



Planningstool

Zwart op Wit

Jo Brunau

Kathleen Jansen

HBO5 Informatica- Optie Programmeren
Academiejaar 2016-2017



1 Dankwoord

Na een intensieve periode van zweugen en zweten is het zo ver. Met het schrijven van dit dankwoord leggen we de laatste hand aan ons eindwerk project, een planningstool voor Zwart op Wit.

Het is een periode waar we beiden veel hebben bijgeleerd. Niet enkel op gebied van programmeren, maar ook persoonlijke ontwikkeling en het samenwerken als team om zo tot een mooi product te komen.

Op de eerste plaats willen we elkaar graag bedanken. Door het uitwerken van dit project werden we een hecht team. We leerden veel bij en ondersteunden elkaar waar nodig. Communicatie is hierin zeer belangrijk.

Ook graag een dankjewel aan onze projectbegeleidster Maaike Van den Bulck. We konden steeds op haar rekenen voor een mooie opvolging en begeleiding van ons project.

Als laatste, maar zeker niet onbelangrijk willen we graag onze partners bedanken. Door het project waren we afgelopen maanden niet altijd even hard aanwezig maar dat werd door jullie met veel begrip opgevangen.

Voor iedereen een welgemeende dankjewel,

Jo Brunau & Kathleen Jansen

2 Samenvatting

Drukkerij Bulckens is een drukkerij waar via de website zwartopwit.com online drukorders kunnen worden besteld. De gegevens van de job worden geïmporteerd en verwerkt in Filemaker Pro voor Mac. Om de jobs in te plannen op de afwerkings machines van de drukkerij en een volwaardig planningstool mee uit te bouwen is deze software jammer genoeg niet geschikt. Er is nood aan een programma die deze jobs gemakkelijk importeert en verdeelt over de planning van de verschillende machines.

Om dit te bereiken zijn we gestart van een csv-file, gegenereerd in Filemaker die geïmporteerd wordt in ons programma. De gegevens van de jobs worden in de database geplaatst en de werktijden berekend. De jobs krijgen de status ‘todo’ en kunnen volgens leverdatum opgeroepen worden. De werknemer dient in te loggen en kan zo het programma gebruiken. In de todo-lijst is het mogelijk de jobs in te plannen op de machines. De jobs kunnen gestart worden waardoor een tijdsregistratie in de database komt, per job en per werknemer. Als de jobs klaar zijn verdwijnen ze uit de lijst. Werknemers, machines, afdelingen etc... kunnen worden toegevoegd en gewijzigd. Ook de parameters die gebruikt worden om de planningstijden te berekenen.

De structuur van het programma is MVC gebouwd (Model-View-Controller). Dit wil zeggen object-georiënteerd waardoor alle parameters van de verschillende objecten gemakkelijk gebruikt kunnen worden, aanpasbaar zijn en waardoor het programma gemakkelijk uit te breiden is. De gebruik van deze structuur is niet voldoende mogelijk in het oorspronkelijke programma Filemaker. Ons programma is dus de perfecte aanvulling en kan stand-alone draaien op een eigen database.

Inhoud

4	Planningstool – Zwart Op Wit	1
4.1	Wie zijn Zwart op Wit.....	1
4.2	Een planningstool?.....	1
4.3	Requirements.....	1
4.3.1	Business & Functional Requirements.....	1
4.3.2	Design Requirements	2
4.4	Technologie.....	2
5	Analys.....	3
5.1	Mindmap	3
5.2	User Stories & BPMN.....	4
5.2.1	Inloggen.....	4
5.2.2	Gebruikers beheren.....	5
5.2.3	Afdelingen beheren.....	6
5.2.4	Machines beheren.....	7
5.2.5	File Importeren.....	8
5.2.6	Job inplannen	9
5.2.7	Job Starten/Stoppen	10
5.3	UML klassendiagramma	11
5.4	Gegevensmodel ERD	12
6	Proof of concept.....	13
7	Technologische concepten.....	14
7.1	MVC.....	14
7.1.1	Introductie.....	14
7.1.2	Model	14
7.1.3	View.....	14
7.1.4	Controller	14
7.1.5	Voordelen.....	14
7.1.6	Nadelen	15
7.1.7	MVC in .net Core	15
7.2	Git.....	18
7.2.1	Introductie.....	18
7.2.2	Basishandelingen.....	18
7.2.3	Branching.....	19
7.3	Bootstrap.....	19

7.4	Rendering engine (razor).....	19
7.5	Identity	19
7.5.1	Introductie.....	19
7.5.2	Identity activeren	20
7.5.3	Registreren + Inloggen	21
7.5.4	Uitloggen	22
7.5.5	Uitbreiding userklasse	22
7.5.6	Rollen.....	23
7.5.7	Gebruikers beheren.....	26
7.5.8	Forgot password.....	27
7.6	Globalisation and localization	28
7.6.1	Introduction.....	28
7.6.2	Gebruik in ons project.....	28
7.7	Components	32
7.7.1	Introduction.....	32
7.7.2	Components vs. Partial View.....	32
7.8	Entity Framework.....	34
7.8.1	Inleiding.....	34
7.8.2	Wat doet Entity Framework?	34
7.8.3	Voordelen.....	34
7.9	CRUD	35
7.9.1	Index.....	35
7.9.2	Create	38
7.9.3	Read.....	39
7.9.4	Update.....	40
7.9.5	Delete	41
7.10	Patterns	41
7.10.1	Inleiding.....	41
7.10.2	Options pattern	41
7.10.3	Simple factory pattern (Calculation method)	42
7.11	CSV Import.....	43
8	Werking programma	45
8.1	Users beheren	45
8.1.1	Introductie.....	45
8.1.2	Index pagina	45
8.1.3	User CRUD	47

8.1.4	Zoeken	50
8.1.5	Navigeren	50
8.2	Departementen beheren	51
8.2.1	Introductie	51
8.2.2	Index pagina	51
8.2.3	Department CRUD	52
8.2.4	Zoeken	55
8.2.5	Navigeren	55
8.3	Machines beheren	55
8.3.1	Introductie	55
8.3.2	Index pagina	56
8.3.3	Machine CRUD	57
8.3.4	Zoeken	63
8.3.5	Navigeren	63
8.4	Time registers beheren	64
8.4.1	Introductie	64
8.4.2	Index pagina	64
8.4.3	Time register CRUD	65
8.4.4	Zoeken	68
8.4.5	Navigeren	69
8.5	Job planning	69
8.5.1	Jobs menu items	69
8.5.2	Jobs starten en stoppen	73
8.5.3	Job flow	73
9	Algemeen Besluit	75
10	Bijlagen	76
	Plan van Aanpak	77
11	Bronnen	81

3 Inleiding

Deze paper omschrijft ons project waarin we op zoek gegaan zijn naar een juiste oplossing voor een werkende planningstool voor drukkerij Bulckens / zwartopwit.com.

We zijn gestart met een mindmap waarin de ruggengraat van alle functionaliteiten aan bod komen. Van daaruit hebben we Userstories aangemaakt. Deze hebben we gebruikt om een idee te vormen van de klassen die we nodig hebben. BPMN schema's zijn ook uitgewerkt. Door middel van een 'proof of concept' zijn we erin geslaagd een framework te bouwen die ons de basis gegeven heeft voor de verdere uitwerking van het programma. De tijd die we gestoken hebben in het framework is van die aard geweest dat we de basis van het programma hebben kunnen bouwen, met name het gedeelte voor de nietmachine, van waaruit we in de toekomst makkelijk de verdere volledige uitbouw van het definitieve planningstool kan uitgewerkt worden.

Na het opstarten van een Git-Repository op GitHub zijn we gestart met de opbouw van het programma. In de opbouw hebben we rekening gehouden met de MVC-structuur om zo het programma overzichtelijk te houden en gemakkelijk uit te breiden.

Ten slot hebben we autorisatie en authenticatie voorzien. Deze eigenschap is cruciaal voor de verdere verwerking van de rest van het programma: de effectieve planningstool zelf. We zijn gestart met de opbouw van het tool voor de afdeling Stitch. Verdere uitbreiding naar de ander afdelingen is voor de nabije toekomst.

4 Planningstool – Zwart Op Wit

4.1 Wie zijn Zwart op Wit

Zwart op Wit is een bedrijf voor het online bestellen van drukwerk. Sinds 2000 is ‘www.zwartopwit.be’ de online afdeling van Drukkerij Bulckens. Een bedrijf met meer dan 80 jaar ervaring.

PrePress, drukken, printen, afwerken en inpakken. De volledige productie is in eigen beheer. Met hun uitgebreide collectie afwerkingsmogelijkheden vindt elke klant wat hij of zij nodig heeft.

4.2 Een planningstool?

Zwart op wit krijgt bestellingen online binnen. Nadat deze bestellingen ontvangen en verwerkt zijn, is de vertaling van het bestelde pakket drukwerk naar de werkvloer toe complex georganiseerd.

