	Schriftelijke neerslag.....	4
2.3	Logboek .....	4
2.4	Presentatie.....	4
2.5	Evaluatie en opvolging school .....	4
3	Algemene Informatie .....	5
3.1	Gegevens student .....	5
3.2	Opdrachtgever.....	5
3.3	Gerelateerde documenten.....	5
4	Stagebedrijf.....	6
5	Stageopdracht.....	7
5.1	Aanleiding van de opdracht .....	7
5.2	Eisen en wensen .....	8
5.2.1	Logging dashboard.....	8
5.2.2	Conveyor.....	8
5.2.3	Automatische rekken.....	8
5.3	Fasering.....	9
5.3.1	Analyse .....	9
5.3.2	Design .....	10
5.3.3	Ontwikkeling.....	13
6	Planning .....	14
6.1	Overzicht.....	14
7	Plan van aanpak .....	15
7.1	Sprint 1.....	15
7.1.1	Voorgesteld taken .....	15
7.2	Sprint 2.....	16
7.2.1	Voorgesteld werk .....	16
7.3	Sprint 3.....	16
7.3.1	Voorgesteld werk .....	16





NETWERK VAN MENSEN

8	Realisatie.....	18
8.1	Global variable.....	18
8.2	Subflow.....	18
8.3	UIbuilder.....	18
8.4	Conveyor.....	18
8.4.1	Scanner → barcode.....	18
8.4.2	Moxa Height API Call.....	19
8.4.3	Test data.....	20
8.4.4	STAP 1: Dashboard Node Red.....	20
8.4.5	STAP 2: Dashboard UIbuilder.....	21
8.5	Automatische rekken.....	22
8.5.1	Aisle X subflow.....	23
8.5.2	Log+send to FTP.....	23
8.5.3	Dashboard Node Red.....	24
8.5.4	Dashboard UIbuilder.....	25
8.6	Logging.....	26
8.6.1	Set filecontent.....	26
8.6.2	Filename generator.....	27
8.6.3	Write to logfile.....	27
8.6.4	Display to the live logging feed.....	27
8.6.5	Write to log file.....	28
8.7	Reflectie.....	29





## 2 Algemene afspraken

### 2.1 Documentatie

Alle documentatie die stage gerelateerd is dient opgeslagen te worden onder een persoonlijke folder van de stagestudent die toegankelijk is voor de stagiair en de stagebegeleider.

### 2.2 Schriftelijke neerslag

Het schriftelijke verloop van de stage dient in dit document te zijn opgenomen. Alle gemaakte keuzes en onderzochte pistes moeten in document zijn te vinden. De student verbindt zich ertoe dit document op een wekelijkse basis aan te vullen en door te sturen aan de stagebegeleider.

### 2.3 Logboek

Er moet door de stagestudent een logboek worden bijgehouden dat wekelijks gedeeld wordt met de hogeschool, de opdrachtgever en de stagebegeleider. Een template kan gevonden worden onder de stagemap.

### 2.4 Presentatie

Elke 2 weken van de stage wordt er een voortgangspresentatie gemaakt voor het stagebedrijf. Een voorbeeld template kan gevonden worden in de map van de stage hogeschool.

### 2.5 Evaluatie en opvolging school

Het is gedurende de stage de verantwoordelijkheid van de stagestudent om de afspraken tussen de school en het stagebedrijf te voorzien en op te volgen. Indien er evaluatiemomenten moeten gedaan worden dan moet de student hiervoor de stagebegeleider triggeren.





### 3 Algemene Informatie

#### 3.1 Gegevens student

Naam	Thiemo Cumps
Adres	Tiendenstraat 10
	2235 Hulshout
	Antwerpen, België
Telefoon	

#### 3.2 Opdrachtgever

Klant	
Contactpersoon Management	Maarten van Gysegem
Email	<a href="mailto:Maarten.VanGysegem@elevate-it.be">Maarten.VanGysegem@elevate-it.be</a>
Telefoon	/
Contactpersoon IT	Frederik Fostier
Email	<a href="mailto:Frederik.Fostier@elevate-it.be">Frederik.Fostier@elevate-it.be</a>
Telefoon	/
3-it bvba (3-it)	
Email	<a href="mailto:Toon.peters@3-it.be">Toon.peters@3-it.be</a>
Telefoon	/
Email algemeen	<a href="mailto:support@3-it.be">support@3-it.be</a>
Telefoon algemeen	+32 (0)14 23 56 00

#### 3.3 Gerelateerde documenten

Naast dit document zijn er nog andere documenten die gerelateerd zijn aan dit document, te weten:

documentnr	omschrijving
Thiemo Cumps-presentatie.pptx	Twee wekelijkse presentatie voor het stagebedrijf
Thiemo Cumps-Logboek	Wekelijkse opvolging voor hogeschool en stagebegeleider.
handleidingen	Handleidingen.ZIP
Code Stage	Stage.Zip



## 4 Stagebedrijf

3-it is een IT bedrijf dat is opgericht in 2007, en actief is vanuit Oevel (Westerlo). Het bedrijf telt zo'n 40 tal medewerkers en is actief in Vlaanderen en Nederland. 3-it is zowel lid van de Cronos groep, als de Hyperion groep. 3-it ondersteunt KMO's en ondernemers in hun complete IT-behoefte. Dit doen ze door middel van 3 pijlers.

De eerste pijler is Consultancy, advies en beheer, met andere woorden de kennis en inzichten om de juiste IT oplossing te bepalen en deze te implementeren en beheren.

De tweede pijler is Innovatieve technologieën, daar IT en de achterliggende technologieën altijd evolueren is 3-it altijd actief en geïnteresseerd in de nieuwste trends en technologieën.

De laatste pijler is Opleidingen, rekrutering en stages. 3-it wil opkomend IT talent mee ondersteunen en de kans geven om nieuwe kennis te leren en ervaringen op te doen.



## 5 Stageopdracht

Bij een bestaande klant (logistieke dienstverlener) wordt het logistieke proces van één magazijn volledig geautomatiseerd door het aansturen van automatisch aangedreven rollenbanen en mobiele rekken. De eerste versie van dit project is reeds in gebruik, in de volgende fase is het de bedoeling om, naast enkele nog uit te voeren optimalisaties, de magazijnmedewerkers meer inzicht te geven in de operationele flow d.m.v. een nog volledig op te zetten monitoring.

Het ontbreken van deze monitoring zorgt ervoor dat het voor een logistieke medewerker niet zichtbaar is waar mogelijke problemen zich bevinden. Hierdoor bestaat het risico dat het volautomatische proces wordt stilgelegd of afgebroken. De medewerker kan dan ook geen gepaste correctie uitvoeren.

De sturing van deze automatisatie werd gemaakt in .NET met een permanente link naar het beheer programma (WMS op basis van Microsoft Business Central) waar alle productdata zich bevindt. Voor het aansturen van de rollenbanen wordt gebruik gemaakt van een Siemens PLC, aangevuld met o.a. Datalogic scanners en hoogtelezers.

Het doel van de stage is dat de student een oplossing biedt d.m.v. een monitoring systeem dat beschikbaar is voor de logistieke medewerker. De verschillende fases van het logistieke proces moeten gevolgd kunnen worden. Op basis van deze informatie/visualisatie kan er dan door een medewerker een (manuele) handeling uitgevoerd worden om te voorkomen dat het proces wordt onderbroken of om het terug op te starten.

De student levert op het einde van zijn stage een compleet werkend dashboard op (technologie nog onbepaald) waarin alle processen van de logistieke flow zijn opgenomen. De data die hiervoor noodzakelijk is, wordt gehaald uit de verschillende systemen die reeds bij de klant aanwezig zijn.

### 5.1 Aanleiding van de opdracht

Korte beschrijving van de probleemstelling en de aanleiding van het probleem bij de klant. Beschrijf:

Het ontbreken van deze monitoring zorgt ervoor dat het voor een logistieke medewerker niet zichtbaar is waar mogelijke problemen zich bevinden. Hierdoor bestaat het risico dat het volautomatische proces wordt stilgelegd of afgebroken. De medewerker kan dan ook geen gepaste correctie uitvoeren. Naast het ontbreken van monitoring zijn er nog enkele optimalisatie problemen. Het eerste is dat de SSCC (barcode) niet altijd wordt ontvangen. Dit namelijk door een netwerkprobleem wanneer de scanners geen verbinding maken. Naast dit is de huidige applicatie gemaakt in .NET wat omgezet moest worden naar Node Red. Tenslotte zijn er nog enkele optimalisatie problemen deze zijn: de conveyor maakt soms valse paren en single pallets mogen niet weggezet worden in een muur locatie





## 5.2 Eisen en wensen

Het doel van het toevoegen van de monitoring is het logistieke proces van het automatisch warehouse efficiënter te maken. Dit op vlak van 3 belangrijke zaken te verbeteren die momenteel al aanwezig zijn in het pakhuis. Namelijk: de conveyor, logging en visualisatie omtrent de verschillende gangen. We kunnen dit project een succes noemen wanneer wij voor de 3 onderstaande punten het doel bereikt hebben.

### 5.2.1 Logging dashboard

Op dit moment is er een logging aanwezig, maar aangezien deze niet altijd zo duidelijk is om te lezen, wil men deze verduidelijken. Een leuke toevoeging zou zijn dat er een zoekbalk komt waarmee er gezocht kan worden op bepaalde kernwoorden in een file. Door deze elementen toe te passen wordt de logging verduidelijkt, wat voor de helpdesk een grote help zou vormen.

### 5.2.2 Conveyor

Tijdens het proces van de conveyor kan er wel eens iets fout gaan. Zo gebeurt het soms dat de barcode van het pallet niet kan worden gescand. In de huidige installatie gaat er gewoon een blauwe lamp branden wanneer er iets fout loopt, maar helaas geeft dit niet veel informatie. ~~wat er voor zorgt dat~~ De werknemer weet dat er iets mis is, maar moet nog steeds hulp gaan zoeken zodat hij/zij het probleem kunnen oplossen. Het zou daarom interessant zijn om een dashboard te hebben waar de werknemer alle nodige informatie kan zien, zodat bij een probleem de kans groter is dat hij het probleem zelf kan oplossen. Tenslotte zou de conveyor animeren een leuke bonus zijn aangezien een afbeelding met animatie veel meer kan zeggen dan woorden.

### 5.2.3 Automatische rekken

Het valt al eens voor dat paletten op een verkeerde plaats worden gezet. Medewerkers die rondrijden met hun heftruck kunnen hun lading enkel lossen op de daarvoor voorziene locatie als de rekken ook effectief automatisch open gaan.

Door onduidelijkheid welke gang openstaat of moet opengaan, worden paletten verkeerd gezet. Als wij dit gaan verduidelijken door een dashboard te maken met de nodige informatie, worden de paletten correct op de juiste plaats gezet waardoor er later problemen worden vermeden. Een toffie bonus hierbij kan zijn dat er ook een grondplan aanwezig is waar je de exacte status van alle rekken kan zien.





## 5.3 Fasering

### 5.3.1 Analyse

Voor de analyse van dit project heb ik besloten om te werken met de MOSCOW-methode . Deze methode heeft als doel om prioriteiten te stellen. Aan de hand van “must”, ”Should”, “Could” en “Won’t” haves. Dit zorgt ervoor dat je een duidelijke project scope krijgt van wat er zeker moet gedaan worden.

#### 5.3.1.1 *Must have*

- Alle .NET functionaliteiten in Node red
- Node red logging pagina voor het volledige system
- Conveyor dashboard
- Status rekken dashboard

#### 5.3.1.2 *Should Have*

- Animatie van hoe de paletten over de conveyorband gaan
- Animatie van hoe de automatische rekken open gaan en sluiten
- Zoekbalk op kernwoord voor de logging pagina

#### 5.3.1.3 *Could Have*

- Kleur legende voor de logging pagina
- De werkelijke stok bijhouden op de status rekken dashboard

#### 5.3.1.4 *Won't Have*

- Interactief dashboard: de werknemers moeten enkel lezen van de dashboard
- Geen monitoring van de heftrucks in het warehouse



### 5.3.2 Design

Het design gedeelte tijdens mijn stage was minder van toepassing aangezien de eerste fase van dit project al live staat bij een klant. Fase 2 van dit project was ik relatief vrij om te kiezen met welke technieken ik mijn oplossing uitwerk. Zolang ik maar in Node-Red werk en ik alle hoofdfunctionaliteiten werkend krijg maakt de technologie niet uit.

#### 5.3.2.1 Partners

Onderaan vind je de partners die van de devices dat zijn gebruikt in de installatie met een link naar de website indien u nog meer informatie over hun wilt weten.

##### 5.3.2.1.1 Avocom

- **Doel** - Internal network, Wifi, internet, router, firewall
- **Website** - <https://www.avocom.be/>

##### 5.3.2.1.2 Ducker

- **Doel** - Conveyors
- **Website** - <https://duecker.biz/>

##### 5.3.2.1.3 Dexion

- **Doel** - Automatische rekken
- **Website** - <https://www.dexion.com/>

##### 5.3.2.1.4 Datalogic

- **Doel** - Scanners
- **Website** - <https://www.datalogic.com/>

##### 5.3.2.1.5 Altebra

- **Doel** - Fire Protection/Sprinklers
- **Website** - <http://www.altebra.be/>

#### 5.3.2.2 Devices

##### 5.3.2.2.1 Afstands sensors

Elke conveyor heeft zijn eigen afstand sensor om de hoogte van het pallet te meten.

- **Merk:** Datalogic
- **Type:** S85-MH-5-Y
- **Handleiding:** [s85\\_distancesensors\\_manual\\_eng.pdf](#)
- **Connection protocol:** 4-20 mA current loop





#### 5.3.2.2.2 Moxa Ethernet remote I/O

De 2 afstands sensors zijn geconnecteerd met de Moxa zodat we ze via het network kunnen connecteren

- **Merk:** Moxa
- **Type:** IoLogik E1242
- **Handleiding:** [ds-av900-en.pdf](#)
- **IP:** 192.168.45.20
- **Power supply:** 139 mA @ 24 VDC
- 4 Analogue inputs
- 4 Digital inputs
- Ethernet/MODBUS/TCP/EtherNet/IP



#### 5.3.2.2.3 Scanners

Elke Conveyor heeft 2 scanners die werken als 1 om de QR code van het binnenkomende pallet in te lezen.

- **Merk:** Datalogic
- **Type:** DS-AV900
- **Handleiding:** [ds-av900-en.pdf](#)
- **IP Scanner conveyor 1:** 192.168.xx.xx
- **IP Scanner conveyor 2:** 192.168.xx.xx



#### 5.3.2.2.4 Conveyors

- **Merk:** Ducker
- **Manual:** [2021-03-27\\_2002472\\_socketinterface1v1.pdf](#)
- **IP conveyor 1:** 192.168.xx.xx
- **IP conveyor 2:** 192.168.xx.xx

#### 5.3.2.2.5 Automatic racks

Er zijn 5 verschillende gangen aanwezig in het warehouse A,B,C,D,F

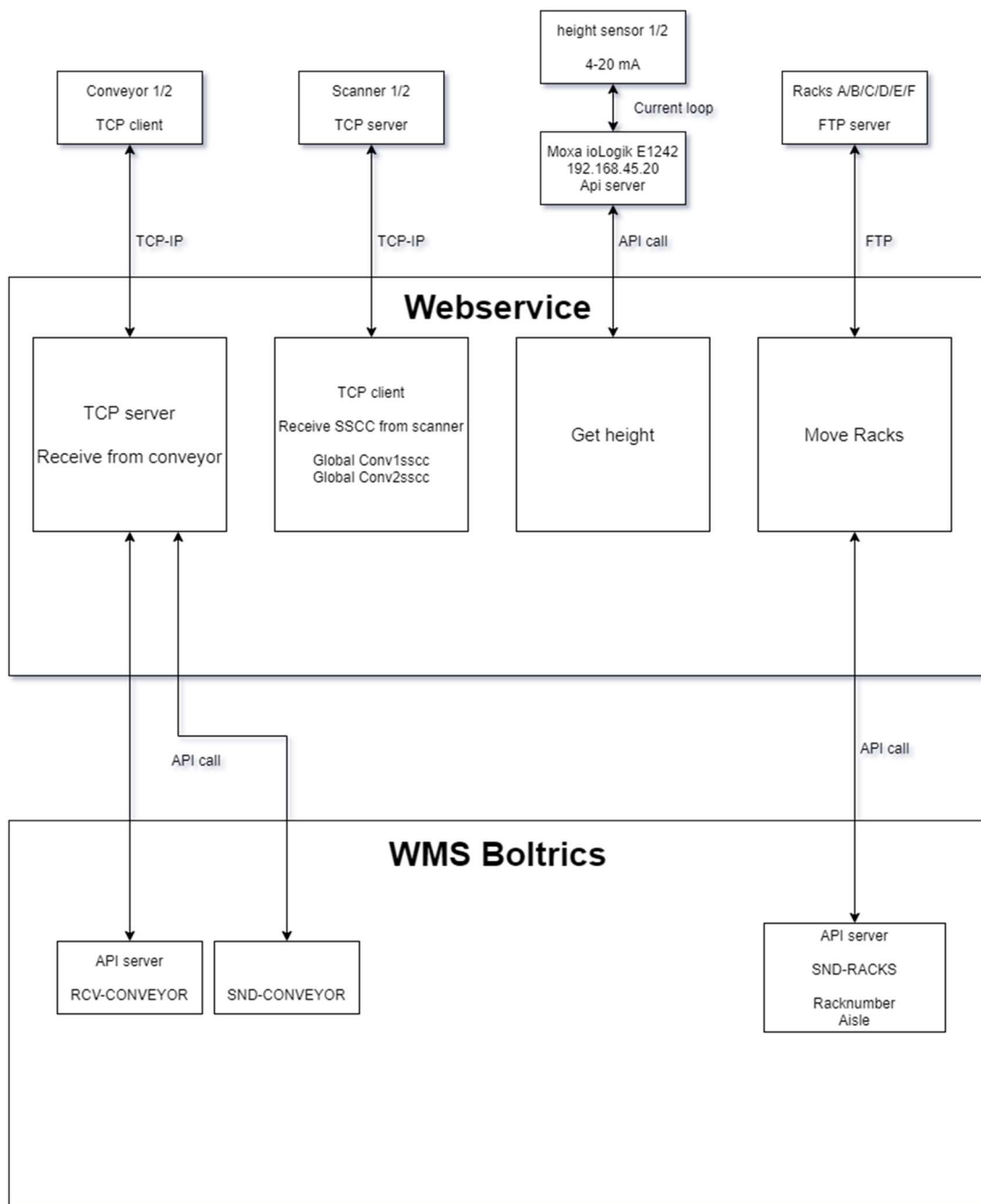
- **Merk:** Dexion
- **Type:** Movo
- **Handleiding:** [dexion\\_movo\\_interface\\_m258\\_ftp\\_wms.pdf](#)
- **IP Rack A:** 192.168.xx.xx
- **IP Rack B:** 192.168.xx.xx
- **IP Rack C:** 192.168.xx.xx
- **IP Rack D:** 192.168.xx.xx
- **IP Rack F:** 192.168.xx.xx