Gegevens die binnenkomen via de website worden geïmporteerd in Filemaker voor Mac. De administratieve opvolging gebeurt via Filemaker. De technische opvolging hiervan gebeurt via MultiPress. Deze software is speciaal ontwikkeld voor de verwerking van grafische gegevens.

De ingave in Multipress vereist voornamelijk een manuele ingave en werkt traag. Een ander nadeel is dat enkel bedienden met een grafische achtergrond deze toepassing kunnen gebruiken.

Onze planningstool maakt dat we Multipress gedeeltelijk kunnen uitsluiten. Ook kunnen de medewerkers voor het afwerken van het drukwerk op de werkvloer via een pc checken welke werken er op een bepaalde dag uitgevoerd moeten worden. De technische gegevens kunnen geraadpleegd worden en ook de tijd die nodig is om een drukwerk klaar te krijgen.

Als het werk afgewerkt is, kunnen we via een nacalculatie controleren wie welk werk heeft uitgevoerd en of dit binnen de vooropgestelde tijd is gebeurd. De verworven data kan gebruikt worden voor controle en bijsturing, maar het biedt ook een mogelijkheid om oplossingen te geven voor toekomstige orders.

4.3 Requirements

Bij het maken van een programma komen heel wat vereisten. We hebben een onderscheid gemaakt tussen enerzijds business & functional requirements, anderzijds design requirements.

4.3.1 Business & Functional Requirements

- Systeem moet op basis van externe Filemaker database gegevens importeren in eigen database
- Gegevens in database moeten bijgewerkt kunnen worden
- Systeem moet a.d.h.v. in de database de geschatte werktijden kunnen berekenen
- De applicatie moet beschikken over een beveiligd identificatieproces
- Er moet een onderscheid gemaakt worden tussen gebruiker en verantwoordelijke
- Nieuwe gebruikers moeten kunnen worden aangemaakt

- Nieuwe afdelingen en machines moeten kunnen worden aangemaakt in geval van uitbreiding van het bedrijf
- Gebruikers moeten een planplaats kunnen kiezen
- Systeem moet drukwerken aan de juiste planplaats toewijzen
- Gebruikers moeten af te werken drukwerk kunnen kiezen volgens levertijd en bestemming.
- Systeem moet werktijden registeren
- Systeem moet onderbrekingen registeren
- Gebruikers moeten opmerkingen kunnen toevoegen aan elke tijdsregistratie
- Systeem moet alle verzamelde gegevens naar de database schrijven en linken aan de gekozen drukwerken
- Systeem moet in realtime kunnen tonen welke gebruikers, welke drukwerken hebben opgestart
- Systeem moet queries kunnen uitvoeren op de database om nacalculaties uit te voeren volgens werknemer en drukwerk

4.3.2 Design Requirements

- De applicatie zal in eerste instantie standalone draaien. Later kan nog bekijken worden of verweven in de Filemaker database tot de mogelijkheid behoort.
- De applicatie dienst geschreven te worden in .NET
- Het resultaat is een web-applicatie
- De database wordt een mySQL

4.4 Technologie

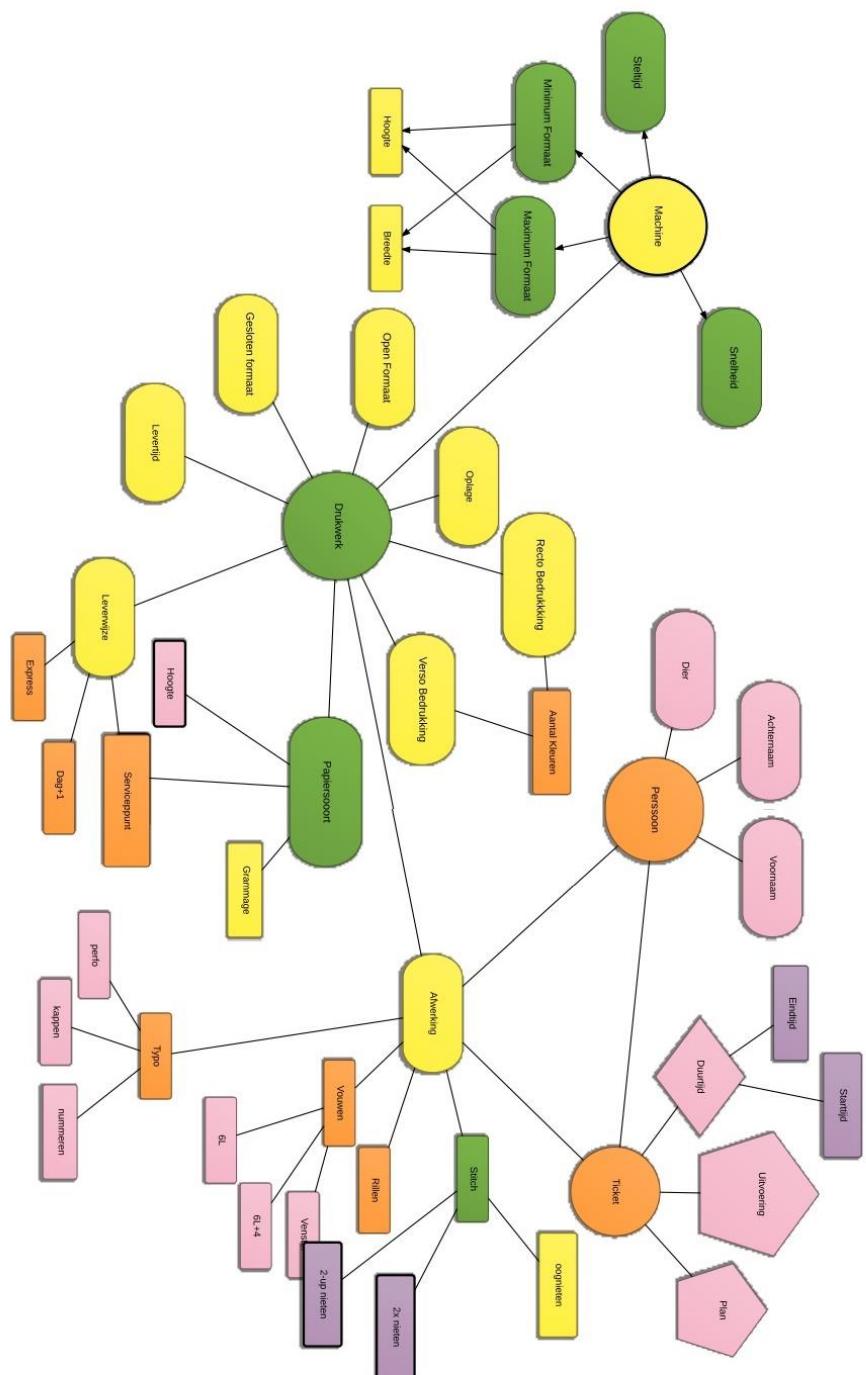
Doordat er in het huidige IT-landschap al gebruik wordt gemaakt van Linux based services (Ruby) en er een MySQL database beschikbaar is de keuze gevallen om technologieën in deze lijn te gebruiken. De data die door de planning tool gebruikt wordt kan worden teruggevonden in Filemaker.

- Asp.net Core c#
- MySQL
- Filemaker
- Razor

5 Analyse

5.1 Mindmap

We zijn onze analyse gestart met het maken van een mindmap. Dit gaf ons een bredere kijk op ons project. Waar denken we aan, wat moeten we allemaal voorzien,... Door het maken van een mindmap werden niet enkel de hoofdbestanddelen, maar ook de verbanden er tussen duidelijker.



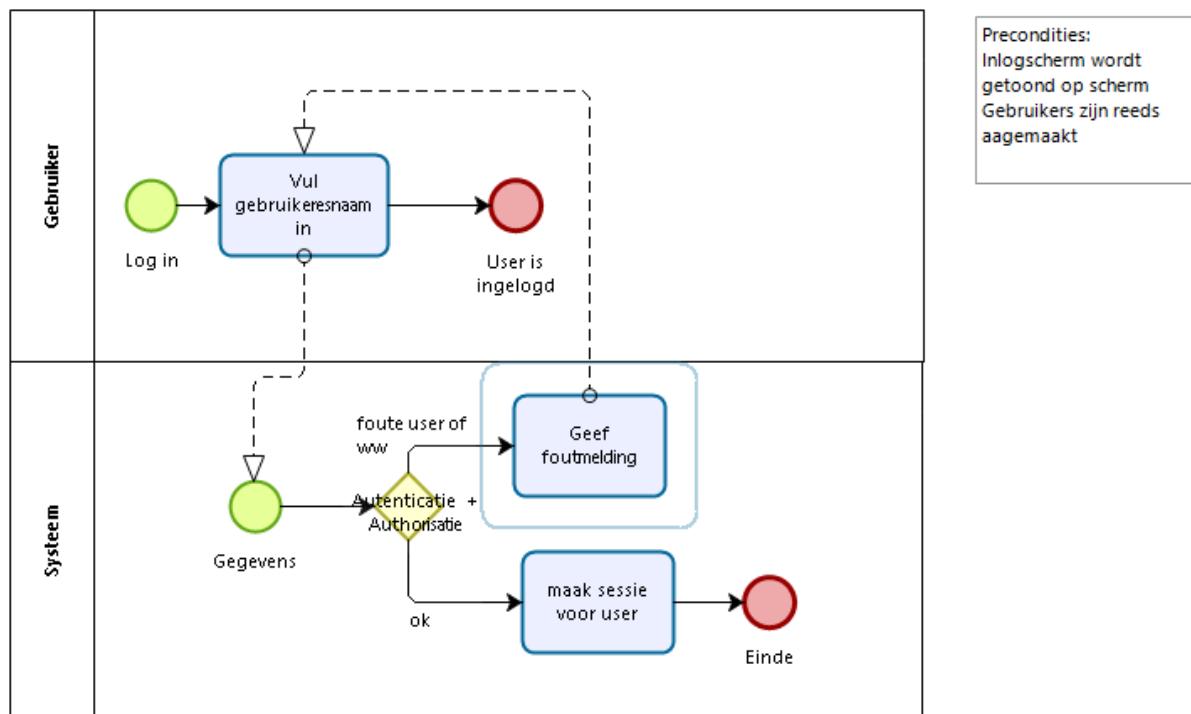
5.2 User Stories & BPMN

5.2.1 Inloggen

5.2.1.1 User story - Inloggen

Hoofddoel	<ul style="list-style-type: none"> Als gebruiker kan ik inloggen met een e-mailadres/username en paswoord.
Bijkomende vereisten	<ul style="list-style-type: none"> De username en paswoord worden gevalideerd door het systeem en geeft een melding indien de validatie mislukt is. Het systeem kijkt welke rol de gebruiker heeft die is ingelogd. Deze kan een verantwoordelijke zijn of een gewone gebruiker. Bij een correcte validatie wordt de gebruiker geauthentiseerd en de hoofdpagina geopend.

5.2.1.2 BPMN - Inloggen

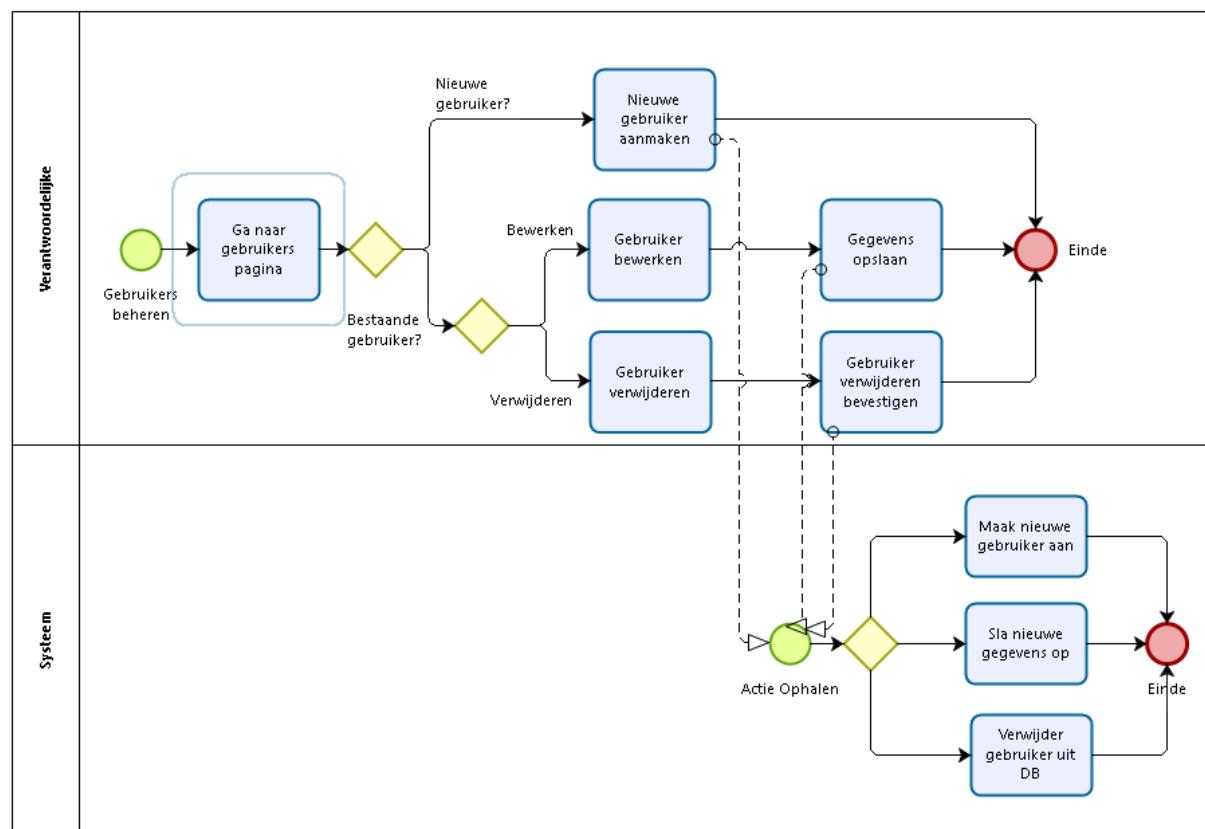


5.2.2 Gebruikers beheren

5.2.2.1 User story – Gebruikers beheren

Hoofddoel	<ul style="list-style-type: none"> Als verantwoordelijke kan ik nieuwe gebruikers aanmaken en bestaande gebruikers bewerken en verwijderen.
Bijkomende vereisten	<ul style="list-style-type: none"> Er zijn twee gebruikers niveaus, “verantwoordelijke” die gebruikers kan beheren en instellingen doen in het systeem. Daarnaast is er een gewone gebruiker die gebruik kan maken van de planningstool. Om het per ongeluk verwijderen van een gebruiker tegen te gaan wordt er eerst een bevestiging gevraagd.

5.2.2.2 BPMN – Gebruikers beheren

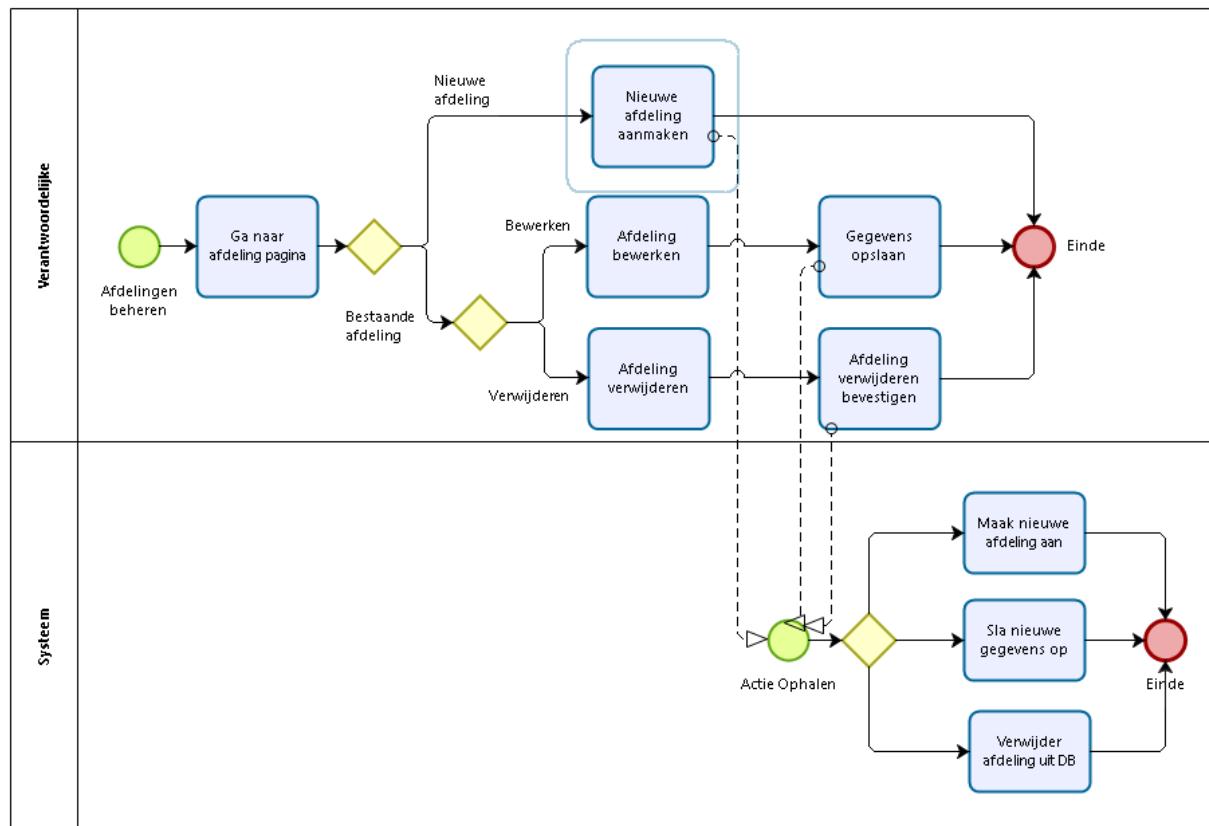


5.2.3 Afdelingen beheren

5.2.3.1 User story – Afdelingen beheren

Hoofddoel	<ul style="list-style-type: none"> Als verantwoordelijke kan ik nieuwe afdelingen aanmaken en bestaande afdelingen bewerken en verwijderen.
Bijkomende vereisten	<ul style="list-style-type: none"> Om het per ongeluk verwijderen van een departement tegen te gaan wordt er eerst een bevestiging gevraagd. Indien er nog een machine aan de afdeling gekoppeld is, kan deze niet verwijderd worden