### 5.3.2.3 Fase 2 : mijn stage

De onderstaande afbeelding is representatie van hoe het huidige systeem werkt en welke verschillende methodes er worden gebruikt. Tijdens mijn stage ga ik hier nog 4 schermen aan toevoegen welke zijn: 2 voor de dashboards van de conveyor, 1 voor de logging en tenslotte nog 1 voor de status van de rekken.





### 5.3.3 Ontwikkeling

Onderaan kan je meer uitleg vinden over welke development omgevingen er zijn gebruikt tijdens het realiseren van de stage opdracht.

#### 5.3.3.1 Node red

Node Red is een flow georiënteerde programmeringstool die gebaseerd is op javascript blokken. Je kan deze blokken hardware, API, online services op toestellen zeer eenvoudig verbinden met elkaar. Er is ook een heel grote community achter Node Red waardoor je voor bijna alles al een library kan vinden. Indien je zelf nog een bepaalde functie moet bekomen kan dit eenvoudig door blokken te combineren in een subflow en/of in de functie blok zelf javascript code te schrijven. Tijdens mijn stage heb ik Node Red gebruikt op een docker pot van Elevate-IT. Dit zorgt ervoor dat je altijd en overall aan de Node Red code kan wat heel handig was wanneer je wisselt tussen remote en op kantoor werken.

#### 5.3.3.2 UIbuilder Visual studio Code (Html, Css en javascript)

UIbuilder is een Node Red library die ik heb toegevoegd dat er voor zorg dat u van 0 een dashboard kan opbouwen. Dit aan de hand van bootstrap, HTML , CCS en javascript. Aangezien UIbuilder niet altijd even goed werkt heb ik het grootste deel van het webdesign in Visual studio code gemaakt aangezien dit een betere workflow heeft. Tenslotte heb ik de Node Red input verbonden met het web dashboard.

#### 5.3.3.3 Version control github

Tijdens de eerste weken van het project was er geen version control aanwezig in het project. Al snel merkte ik dat dit toch wel nodig was gedurende dit project. Voor Node Red was er geen probleem aangezien die omgeving elke dag wordt geback-upt. Ook om github te laten werken moet je dit aanzetten in de Node red omgeving. Eens deze optie aan stond heb ik ervoor gezorgd dat er altijd een up-to-date github repository aanwezig was door dagelijkse commits uit te voeren.

#### 5.3.3.4 Communicatie















Communicatie is zeer belangrijk gedurende een project. Dit werd tijdens de stage gedaan door een dagelijkse stand-up meeting. Wanneer er een vraag of een goed idee is om toe te voegen aan het project werd er tussen mij en mijn stagementor gecommuniceerd via Microsoft teams dit heeft voor een vlotte communicatie gezorgd.



## 6 Planning

### 6.1 Overzicht

Onderaan vindt u een schematische voorstelling van de verschillende sprint met de bijbehorende taken. Ook kan je er mijn ingeschatte planning van vinden. Deze komt in het algemeen goed overeen met de werkelijke prestaties die je in mijn logboek kan vinden. Het grootste struikelblok tijdens de planning was het veranderen van de prioriteiten. Dit gebeurde wanneer er een nieuw idee kwam die een grotere meerwaarde kon betekenen dan een bestaande taak. Bijvoorbeeld de zoekbalk op de SSCC is er pas later bijgekomen.

Aa Projects	≡ Sprint	📅 Timeline
 Basis Node Red	sprint 1	March 3, 2022 → March 6, 2022
 Random Height functie MOXA	sprint 1	March 8, 2022 → March 11, 2022
 Dummy Scanner functie	sprint 1	March 11, 2022 → March 16, 2022
 XML → SFTP Server <span>💬 1</span> <a href="#">🔗 OPEN</a>	sprint 1	March 16, 2022 → March 18, 2022
 Dummy Wms bericht Conveyor	sprint 1	March 18, 2022 → March 23, 2022
 Template + layout van de dashboards	sprint 2	March 23, 2022 → March 30, 2022
 Moxa Dummy API	sprint 1	March 25, 2022 → March 30, 2022
 Dashboard : Conveyor 1 en 2	sprint 2	March 25, 2022 → March 30, 2022
 Dashboard : status rekken <span>💬 1</span>	sprint 2	March 30, 2022 → April 4, 2022
 Logfile System:	sprint 2	April 4, 2022 → April 6, 2022
 Logfile Conveyor 1 en 2	sprint 2	April 6, 2022 → April 8, 2022
 Zoekbalk op SSCC <span>💬 2</span>	sprint 3	April 11, 2022 → April 13, 2022
 Status rekken weergeven met svg	sprint 3	April 13, 2022 → April 30, 2022
 status conveyor weergeven met animatie <span>💬 1</span>	sprint 3	April 30, 2022 → May 13, 2022
<a href="#">Logpagina meer leesbaar maken d.m.v kleuren</a>	sprint 3	May 16, 2022 → May 20, 2022



## 7 Plan van aanpak

### 7.1 Sprint 1

In de eerste fase van dit project heeft men de hele installatie in .NET gemaakt. Deze applicatie werkt goed, maar aangezien in fase 2 de logging en monitoring wordt gemaakt in Node Red is het wel beter om de huidige applicatie ook in Node Red om te zetten. Hiermee worden toekomstige aanpassingen eenvoudiger en heb je 1 standard over het gehele project.

#### 7.1.1 Voorgesteld taken

Wanneer men kijkt naar de huidige installatie kan men deze in 4 delen opsplitsen:

##### 7.1.1.1 Conveyor:

De conveyor krijgt verschillende data binnen; dit van de scanner, de hoogte sensors en van het WMS systeem. In dit gedeelte gaan we deze ontvangen en bewerken indien nodig. Tenslotte zal deze het bericht naar de WMS sturen. Om zo de WMS response te ontvangen.

##### 7.1.1.2 Scanners:

De scanners krijgen een input message binnen, hieruit moet de SSCC(de barcode) gehaald worden. Deze barcode kan men gebruiken bij de conveyor.

##### 7.1.1.3 Automatische rekken:

De automatische rekken werken aan de hand van een Txt file met als inhoud de gang en errorcode. Om te bepalen welke gang, error en area het systeem moet gaan openen wordt er een XML ontvangen van het WMS waarin je alle nodige informatie kan gaan uithalen.

##### 7.1.1.4 Moxa hoogtesensor:

De hoogte sensor krijgt een bepaalde waarde binnen, deze ga ik omvormen met de formule van de respectievelijke sensor om zo een leesbare hoogte in millimeter te bekomen. Deze informatie moet aan de hand van een API call beschikbaar zijn.







## 7.2 Sprint 2

Op het huidige moment is er niets van logging aanwezig. Dit zorgt voor enkele problemen wanneer er iets fout loopt. Een voorbeeld hiervan is wanneer er iets misloopt bij de conveyor gaat er een blauwe lamp signaleren dat er actie ondernomen moet worden. De werknemers hebben geen manier om te controleren wat er juist fout is en verliest hierdoor veel tijd. Het doel van deze sprint is om de logging werkend te krijgen met dummy data. Deze data gaat moeten gedisplayd worden op 4 schermen: 1 per conveyor, een scherm voor de status van de rekken en een logging scherm. De combinatie van deze 4 schermen zal er voor zorgen dat er monitoring voor het hele systeem aanwezig is.

### 7.2.1 Voorgesteld werk

#### 7.2.1.1 Dashboard Conveyor 1 en 2:

Aangezien de enige vorm van monitoring een blauwe lamp is, zou het handig kunnen zijn met een dashboard te werken waarop alle belangrijke informatie te vinden. Deze informatie zijn: de hoogte, ConveyorCommand, SSCC, SKU, DuckerID, DuckerCounter.

#### 7.2.1.2 Logging:

Het doel van de logging is om het proces van het systeem stap voor stap te volgen. Vervolgens moet al deze informatie worden opgeslagen in een txt bestand dat wordt gedisplayd op een dashboard als een live feed.