5.2.3.2 BPMN – Afdelingen beheren

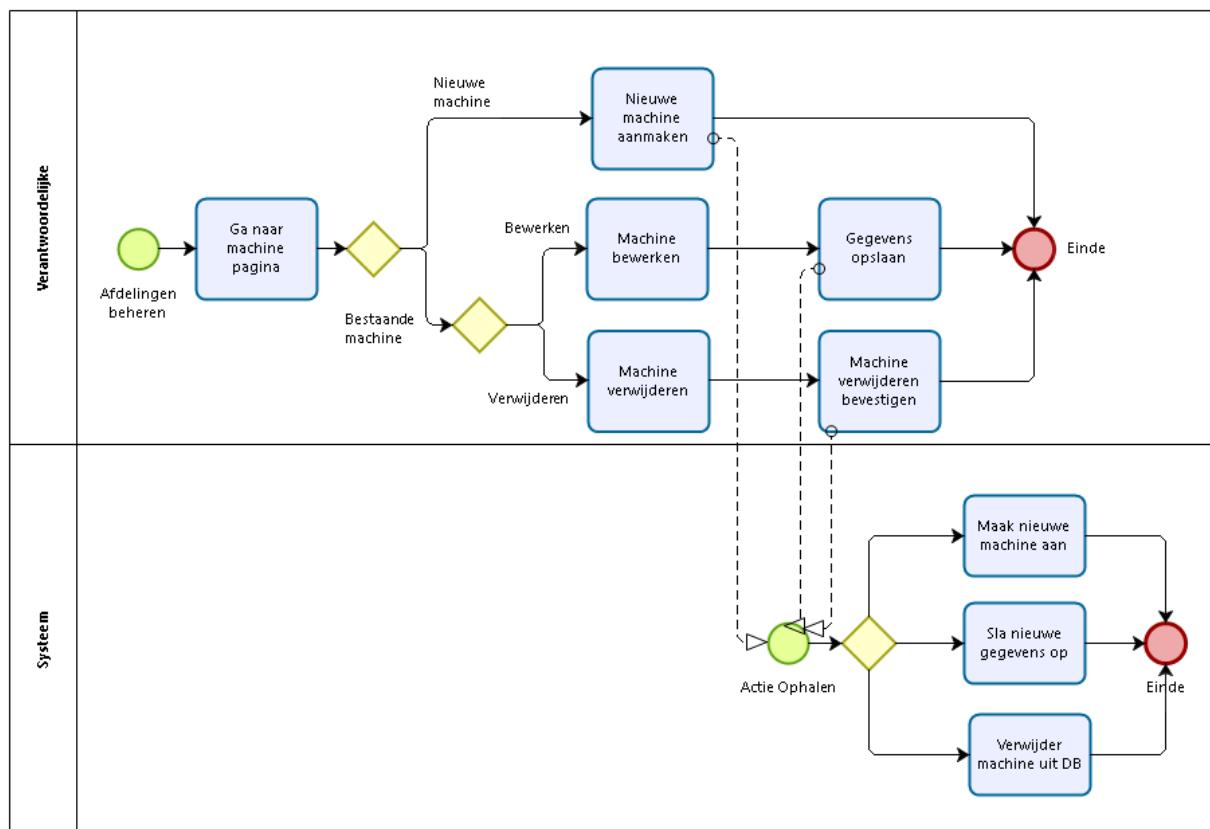


5.2.4 Machines beheren

5.2.4.1 User story – Machines beheren

Hoofddoel	<ul style="list-style-type: none"> Als verantwoordelijke kan ik nieuwe machines aanmaken en bestaande machines bewerken en verwijderen.
Bijkomende vereisten	<ul style="list-style-type: none"> Om het per ongeluk verwijderen van een machine tegen te gaan wordt er eerst een bevestiging gevraagd. Bij het aanmaken van een nieuwe machine moet deze aan een bestaande afdeling gekoppeld worden

5.2.4.2 BPMN – Machines beheren

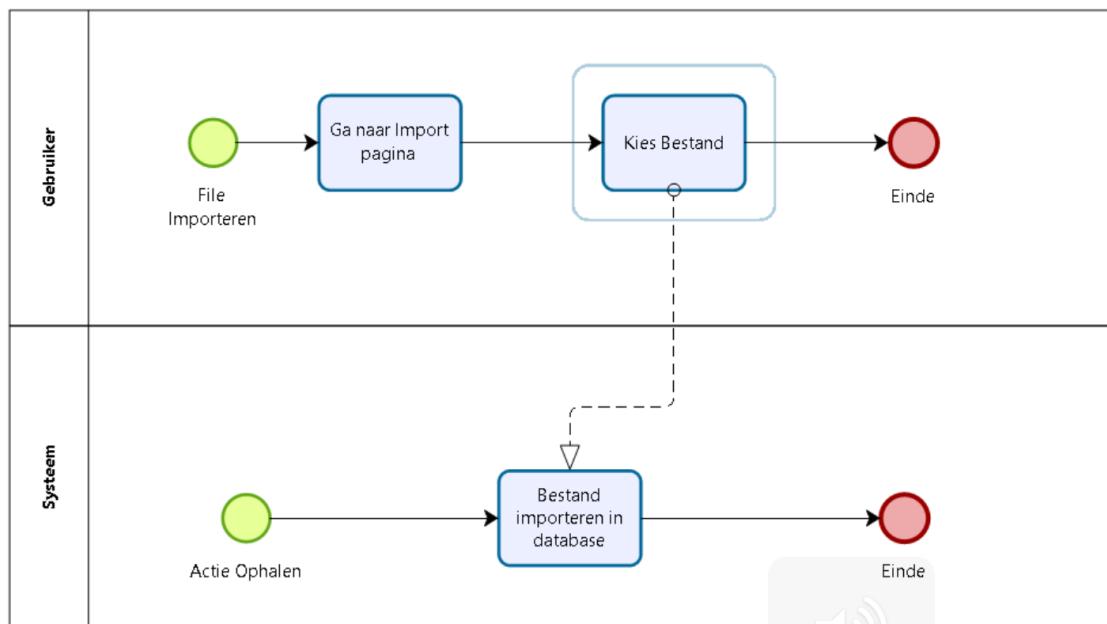


5.2.5 File Importeren

5.2.5.1 User Story – File importeren

Hoofddoel	<ul style="list-style-type: none">Als verantwoordelijke kan ik een bestand met werkgegevens inlezen.
Bijkomende vereisten	<ul style="list-style-type: none">Het bestand is een csv-file afkomstig uit FileMaker Pro.De gegevens worden naar de database gestuurd naar meerdere tabellen

5.2.5.2 BPMN – File importeren

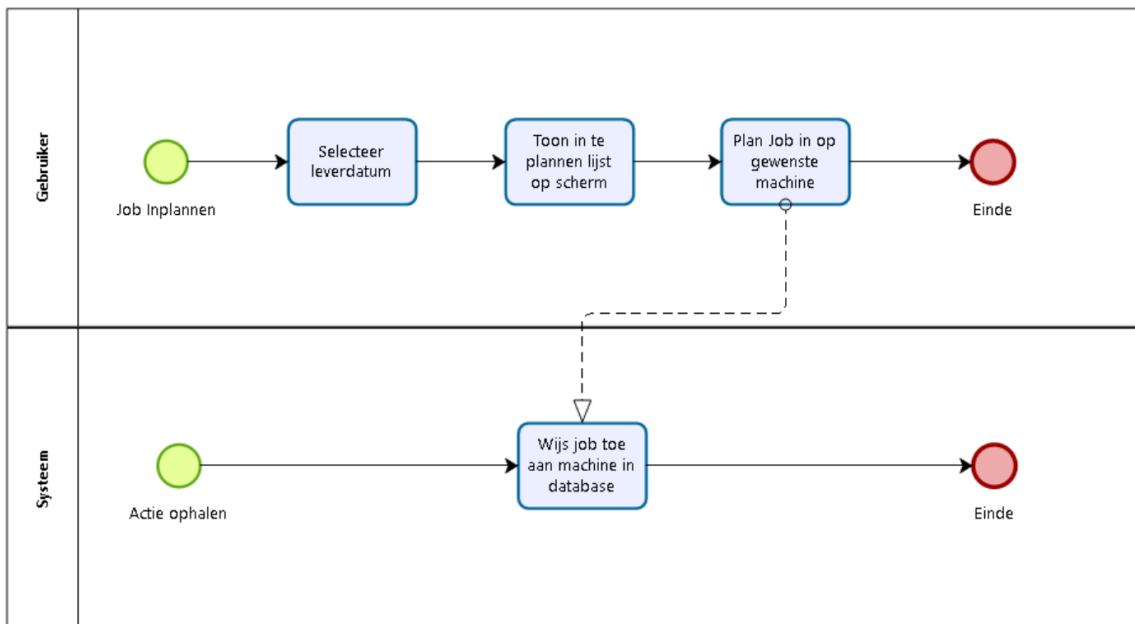


5.2.6 Job inplannen

5.2.6.1 User story – Job Inplannen

Hoofddoel	<ul style="list-style-type: none">Als verantwoordelijke ik een job inplannen.
Bijkomende vereisten	<ul style="list-style-type: none">Juiste leverdatum moet geselecteerd zijn

5.2.6.2 BPMN – Job inplannen

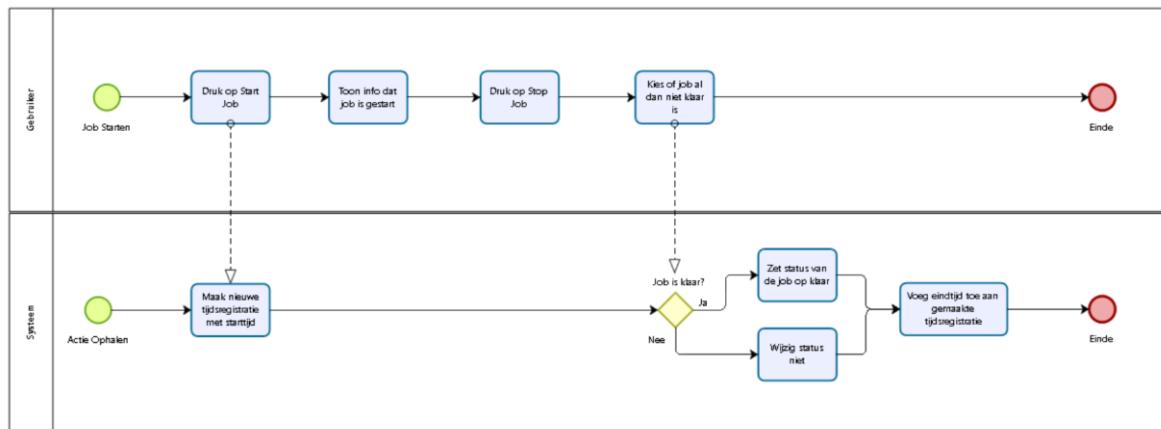


5.2.7 Job Starten/Stoppen

5.2.7.1 User story – Job Starten/Stoppen

Hoofddoel	<ul style="list-style-type: none"> Als gebruiker kan ik de tijdsregistratie van een drukwerk starten
Bijkomende vereisten	<ul style="list-style-type: none"> Job bestaat en is nog niet afgewerkt. Lijst met afgewerkte jobs wordt getoond op scherm

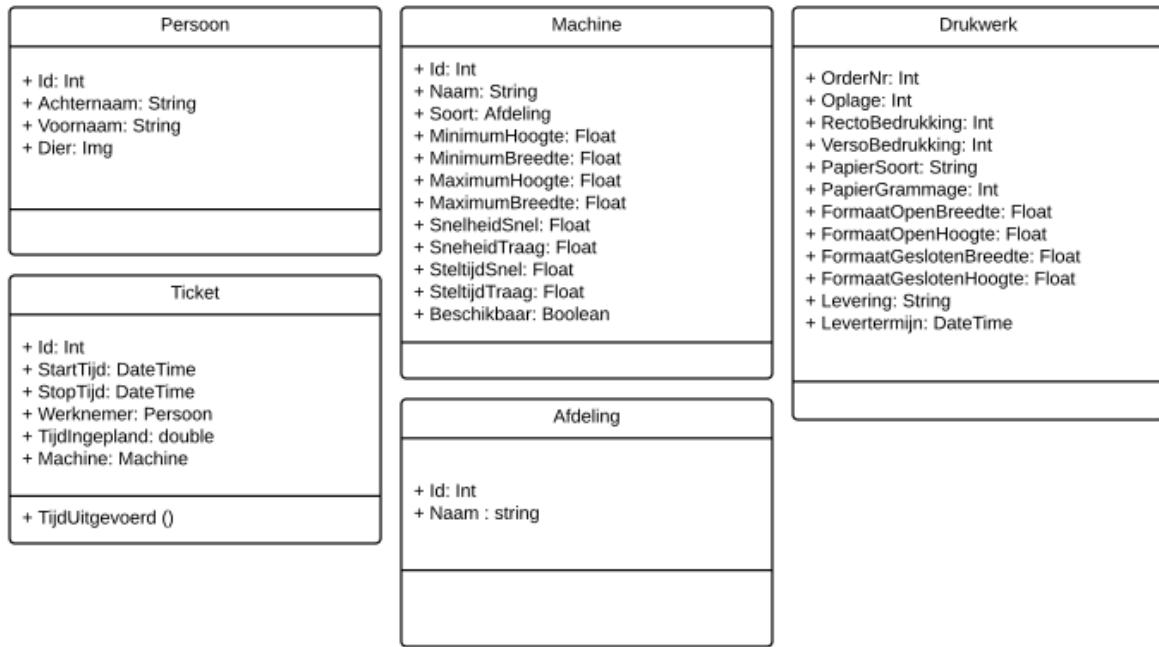
5.2.7.2 BPMN – Job Starten/Stoppen



5.3 UML klassendiagramma

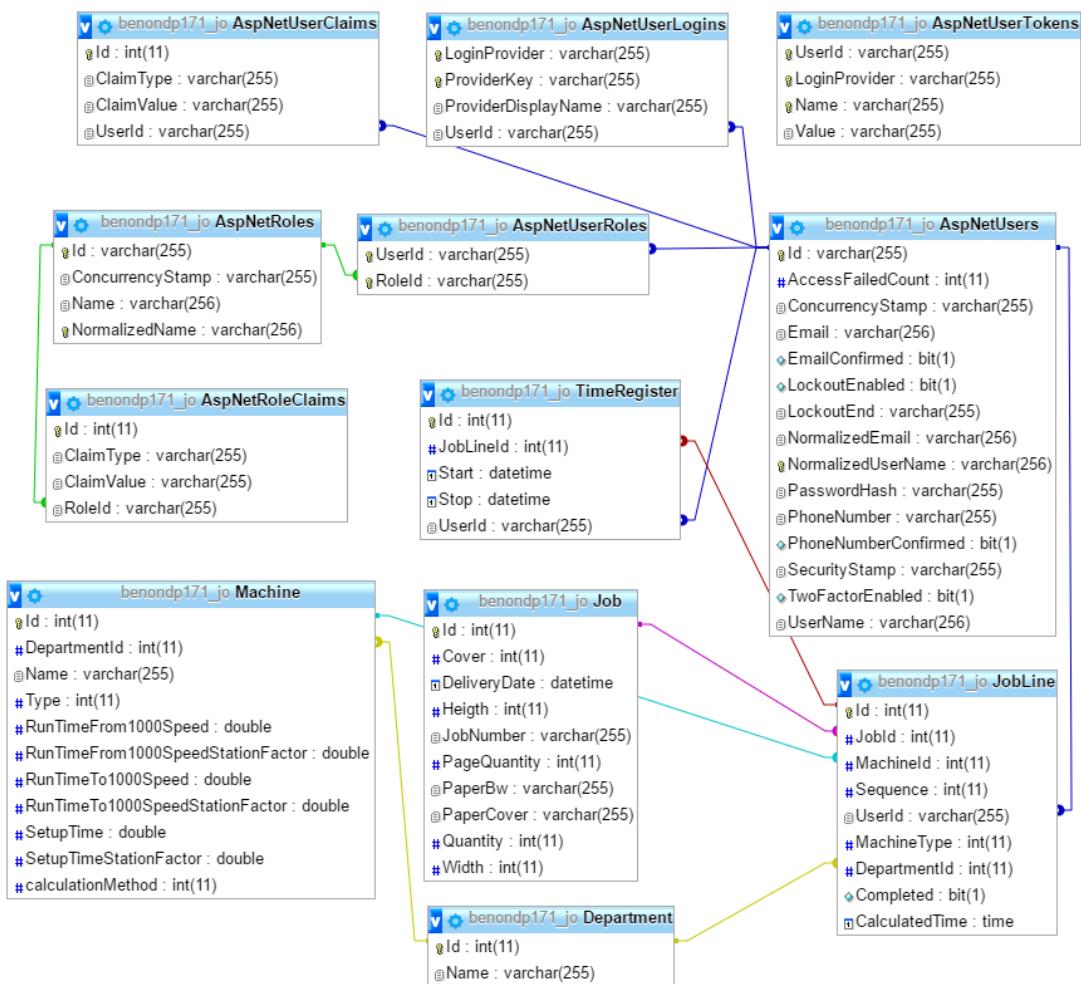
Bij het uitwerken van het UML klassendiagramma waren we gestart met het noteren van onze hoofd entiteiten. Jammer genoeg zijn we door alle drukte door vergeten hier mee verder te gaan. Door met het project bezig te zijn, werd dit als bijna automatisch meer uitgewerkt. Als we hier op terugblikken is dit zeker iets waar de in de toekomst anders mee zouden omgaan. Door het goed uitwerken van een klassendiagramma was er meer duidelijkheid geweest, wat ook weer positieve voordelen gehad zou hebben op gebied van tijd.