#### 7.2.1.3 Status rekken:

Om de situatie van de rekken te vereenvoudigen zou het handig zijn dat je voor elke area kan zien welke gang er open staat. Indien er een error is moet deze ook worden vermeld op het dashboard. Bijvoorbeeld er is nog een persoon in de gang waardoor deze niet kan openen als de werknemer dit kan zien kan en wachten tot de correcte gang openen gaat of deze manueel openen om zo te voorkomen dat er problemen komen met de stock.

## 7.3 Sprint 3

De taken van sprint 1 en 2 zorgen ervoor dat er een werkende logging/monitoring van het systeem aanwezig is, maar het probleem is dat men hier heeft is dat er veel informatie staat op 1 scherm wat voor verwarring kan zorgen. Door de situatie af te beelden op de schermen krijgt men meteen ook al een visueel beeld, wat er voor zorgt dat het dashboard duidelijker en leesbaarder wordt. Iemand die voor de eerste keer het dashboard ziet, moet meteen kunnen zien wat er wordt getoond

### 7.3.1 Voorgesteld werk

#### 7.3.1.1 Logging

Aangezien elke stap van het systeem wordt gemonitord, is het soms moeilijk om achter een probleem te gaan zoeken bij een bepaald pallet of op een bepaald tijdstip. Om dit proces te vereenvoudigen is het toevoegen van een zoekbalk op keywords een zeer goede uitbreiding. Indien het mogelijk zou zijn zou een legende op basis van kleuren ook nog een goede toegevoegde waarde brengen.





### 7.3.1.2 Visualisatie

Zoals hierboven is vermeld zouden animaties op het dashboard de dingen veel duidelijker maken. Dit wordt gerealiseerd door een zijaanzicht bij de conveyor met alle informatie van het huidige pallet in de afbeelding te plaatsen. Op die animatie toon ik dan de laatste 4 a 5 paletten. Voor de status van de rekken zou het interessant zijn om het grondplan te tonen waar je kan zien welke rekken openstaan en wat hun errorcode is.





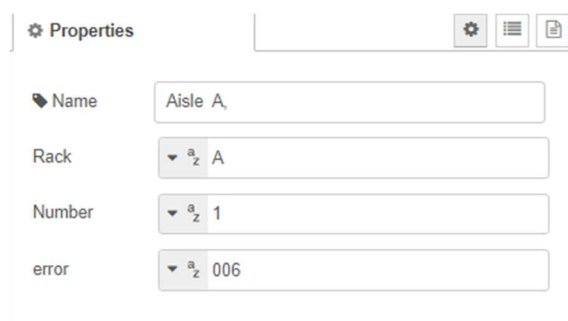
## 8 Realisatie

### 8.1 Global variable

Een global variable in Node Red is een variable die je overall in de node red omgeving kan oproepen.

### 8.2 Subflow

Een subflow kan je bekijken als een functie. Je kan hier zelf een combinatie van nodes maken en deze later oproepen met een subflow node. Je kan hier ook gebruik maken van environment variables wat een variable is die in de subflow wordt gebruikt, maar deze moet gedefinieerd worden in de subflow node om er een waarde aan te geven. In de afbeelding onderaan kan je een voorbeeld zien van een subflow node met Rack, Number en error als environment variables.



The screenshot shows the 'Properties' tab of a subflow node in Node-RED. It has four input fields: 'Name' with the value 'Aisle A', 'Rack' with a dropdown menu showing 'A', 'Number' with a dropdown menu showing '1', and 'error' with a dropdown menu showing '006'. Each dropdown menu has a small 'a\_z' icon next to it.

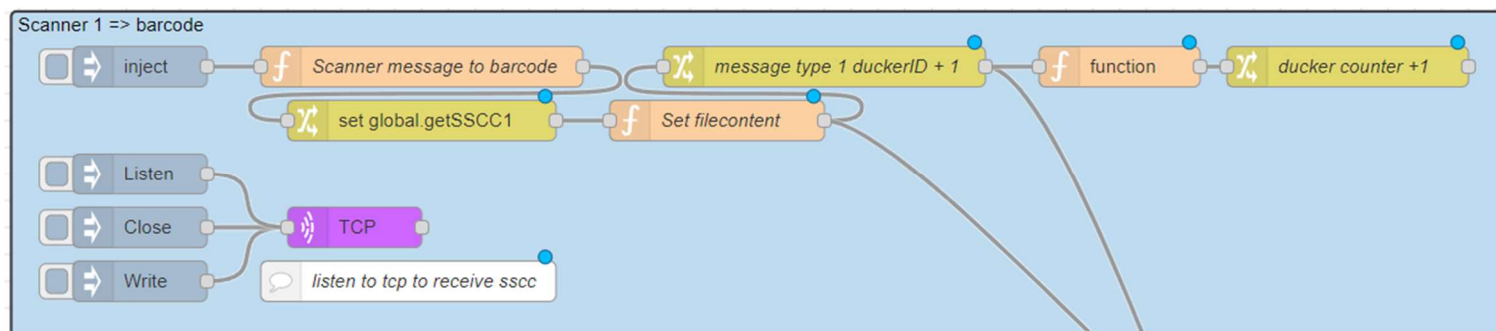
### 8.3 UIbuilder

UIbuilder is een Node Red library toe de mogelijk bied om een dashboard van scratch op te bouwen. Met de optie om Node Red payload messages als input voor het dashboard te gebruiken. Van deze flexibiliteit heb ik gebruikt gemaakt om een volledige dashboards op te bouwen met alle informatie en een duidelijke animatie van zowel voor de conveyor en de automatische rekken

### 8.4 Conveyor

#### 8.4.1 Scanner → barcode

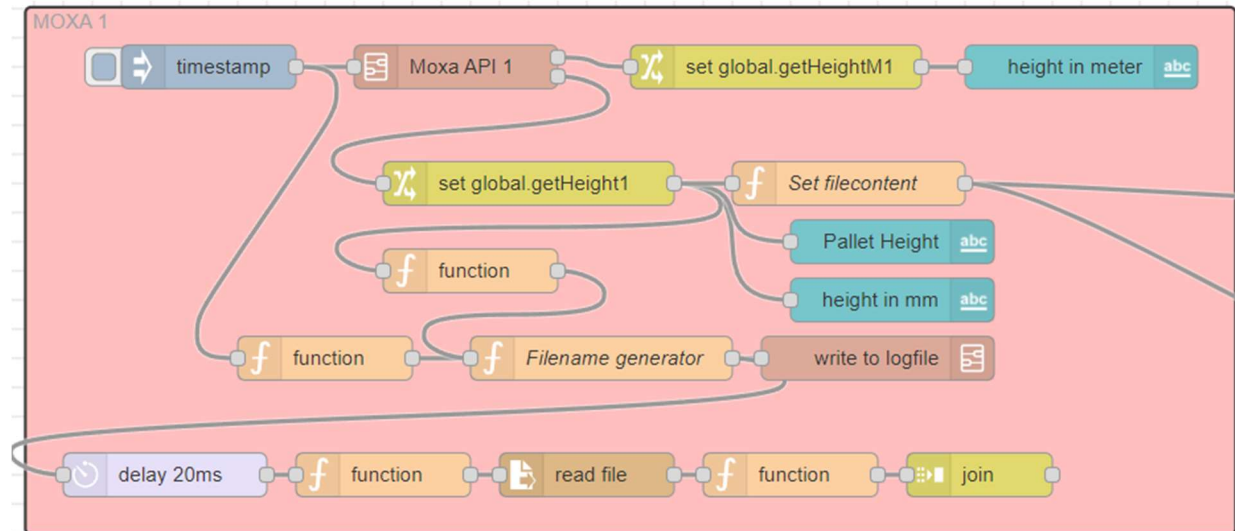
Het doel van deze functie is om de data van de scanners te simuleren. We gaan sample data sturen via een inject node. Vervolgens gaan we uit deze data de barcode halen met een functienode. Deze barcode gaan we hierna in een global variable zetten en we gaan deze barcode wegschrijven naar de Logfile. Elke keer als er een barcode wordt gescanned betekent dit dat er een nieuw pallet op de conveyor komt. Bij elk nieuw pallet moet de message type op 1 gezet worden en wordt de DuckerID met 1 verhoogd. Tenslotte moet de duckerCounter bij elke message met 1 verhoogd worden



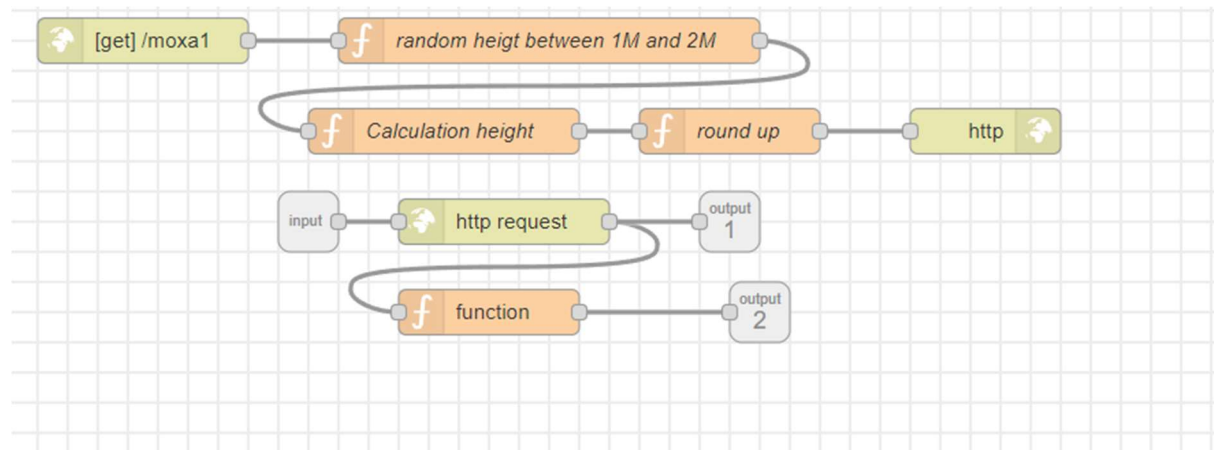


#### 8.4.2 Moxa Height API Call

Deze functie gaat de moxa hoogte sensors die aanwezig zijn op de conveyor belt simuleren. Dit gebeurt door een API call te maken naar een Random Height generator. Deze API call moet ook getoond worden op het dashboard en de logging van de conveyor en het systeem.