We geven ons origineel klassendiagramma wel graag mee, ten volste beseffend dat dit niet voldoende is.



Een klassendiagram vanuit het project zelf, op basis van onze code, hebben we jammer genoeg niet kunnen genereren. Dit omdat de volledige functionaliteit voor klasse generatie in .net Core niet goed werkt. We meerdere pogingen ondernomen om hier een oplossing voor te vinden, maar kwamen telkens weer problemen tegen. Aangezien er nog steeds gewerkt wordt aan het .net Framework hopen we dit op een later tijdstip nog te kunnen genereren.

5.4 Gegevensmodel ERD



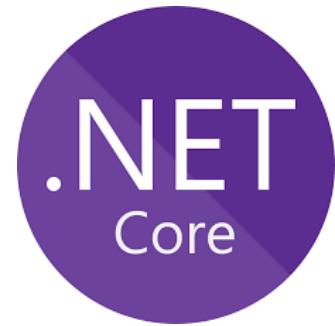
6 Proof of concept

Een hele periode van ons project werd gespendeerd aan een “Proof of Concept”. Deze proof of concept was nodig om na te gaan of de geselecteerde technologieën goed samenwerken. Deze technologieën werden geselecteerd op basis van de huidige werking van de klant.

De klant heeft een server draaien met daarop een MySQL database. Momenteel draaien alle servers een linux besturingssysteem, iets wat de klant graag wou behouden. Op basis van deze criteria en de gezien leerstof hebben wij onze volgende technologieën gekozen.

Omdat C# goed bij ons lessenpakket aansloot, hebben we gekozen voor .NET Core. Deze versie van .NET draait ook onder Linux. Als database kozen we voor een MySQL database.

Het was een hele uitdaging om alle gekozen dependencies goed onderling met elkaar te laten samenwerken. De extra moeilijkheid was dat er nog geen officiële MySQL adapter was toen wij ons project begonnen zijn.

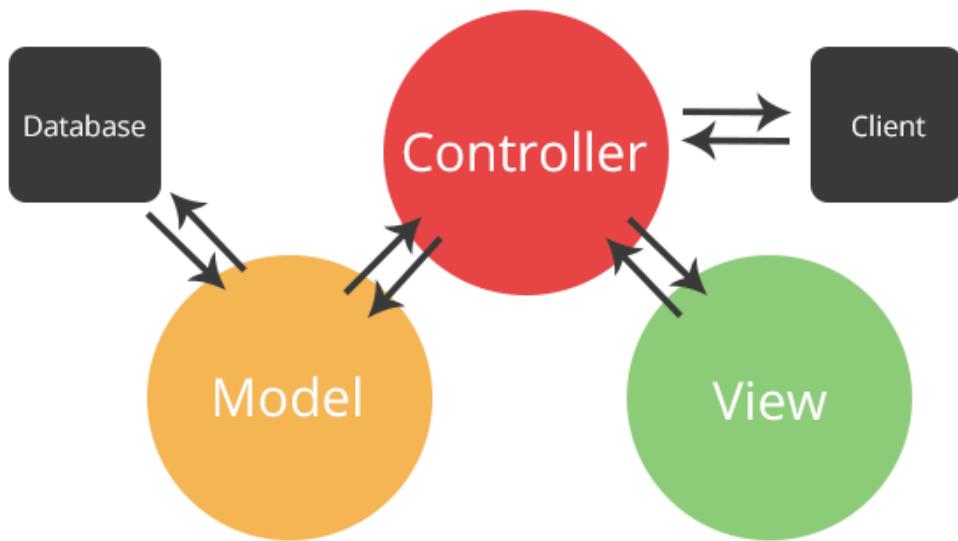


7 Technologische concepten

7.1 MVC

7.1.1 Introductie

MVC staat voor Model-View-Controller. Dit design pattern deelt een applicatie op in drie onderling verbonden delen. Door de opdeling in deze drie componenten zal je een applicatie krijgen die makkelijker te onderhouden is. Uiteindelijk zal dit ervoor zorgen dat de code ook makkelijker opnieuw gebruikt kan worden. Ook zullen meerdere mensen makkelijker samen kunnen werken aan de applicatie, aangezien iedereen duidelijk overzicht heeft van de structuur.



7.1.2 Model

Het model behandeld alle data, logica en bewerkingen. Ook is het model de component die communiceert met de database.

7.1.3 View

De View is eigenlijk de presentatie laag. Deze ontvangt de nodige gegevens van de Controller en toont de weergave aan de user.

7.1.4 Controller

De controller is de schakel tussen het model en de view. Deze geeft de nodige informatie aan de view, handelt interacties af en werkt het model bij. Best practise is dat de View en het Model nooit rechtstreeks met elkaar communiceren.

7.1.5 Voordelen

- Developers kunnen gemakkelijk gelijktijdig aan de applicatie werken
- Duidelijke structuur
- Herbruikbaar
- Gemakkelijk aanpasbaar
- Een model kan meerdere views hebben

7.1.6 Nadelen

- Voor zeer grote applicaties kan deze structuur moeilijk te onderhouden worden en moeten er extra lagen woren toegevoegd om te vermijden dat de controllers gigantische cluwes van code worden

7.1.7 MVC in .net Core

Een voorbeeld uit onze code over hoe wij MVC hebben toegepast. In dit specifiek deel behandelen we de departement entiteit (model, controller en view).

7.1.7.1 Model

Het model is de representative van de table Department in code. Door code first te werken hebben wij eerst het model aangemaakt wat later naar de database werd gemapt via Entity framework.

```
namespace ZwartOpWit.Models
{
    public class Department
    {
        public Department()
        {
            Machines = new List<Machine>();
        }

        public int Id { get; set; }

        [Required]
        [StringLength(255, MinimumLength = 3)]
        public string Name { get; set; }

        public ICollection<Machine> Machines { get; set; }
    }
}
```

MVC – voorbeeld Department model

7.1.7.2 Controller

In de DepartmentController maken we gebruik van het Model “Department” de data die we willen weergeven in de view klaar te zetten. In dit geval de lijst van departementen waarop een filter en een rangschikking kunnen staan. Deze data wordt dan met behulp van het viewModel “DepartmentListVM” doorgegeven aan de view.

```

[Authorize(Policy = "Admin")]
public class DepartmentController: Controller
{
    private readonly AppDbContext _context;
    const int PageSize = 3;

    public DepartmentController(AppDbContext context)
    {
        _context = context;
    }

    [HttpGet]
    public async Task<IActionResult> Index( string sortOrder,
                                            string currentFilter,
                                            string searchString,
                                            int? page)
    {
        DepartmentListVM departmentListVM = new DepartmentListVM();
        departmentListVM.currentSort = sortOrder;
        departmentListVM.currentFilter = searchString;
        departmentListVM.nameSortParm = String.IsNullOrEmpty(sortOrder) ? "name_desc" : "";
        |
        if (searchString != currentFilter)
        {
            page = 1;
        }

        var departments = _context.Departments.AsQueryable();

        if (!String.IsNullOrEmpty(searchString))
        {
            departments = departments.Where(s => s.Name.Contains(searchString));
        }

        switch (sortOrder)
        {
            case "name_desc":
                departments = departments.OrderByDescending(u => u.Name);
                break;
            default:
                departments = departments.OrderBy(u => u.Name);
                break;
        }

        departmentListVM.departmentList = await PaginatedList<Department>.CreateAsync(departments.AsNoTracking(), page ?? 1, PageSize);

        return View(departmentListVM);
    }
}

```

MVC – Voorbeeld Department Controller

7.1.7.3 View

In het view wordt de data uit het.viewmodel gebruikt om de pagina op te bouwen. De view zal nooit rechtstreeks met het Model (Database) communiceren. Het zal altijd langs een controller gaan. Als best practice wordt alle data die tussen controller en view gestuurd wordt in een view model geplaatst.

```

using Microsoft.AspNetCore.Mvc.Localization
@inject IViewLocalizer Localizer
@model ZwartOpWit.Models.ViewModels.DepartmentListVM
{
    var prevDisabled = !Model.departmentList.HasPreviousPage ? "disabled" : "";
    var nextDisabled = !Model.departmentList.HasNextPage ? "disabled" : "";
}
<div>
    <div>
        <h2>@Localizer["Title"]</h2>
        <p><a href="#" asp-controller="Department" asp-action="Create" class="btn btn-info">@Localizer["New"]</a></p>
    </div>
    <form asp-action="Index" method="get">
        <div class="form-actions no-color">
            <p>
                <input type="text" name="SearchString" value="@Model.currentFilter" />
                <input type="submit" value="@Localizer["Search"]" class="btn btn-default" /> |
                <a href="#" asp-action="Index" class="btn">@Localizer["BackToFullList"]</a>
            </p>
        </div>
    </form>
    <div>
        <table class="table table-striped">
            <thead>
                <tr>
                    <th><a href="#" asp-action="Index" asp-route-sortOrder="@Model.nameSortParm" asp-route-currentFilter="@Model.currentFilter">@Localizer["DepartmentName"]</a></th>
                    <th>@Localizer["Action"]</th>
                </tr>
            </thead>
            <tbody>
                <tr>
                    <td>@d.Name</td>
                    <td>
                        <a href="#" asp-controller="Department" asp-action="Read" asp-route-id="@d.Id" class="btn btn-default"><span class="glyphicon glyphicon-eye-open"></span></a>
                        <a href="#" asp-controller="Department" asp-action="Update" asp-route-id="@d.Id" class="btn btn-default"><span class="glyphicon glyphicon-pencil"></span></a>
                        <a href="#" asp-controller="Department" asp-action="Delete" asp-route-id="@d.Id" class="btn btn-default"><span class="glyphicon glyphicon-trash"></span></a>
                    </td>
                </tr>
            </tbody>
        </table>
        <a href="#" asp-action="Index"
           asp-route-sortorder="@viewData["CurrentSort]"
           asp-route-page="@Model.departmentList.PageIndex - 1"
           asp-route-currentfilter="@viewData["CurrentFilter"]"
           class="btn btn-default @prevDisabled"
           href="#">@Localizer["Previous"]</a>
        <a href="#" asp-action="Index"
           asp-route-sortorder="@viewData["CurrentSort]"
           asp-route-page="@Model.departmentList.PageIndex + 1"
           asp-route-currentfilter="@viewData["CurrentFilter"]"
           class="btn btn-default @nextDisabled"
           href="#">@Localizer["Next"]</a>
    </div>

```

MVC – Voorbeeld Department View

7.2 Git

7.2.1 Introductie

Git is een Source Control systeem voor het beheren van code. Dit maakt het makkelijk om de broncode van een programma op een gedeelde plaats te hebben, zodat wijzigen over alle kopieën automatisch meegedeeld worden.

7.2.2 Basishandelingen

7.2.2.1 Commit

Git bekijkt welke wijzigingen er zijn gebeurd t.o.v. de lokale repository en geeft deze weer in een lijst. Als je deze commit dan door voert worden deze opgenomen in de lokale repository.

Belangrijk is wel dat je aanduidt om “alle files” mee te nemen, aangezien nieuwe files niet standaard mee ingecheckt worden.

7.2.2.2 Push

Een push volgt na een commit. Alle commits die gebeurd zijn sturen we dan ook door naar onze remote repository, zodat deze ook zichtbaar worden voor andere medewerkers.

Als jou lokale versie achter loopt op de versie van de remote, zal je een foutmelding krijgen. Daarom is het belangrijk om eerst altijd te pullen en daarna de pushen.

7.2.2.3 Pull

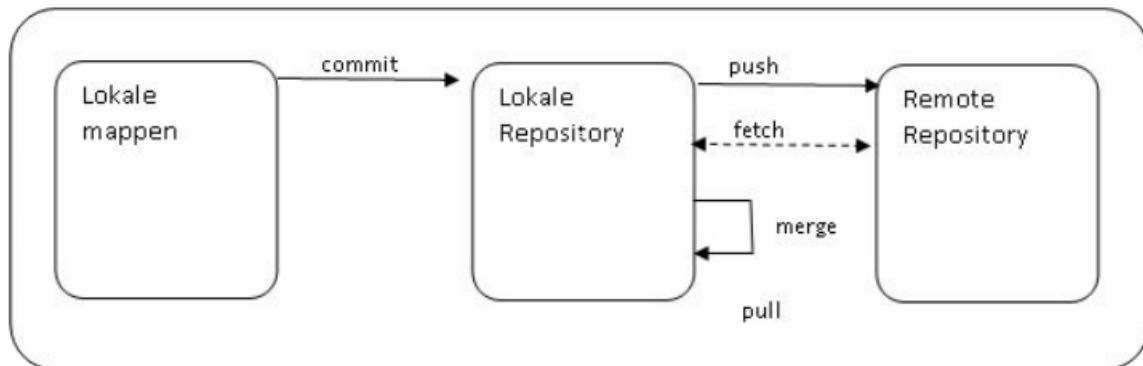
Bij een Pull gaat Git kijken of er wijzigingen staan op de remote repository. Indien je hier mee verder gaat, worden de wijzigingen zowel op de bestanden zelf als op de lokale repository toegepast.

7.2.2.4 Fetch

Bij een fetch ga je enkel de wijzigingen ophalen van de remote repository en deze toepassen op de lokale repository. In tegenstelling tot een pull worden de bestanden niet gewijzigd.

7.2.2.5 Merge

Een merge is een manier om manueel wijzigingen door te voeren aan de repository of bestanden. Het maakt het mogelijk om twee branches samen te voegen.



7.2.3 Branching

Bij branching maak je een aftakking van de repository aan. Deze aftakking krijgt alle wijzigingen mee die voorheen zijn gebeurd op de hoofdtak. Eenmaal afgetakt is het aan de user zelf om deze up to date te houden. Je tak blijft naast de hoofdtak bestaan. Je kan deze ook op de remote repository zetten, zodat deze zichtbaar is voor anderen. Dit is makkelijk voor bijvoorbeeld code reads.

Desondanks dat het systeem van branching zeer handig is, hebben we besloten om enkel met een "master" te werken. Aangezien ons team uit slechts twee personen bestaat, was dit voldoende.

7.3 Bootstrap

Bootstrap is een open source front-end framework voor het designen van websites en webapplicaties. Het bevat HTML en CSS templates voor zowel tekts, knoppen, navigatie, andere componenten en optioneel zelfs javascript extensies.

Bootstrap maakt het de front-end developer gemakkelijk om snel een mooi resultaat te krijgen.

Wij hebben gekozen om met bootstrap te werken voor het snelle maar toch ordelijke visuele resultaat. Daarnaast maakt bootstrap het gemakkelijk om responsive te werken. Momenteel wordt er bij zwart op wit enkel gebruik gemaakt van een desktops en laptops. Wil het bedrijf later overschakelen op tablets of andere, vormt dit geen enkel probleem. De applicatie is er al helemaal klaar voor.

7.4 Rendering engine (razor)

Razor wordt gebruikt voor het creëren van dynamische webpagina's. Razor is een syntax die gebruikt kan worden om een html pagina in een model te gieten. De razor syntax maakt gebruik van het "@" teken en vereist geen sluitingstag.

De syntax van Razor maakt de html pagina's korter en zorgt er voor dat de code compacter wordt. Met andere woorden, meer functionaliteit, door minder code.

7.5 Identity

7.5.1 Introductie

Doordat het project nood heeft aan identificatie van gebruikers en deze verschillende rollen hebben is er nood aan een autorisatie en authenticatie framework. Hier hebben we geopteerd voor een implementatie van het .net Core Identity Framework.

Gebruikers krijgen een account die ze kunnen gebruiken om in te loggen met een username en paswoord. Het identity framework biedt ook de mogelijkheid om te werken met een externe authenticatie provider zoals Facebook, Google, Twitter,... Deze hebben we voor ons project niet toegepast aangezien het om een interne applicatie gaat.

Om gebruik te kunnen maken van "Identity" moet er eerst het Nuget Package "Microsoft.AspNetCore.Identity.EntityFrameworkCore" geïnstalleerd worden.

7.5.2 Identity activeren

Om Identity te activeren moet deze als een service toegevoegd worden aan de startup.cs klasse (Identity, Screenshot 1). Deze klasse zorgt er voor dat deze service door middel van dependency injection beschikbaar wordt in de applicatie.

```
// This method gets called by the runtime. Use this method to add services
public void ConfigureServices(IServiceCollection services)
{
    // Add framework services.
    services.AddEntityFramework()
        .AddSqlServer()
        .AddDbContext<ApplicationContext>(options =>
            options.UseSqlServer(Configuration["Data:DefaultConnection:Con

    services.AddIdentity<ApplicationUser, IdentityRole>()
        .AddEntityFrameworkStores<ApplicationContext>()
        .AddDefaultTokenProviders();

    services.AddMvc();

    // Add application services.
    services.AddTransient<IEmailSender, AuthMessageSender>();
    services.AddTransient<ISmsSender, AuthMessageSender>();
}
```

Identity, Screenshot 1: Identity Service toevoegen aan startup.cs

Naast het activeren van de service moet ook de optie “Useldentity” in de “Configure” methode van de startup klasse worden toegevoegd. Dit zorgt tegelijk ook voor cookie-based authenticatie.