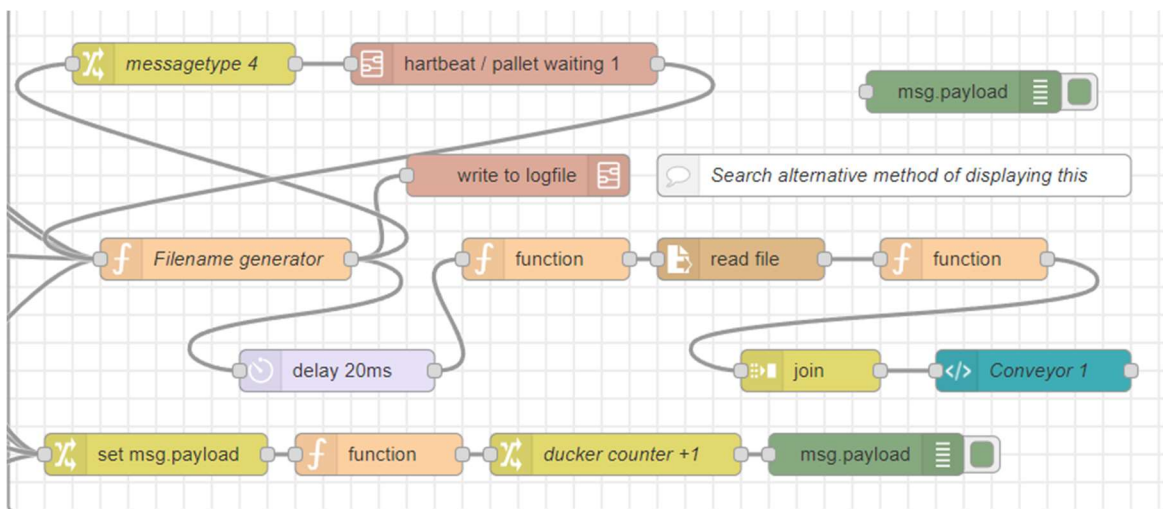
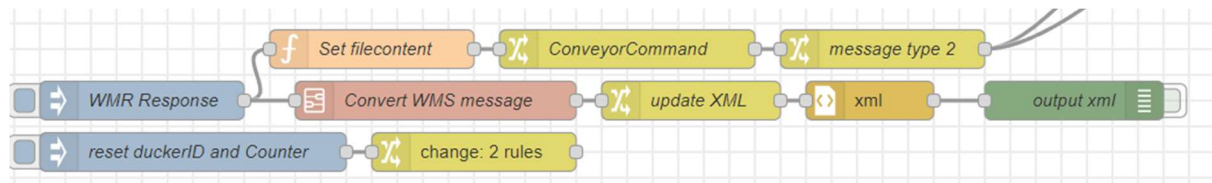
In de eerste node kan je het adres van de API call ingeven. Vervolgens gaat deze naar een random Height functie blok die een waarde tussen 1 en 2 meter weergeeft. Deze waarde is niet leesbaar, dus hier wordt een formule op toegepast en vervolgens naar boven afgerond. Dit geeft als resultaat een leesbare hoogte in millimeter





### 8.4.3 Test data

Om de logging werking van de conveyor volledig te simuleren heb ik een functie gemaakt met hard gecodeerde waarden die het bericht van de WMR gaan simuleren. Uit dit bericht wordt alle belangrijke informatie gehaald en vervolgens wordt deze informatie in de global variables gezet om weg te sturen naar de logging file. Onderaan de code is er ook nog een reset knop voor de DuckerID en Duckercounter indien deze te groot word zijn deze eenvoudig te resetten.



### 8.4.4 STAP 1: Dashboard Node Red

De eerste stap van de monitoring was een basis dashboard met alle nodige informatie. Dit is gerealiseerd door de node-red-dashboard library. Dit is een eenvoudige dashboard library met verschillende blokken om data te displayen met verschillende manieren bijvoorbeeld grafieken, tekst, afbeeldingen, ... . Data input van deze data is hard gecodeerd, maar de data wordt wel bewerkt zoals in de live omgeving. Dit zorgt ervoor dat het live zetten van deze testomgeving zeer snel en eenvoudig kan gebeuren.

Conveyer 2		Moxa 2	
Conveyer Number	2	height in mm	1532
Conveyer Command	730	height in meter 1.532	
Pallet SSCC	154250302501951717		
Pallet SKU	5		
Pallet Height	1532		
DuckerID	170		
Ducker Counter	374		

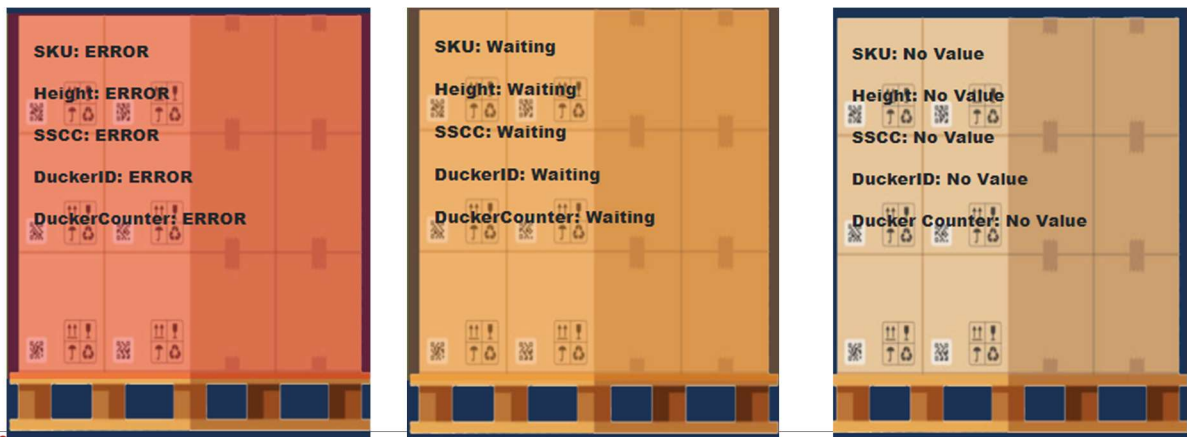




NETWERK VAN MENSEN

#### 8.4.5 STAP 2: Dashboard UIbuilder

Voor het dashboard van de conveyor heb ik UIbuilder gebruikt om een dashboard te bekomen met alle informatie van de paletten en een animatie van het zij aanzicht van de conveyorbelt. Het doel van deze animatie is, dat iemand die niet weet hoe de conveyor werkt toch duidelijk kan zien wat er gaande is op de conveyor. De gebruikte technieken voor dit te realiseren zijn HTML, CSS, Javascript en bootstrap 5. De UIbuilder node in gaat data binnen krijgen wat in dit geval hard gecodeerde informatie (SKU, Height, SSCC, DuckerID en DuckerCounter) is. Aan de hand van de SKU gaat deze controleren of deze gelijk is aan de SKU van het vorige pallet. Indien dit het geval is gaan deze 2 een paar vormen wat inhoud dat ze dezelfde kleur krijgen. Zoals u onderaan kan zien met de 2 groene paletten. Indien dit niet het geval is gaan de single paletten telkens tussen SKU1 en SKU2 afwisselen. Links bovenaan kan je ook aan zwart blokje zien waar je de Node Red input van de SKU en de errorcode kunt aflezen. Status Okay wacht het pallet op het WMS response bericht eens dit is ontvangen zal het pallet 1 positie naar links verplaatsen. Indien de error Waiting is wacht het pallet tot de error is opgelost bijvoorbeeld de barcode word niet goed gescanned. Tenslotte is er de Error status wat inhoud dat er een probleem in de workflow is. Hierbij moet de werknemer contact op nemen met een techniekier ok dit op te lossen. Je kan alle code bekijken in de gerelateerde documenten.

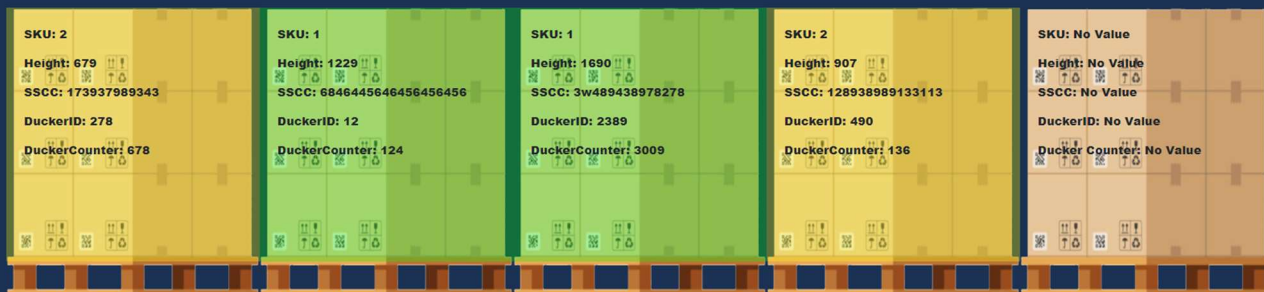


Conveyer 1

connection:  
IP:192.168.45.25

"1 OK"

Workflow: OK







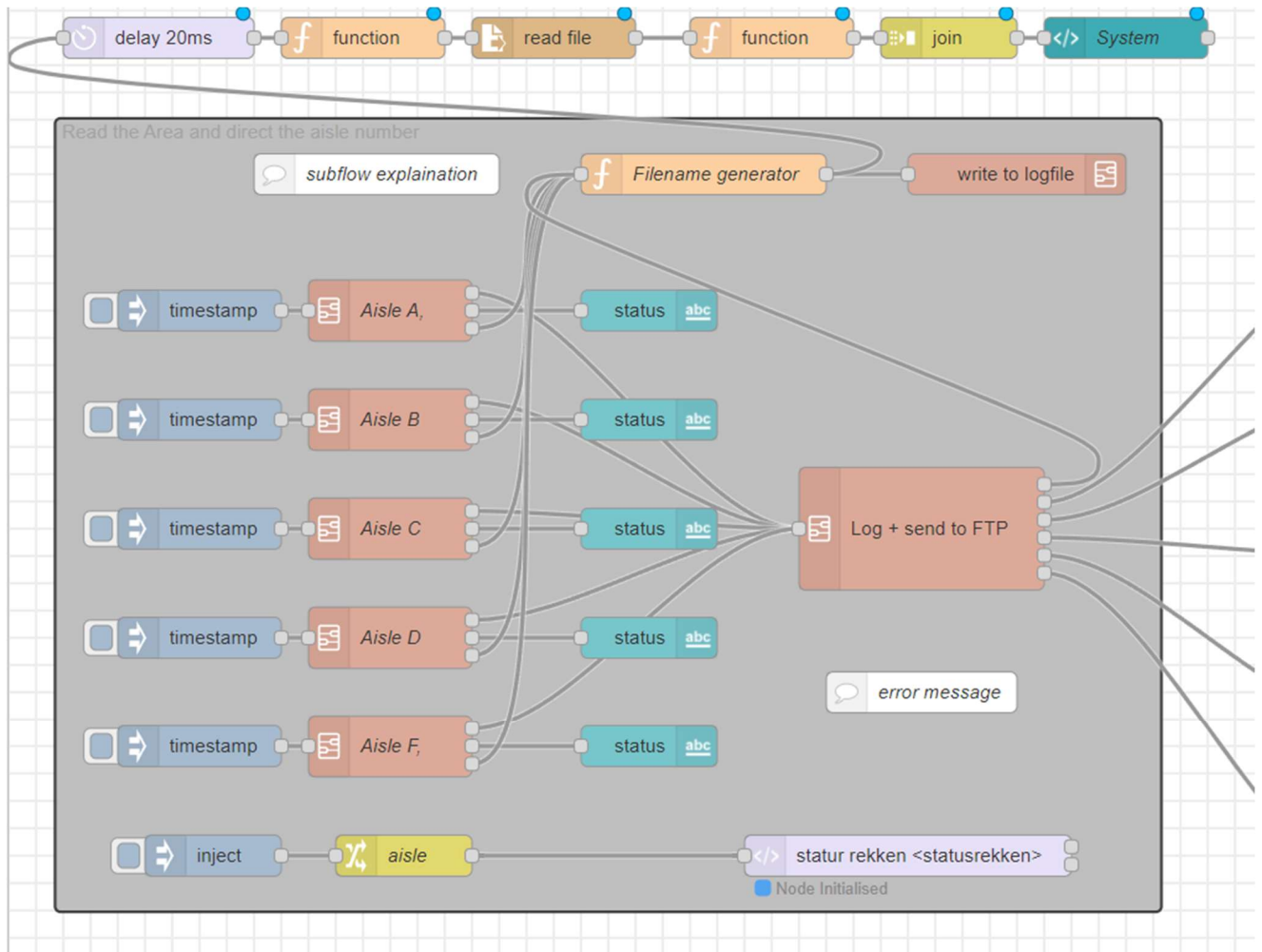
NETWERK VAN MENSEN

## 8.5 Automatische rekken

Om een gang van de automatische rekken te openen moet er een txt file met de gang en errorcode naar de SFTP sever van de area gestuurd worden. Om dit te realiseren is de functie in 2 delen opgesplitst; een deel voor de informatie te ontvangen van het WMS systeem en een volgende deel is de informatie in de vorm van een txt sturen naar een SFTP server via Node Red. Onderaan kan je een voorbeeld vinden van het WMS bericht in XML formaat en de txt file dat naar de SFTP wordt gestuurd.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns0:Message xmlns:ns0="www.boltrics.nl/OpenAisle:v1.00">
<ns0:OpenRack>
<ns0:RackNumber>B</ns0:RackNumber>
<ns0:AisleNumber>2</ns0:AisleNumber>
</ns0:OpenRack>
</ns0:Message>
```

```
003 Anlage betriebsbereit
XXXX_XXXXXX-----
1234567890123456
Anzahl Fahrbewegungen : 12969
Steuerung eingeschaltet : J
```

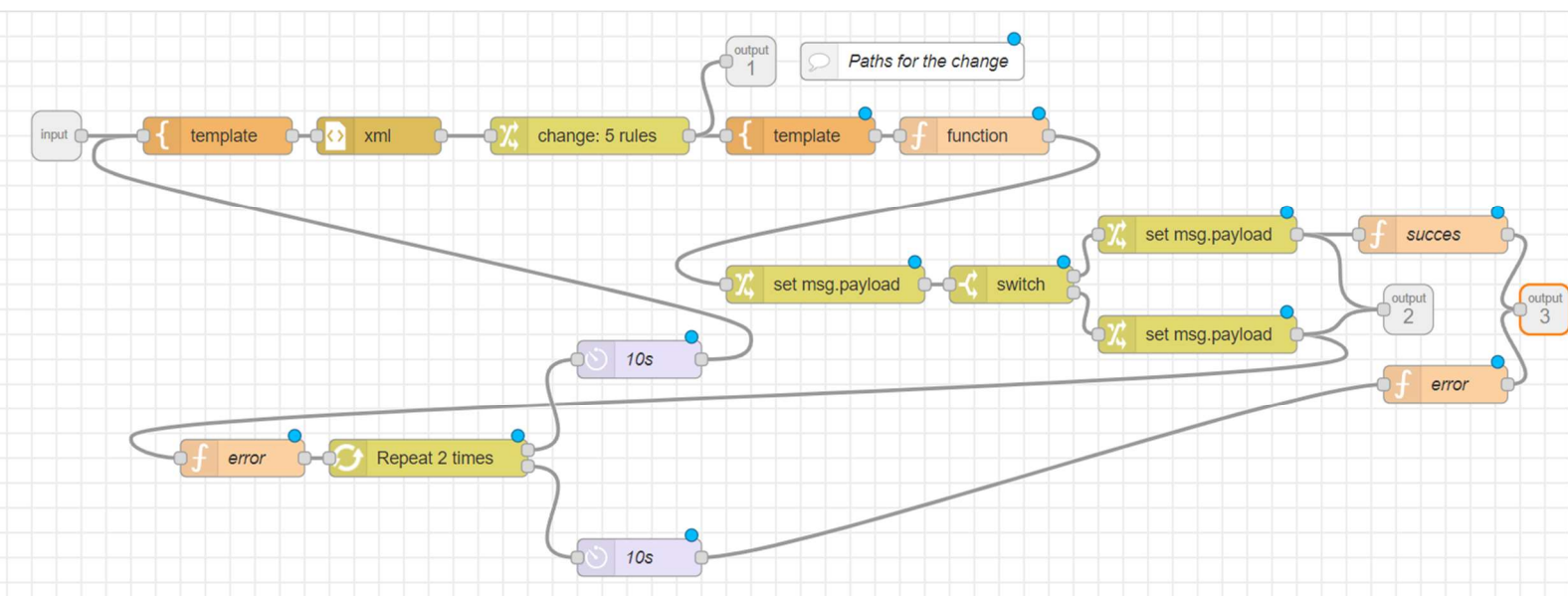




### 8.5.1 Aisle X subflow

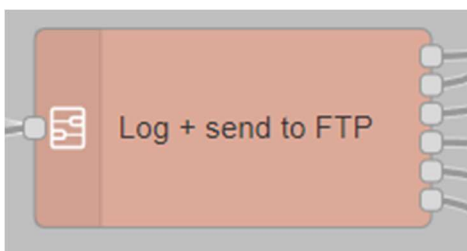
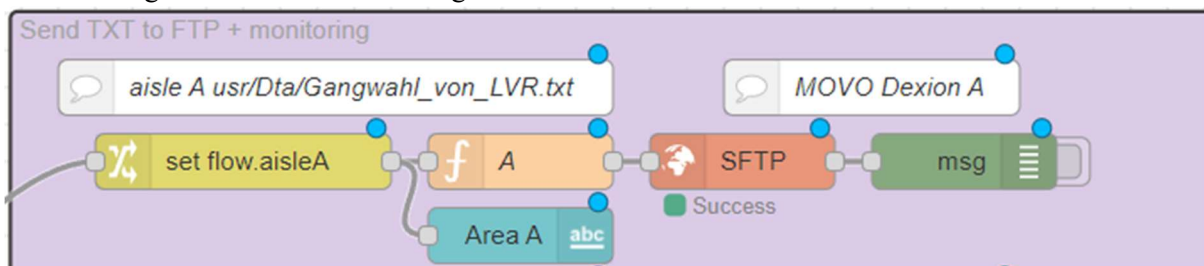
Deze subflow gaat het WMS bericht simuleren door een Template node de inhoud van de XML die het WMR systeem stuurt. In deze Template gaan we de Area, Aisle en Errorcode veranderen naar de meegegeven environment variables. Dit zorgt ervoor dat je elk mogelijke input kan simuleren.

Vervolgens gaat deze de errorCode controleren als deze 003 is gaat deze een success msg.payload doorsturen ander zal deze msg.payload een error message zijn. Tenslotte kan je onderaan zien indien er een error is dat het deze function 2 keer herhaald om de 10 seconden. Als het na 3 keer niet gelukt is word dit ook op de systeem logging aangetoont.



### 8.5.2 Log+send to FTP

Deze node gaat de input sturen naar de log file en in deze input gaat die controleren naar welke area de txt moet gestuurd worden en vervolgens de file ook naar de SFTP.





### 8.5.3 Dashboard Node Red

De eerste stap van de monitoring was een basis dashboard met alle nodige informatie. Dit is gerealiseerd door de node-red-dashboard library. Dit is een eenvoudige dashboard library met verschillende blokken om data te displayen met verschillende manieren bijvoorbeeld grafieken, tekst, afbeeldingen, ... . Data input van deze data is hard gecodeerd, maar de data wordt wel bewerkt zoals in de live omgeving. Dit zorgt ervoor dat het live zetten van deze testomgeving zeer snel en eenvoudig kan gebeuren.

Aisle			
Area A	1	status	ERROR
Area B	2	status	systeem is okay
Area C	5	status	systeem is okay
Area D	9	status	systeem is okay
Area F	10	status	systeem is okay



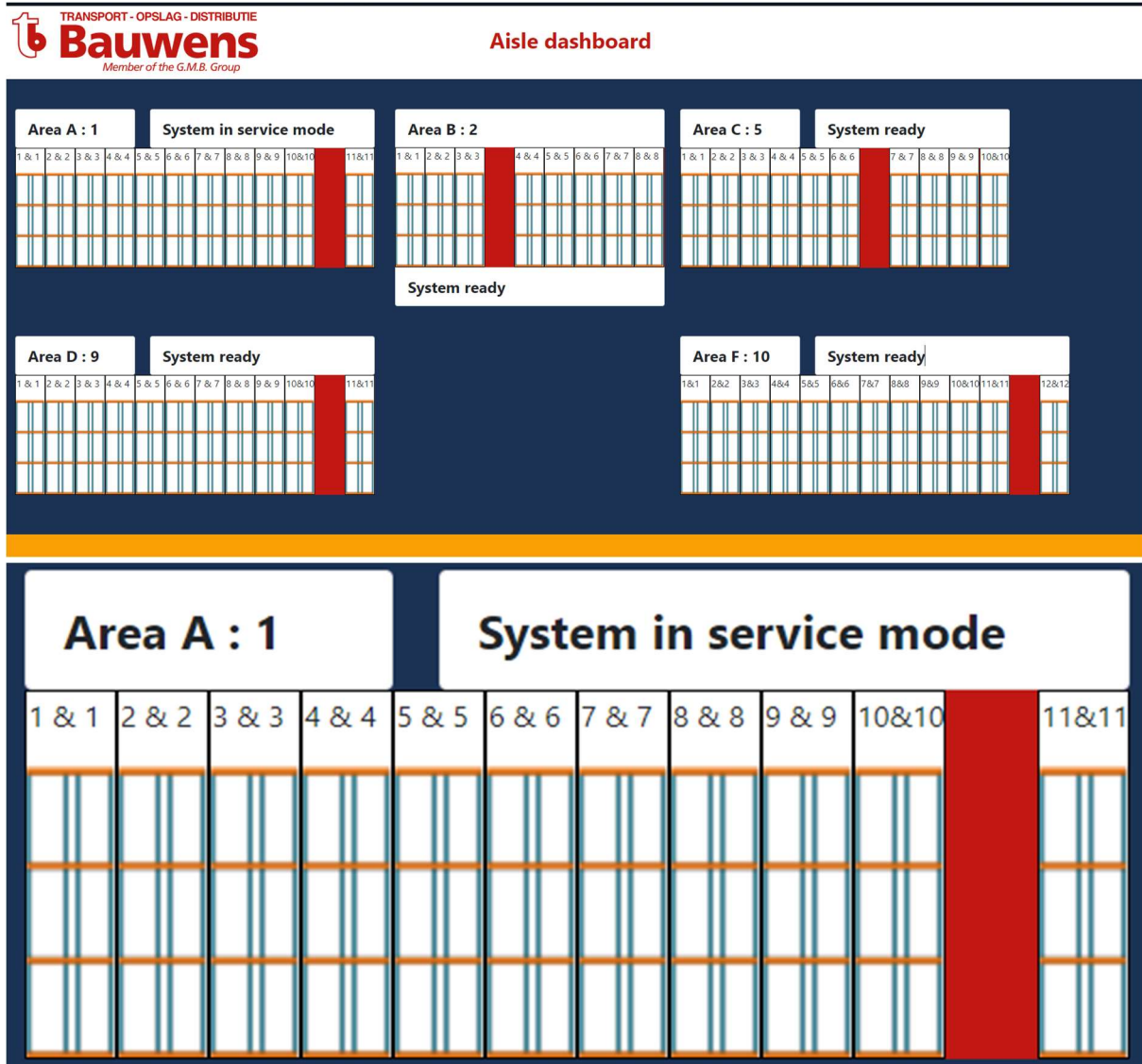




NETWERK VAN MENSEN

#### 8.5.4 Dashboard UIbuilder

Om het dashboard van de automatische rekken maken heb ik besloten om het bovenaanzicht te nemen aangezien dit voor een duidelijke representatie van het warenhuis geeft. De werking van de UIbuilder code is zeer simpel ik ga de 3 environment variables van de bovenstaande functie nemen als de input voor de UIbuilder. Met deze informatie gaat die de animatie controleren en de label veranderen naar de nieuwe input. Indien u de code wilt inlezen kan u deze vinden in de bijlage met extra comments hierover.





NETWERK VAN MENSEN

## 8.6 Logging

Voor de logging zijn er 2 verschillende logging feeds aanwezig op het dashboard. Er is system logging wat inhoudt dat alles systeem updates en berichten hier gelogd worden. In deze installatie zijn dit voornamelijk de API call van de Moxa hoogte sensors en de request om een gang te openen. De andere logging feed is 1 per conveyor belt. Op deze logging feed kan je het hele process van de conveyor belt lijn per lijn volgen. De logging is mogelijk gemaakt door 4 functies die onderaan worden beschreven

### 8.6.1 Set filecontent

De filecontent functie werkt zeer simpel bovenaan is de date functie dat er voor gaat zorgen dat je het exacte tijd stip tot op de seconde kan gaan oproepen. Vervolgens zie je dat de payload gelijk stelt aan “uur/minuut/seconde” gevolgd door text en alle variable met de informatie die we willen loggen. Tenslotte stuurt men dit weg als een payload message.

```
// Get the current time and convert it to text
var now = new Date();
var yyyy = now.getFullYear();
var mm = now.getMonth() < 9 ? "0" + (now.getMonth() + 1) : (now.getMonth() + 1); // getMonth() is zero-based
var dd = now.getDate() < 10 ? "0" + now.getDate() : now.getDate();
var hh = now.getHours() < 10 ? "0" + now.getHours() : now.getHours();
var mmm = now.getMinutes() < 10 ? "0" + now.getMinutes() : now.getMinutes();
var ss = now.getSeconds() < 10 ? "0" + now.getSeconds() : now.getSeconds();
// hh+ ":"+ mm + ":" + ss + " INFO Conveyor 1 "+
msg.payload = hh+ ":"+ mm + ":" + ss + " INFO Conveyor 1 "+ "Duckercounter: " +
| | | | msg.payload.DuckerCounter + " DuckerID: " + msg.payload.DuckerID + " New Pallet"
return msg;
```

#### Conveyor 1

```
11:13:21 INFO SSCC 1542503025019512365 received from scanner 1
11:13:22 INFO Conveyor 1 Duckercounter: 6781 DuckerID: 5 New
Pallet
11:13:23 INFO Conveyor 1 Duckercounter: 6781 DuckerID: 5 New
Pallet
11:13:25 INFO Response WMS:
11:13:25 INFO Message for conveyor 1 Succesfully received
11:13:25 INFO SSCC: 1542503025019512365 | Height: 1.4mm
11:13:25 INFO ConveyorCommand: [object Object] Sku:2
11:13:25 INFO DuckerID: 5 | DuckerCounter: 6781
11:13:25 INFO Conveyor 1 received message correctly SSCC:
1542503025019512365
11:13:35 INFO Conveyor 1 heartbeat message: 1
11:13:45 INFO Conveyor 1 heartbeat message: 2
```

#### System

```
14:04:17 INFO API Heigth from moxa 1
14:04:17 INFO Received height correctly
14:05:30 INFO API Heigth from moxa 1
14:05:30 INFO Received height correctly
14:07:58 INFO Request to move rack A received.
14:07:58 INFO Opening aisle 1 from rack A (3 attempts, 10sec
timeout)
14:08:03 INFO Request to move rack B received.
14:08:03 INFO Opening aisle 2 from rack B (3 attempts, 10sec
timeout)
14:08:03 INFO Succesfully openend the aisle
14:08:07 INFO Request to move rack A received.
14:08:07 INFO Opening aisle 1 from rack A (3 attempts, 10sec
timeout)
14:08:08 INFO Request to move rack A received.
14:08:08 INFO Opening aisle 1 from rack A (3 attempts, 10sec
timeout)
```



Nijverheidsstraat 48B  
2260 Oevel (Westerlo)  
T. +32 14 23 56 00



BE 0888.727.559  
BE03 7785 9660 1484  
info@3-it.be  
www.3-it.be



NETWERK VAN MENSEN

### 8.6.2 Filename generator

Het doel van deze functie is de filename te genereren en deze het juiste pad en inhoud te geven. Vervolgens is er nog een controle om te controleren of ~~het~~ de file bestaat .

```
// Get the current time and convert it to text
var now = new Date();
var yyyy = now.getFullYear();
var mm = now.getMonth() < 9 ? "0" + (now.getMonth() + 1) : (now.getMonth() + 1); // getMonth() is zero-based
var dd = now.getDate() < 10 ? "0" + now.getDate() : now.getDate();
var hh = now.getHours() < 10 ? "0" + now.getHours() : now.getHours();
var mm = now.getMinutes() < 10 ? "0" + now.getMinutes() : now.getMinutes();
var ss = now.getSeconds() < 10 ? "0" + now.getSeconds() : now.getSeconds();

// Generate out file name pattern
// msg.fname = "Log Conveyor 1" + yyyy + "-" + mm + "/" + dd + ".txt";
msg.fname = "Log Conveyor 1 " + yyyy + "-" + mm + "-" + dd + ".txt";
// Full filename with path for the file node later
msg.filename = "/data/Logs/" + msg.fname;

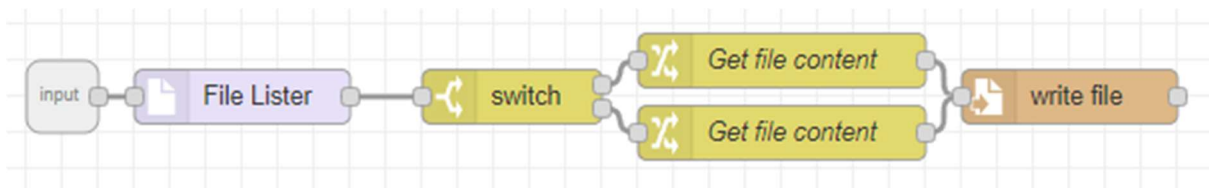
// We save the current payload into a different place on the msg object
msg.filecontent = msg.payload;

// We are passing the file name search pattern to fs node to tell us if the file exists or not
msg.payload = {"pattern":msg.fname};

node.status({fill:"blue",shape:"ring",text:msg.fname});
return msg;
```

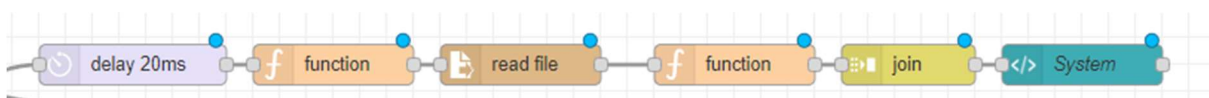
### 8.6.3 Write to logfile

In deze subflow wordt er gecontroleerd of de file al bestaat aan de hand van de File Lister en de switch node waar een Count functie is ingesteld. Indien de file niet bestaat wordt er een nieuwe file aangemaakt. Wanneer de logfile al bestaat wordt de nieuwe lijn aan de file toegevoegd.



### 8.6.4 Display to the live logging feed

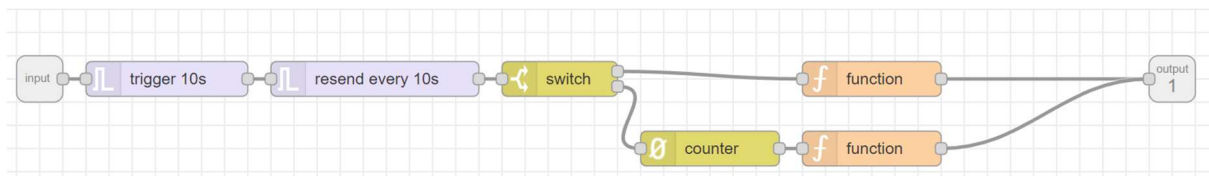
Dit is de functie om de logfile te tonen op de live feed op het dashboard. Dit is gerealiseerd door een function die het path van de logfile selecteert. Vervolgens gaat deze naar de read file node die logfile lijn per lijn inleest zodat er een <br> kan aan toegevoegd worden. Tenslotte worden alle lijnen terug tot 1 payload gevoegd met de join node zodat deze gedisplayed kan worden met de dashboard node.



### 8.6.5 Write to log file

Vanaf het systeem 10 seconde geen nieuw bericht heeft ontvangen gaat deze de heartbeat functie activeren. Deze functie gaat elke 10 seconde een bericht sturen naar de logging als controle dat het systeem nog steeds in werking is. Ik heb deze functie ook gecombineerd met de new pallet flow. Dit houdt in ~~sons~~ als er een nieuw pallet binnenkomt moet het WMS systeem dit verwerken en moet deze in de waiting flow gezet worden. Dit wordt gedaan door een switch node dat de SKU gaat controleren aangezien de SKU waarde de juiste message gaat sturen naar de logging. De verschillende SKU waardes zijn:

- 1,2 afwisselend voor normal flow,
- 0 waiting flow
- 5 error





## 8.7 Reflectie

Deze stage kan voor mij betreffen gezien worden als een succes. Alle elementen die er verwacht werden zijn er ook gerealiseerd met nog bepaalde extra's.

Ik heb de .NET functionaliteiten overgezet naar Node Red zodat je het hele proces binnen Bauwens kan simuleren. Het doel van deze simulatie was om realistische test data te krijgen op de dashboards.

Vervolgens heb ik alle gevraagde dashboard gemaakt en indien nodig hierbij nog aanpassingen gemaakt zodat zo duidelijk mogelijk zijn. Deze dashboard zijn 1 voor beide conveyors, een grondplan met alle automatische rekken van het warehouse en tenslotte nog een logging dashboard.

Tenslotte met de tijd die erover was heb ik aan de dashboard ook nog een animatie gemaakt. Aan de hand van deze animatie kan je het hele proces ook visueel volgen. Dit is een heel grote bonus aangezien een afbeelding veel meer kan zeggen dan woorden.

Alle realisaties staan ook klaar om in live omgeving klaar te zetten, maar helaas door de drukte bij Elevate-IT is dit niet binnen de stage periode gerealiseerd. Gelukkig is er documentatie aanwezig zodat ze zonder mijn hulp het systeem kunnen live zetten,

Door het integreren in een professionele omgeving heb ik op een korte periode enorm veel bijgeleerd. Dit zowel op softskills zoals plannen, vergaderen, communicatie, ... en technische skills. Tenslotte was deze stage ook een persoonlijke verrijking. Ik heb in een leuke nieuw bedrijf en collega's leren kennen. Wat zeker een goede aanloop is voor mijn professionele carrière.

Ik ben tevreden over mijn stage bij 3-IT/Elevate-IT hierbij wil ik ook beide bedrijven bedanken voor mij deze mooie kans te bieden om bij hun stage te mogen lopen