In de configure klassen kunnen ook andere configuraties worden toegevoegd zoals het aantal log in pogingen, het pad naar de loginpagina, het log out pad,... (Identity, screenshot 2)

```

services.Configure<IdentityOptions>(options =>
{
    // Password settings
    options.Password.RequireDigit = true;
    options.Password.RequiredLength = 8;
    options.Password.RequireNonAlphanumeric = false;
    options.Password.RequireUppercase = true;
    options.Password.RequireLowercase = false;

    // Lockout settings
    options.Lockout.DefaultLockoutTimeSpan = TimeSpan.FromMinutes(30);
    options.Lockout.MaxFailedAccessAttempts = 10;

    // Cookie settings
    options.Cookies.ApplicationCookie.ExpireTimeSpan = TimeSpan.FromDays(1);
    options.Cookies.ApplicationCookie.LoginPath = "/Account/LogIn";
    options.Cookies.ApplicationCookie.LogoutPath = "/Account/LogOut";

    // User settings
    options.User.RequireUniqueEmail = true;
});

// This method gets called by the runtime. Use this method to configure the
public void Configure(IApplicationBuilder app, IHostingEnvironment env, ILoggerFactory loggerFactory)
{
    loggerFactory.AddConsole(Configuration.GetSection("Logging"));
    loggerFactory.AddDebug();

    if (env.IsDevelopment())
    {
        app.UseBrowserLink();
    }
}

```

Identity, Screenshot 2: Toevoegen van configuraties

7.5.3 Registreren + Inloggen

Identity beschikt over een registratiepagina voor gebruikers. Deze gebruiken we niet omdat we willen dat een admin alle gebruikers aan maakt.

Een login wordt afgehandeld via de loginmethode in de accountController. Deze maakt achterliggend gebruik van de klasse SignInManager. Dit is een identity framework klasse die alle logica afhandelt in verband met inloggen. Aan deze methode wordt een e-mail en paswoord mee gegeven waarna deze bepaald of dit e-mail adres bestaat en dat dit daarvoor het juist paswoord is. Achterliggend wordt hier ook gewerkt met encryptie en salts (Identity, Screenshot 3).

```

[HttpPost]
[AllowAnonymous]
[ValidateAntiForgeryToken]
public async Task<IActionResult> Login(LoginViewModel model, string returnUrl = null)
{
    ViewData["ReturnUrl"] = returnUrl;
    if (ModelState.IsValid)
    {
        // This doesn't count login failures towards account lockout
        // To enable password failures to trigger account lockout, set lockoutOnFailure: true
        var result = await _signInManager.PasswordSignInAsync(model.Email, model.Password, model.RememberMe, lockoutOnFailure:
        if (result.Succeeded)
        {
            _logger.LogInformation(1, "User logged in.");
            return RedirectToLocal(returnUrl);
        }
        if (result.RequiresTwoFactor)
        {
            return RedirectToAction(nameof(SendCode), new { ReturnUrl = returnUrl, RememberMe = model.RememberMe });
        }
        if (result.IsLockedOut)
        {
            _logger.LogWarning(2, "User account locked out.");
            return View("Lockout");
        }
        else
        {
            ModelState.AddModelError(string.Empty, "Invalid login attempt.");
            return View(model);
        }
    }

    // If we got this far, something failed, redisplay form
    return View(model);
}

```

Identity, Screenshot 3: Login

7.5.4 Uitloggen

Er bestaat natuurlijk ook de mogelijkheid om uit te loggen door op de Log Out knop te duwen. Deze roept de LogOut actie op in de Account controller. Deze verwijdert de gebruiker uit de cookie. Dit loopt ook weer via de SignInManager. (Identity, Screenshot 4)

```

[HttpPost]
[ValidateAntiForgeryToken]
public async Task<IActionResult> LogOut()
{
    await _signInManager.SignOutAsync();
    _logger.LogInformation(4, "User logged out.");
    return RedirectToAction(nameof(HomeController.Index), "Home");
}

```

Identity, Screenshot 4: Logout

7.5.5 Uitbreidung userklasse

Het toevoegen van .net Core Identity, zorgt ook voor wijzigingen in het datamodel. Zo worden er verschillende tabellen toegevoegd waaronder de usertabel “AspNetUsers”. Deze tabel bevat alle users. Door onze eigen Userklasse hier van te laten overerven kunnen we extra velden aan deze tabel toevoegen. (Identity, Screenshot 5)

```
namespace ZwartOpWit.Models
{
    public class User : IdentityUser
    {
        public User()
        {
            TimeRegisterList = new List<TimeRegister>();
            JobLineList = new List<JobLine>();
        }

        public List<TimeRegister> TimeRegisterList { get; set; }
        public List<JobLine> JobLineList { get; set; }
    }
}
```

Identity, Screenshot 5: Eigen Userklasse erft over van IdentityUser

7.5.6 Rollen

In de zwart op wit applicatie wordt er gebruik gemaakt van 2 rollen. Een rol Employee (Werknemer) en een Admin rol. Deze zullen beide verschillende rechten hebben. De Admin rol zal dezelfde rechten hebben als de employee rol met daarboven op nog extra rechten om het systeem te configureren.

Elke gebruiker zal in het systeem maar 1 rol hebben en zal om het systeem te gebruiken zich moeten authentiseren via een login formulier.

7.5.6.1 Definiëren van de rollen

Om de rollen te definiëren maken we gebruik van een static class die uitgevoerd wordt in de “configure” methode van de klasse “startup.cs”. Deze methode gaat kijken of de rollen gedefinieerd in de lijst al bestaan. Bestaan ze nog niet worden ze aangemaakt. Op deze manier zorgen we ervoor dat de rollen altijd aangemaakt zijn ook al is de database leeg.

```

public static class RolesData
{
    private static readonly string[] Roles = new string[] { "Admin", "Employee" };

    1 reference | 0 changes | 0 authors, 0 changes
    public static async Task SeedRoles(IServiceProvider serviceProvider)
    {
        using (var serviceScope = serviceProvider.GetRequiredService<IServiceScopeFactory>().CreateScope())
        {
            var dbContext = serviceScope.ServiceProvider.GetService<AppDbContext>();

            if (dbContext.Database.GetPendingMigrations().Any())
            {
                await dbContext.Database.MigrateAsync();

                var roleManager = serviceProvider.GetRequiredService<RoleManager<IdentityRole>>();

                foreach (var role in Roles)
                {
                    if (!await roleManager.RoleExistsAsync(role))
                    {
                        await roleManager.CreateAsync(new IdentityRole(role));
                    }
                }
            }
        }
    }
}

```

Autorisatie – role seed klasse

```

0 references | dasniva, 4 hours ago | 2 authors, / changes
public void Configure(IApplicationBuilder app, IHostingEnvironment env, ILoggerFactory loggerFactory)
{

    loggerFactory.AddConsole(Configuration.GetSection("Logging"));
    loggerFactory.AddDebug();

    if (env.IsDevelopment())
    {
        app.UseDeveloperExceptionPage();
        app.UseBrowserLink();
    }
    else
    {
        app.UseExceptionHandler("/Home/Error");
    }

    //Translations
    app.UseRequestLocalization(new RequestLocalizationOptions
    {
        DefaultRequestCulture = new RequestCulture(defaultculture),
        // Formatting numbers, dates, etc.
        SupportedCultures = supportedCultures,
        // UI strings that we have localized.
        SupportedUICultures = supportedCultures
    });

    //Roles
    RolesData.SeedRoles(app.ApplicationServices).Wait();
}

```

Autorisatie – role seed

7.5.6.2 Definiëren van Policies

Policies zijn verzamelingen van 1 of meer rollen. Door policies uit te werken worden rollen losgekoppeld van het autorisatie systeem. Een policy krijgt de rechten en aan een policy kunnen rollen worden toegevoegd.

Policies worden aangemaakt in de “configureService” methode van de klasse “startup.cs”. Deze policies worden later gebruikt in de view om te kijken of delen gerendered of niet gerenderd moeten worden.

```
services.AddAuthorization(options =>
{
    options.AddPolicy("RequireAdminRole", policy => policy.RequireRole("Admin"));
});

services.AddAuthorization(options =>
{
    options.AddPolicy("RequireEmployeeRole", policy => policy.RequireRole("Employee"));
});
```

Autorisatie – policies declareren

7.5.6.3 Toepassen van autorisatie

Autorisatie binnen het identity framework kan worden afgehandeld in zowel de view als de controller. Voor de ervaring van de gebruiker is het beste om van beide gebruik te maken zodat de gebruiker visueel enkel ziet waar de gebruiker toegang toe heeft.

7.5.6.3.1 Controller autorisatie

In een controller kan autorisatie op 2 niveaus afgedwongen worden. Op controller niveau of op methode niveau. Voor beide gebeurt dit met dezelfde annotatie. Voor het zwart op wit project doen wij dit enkel op controller niveau. Daarnaast maken we hier ook gebruik van policies i.p.v. rollen.

```
namespace ZwartOpWit.Controllers
{
    [Authorize(Policy = "Admin")]
    public class DepartmentController : Controller
    {
        private readonly AppDbContext _context;
```

Autorisatie – Controller security

7.5.6.3.2 Autorisatie in views

In views kan ook gekeken worden welke rollen en policies er aan de huidige gebruiker heeft. Dit maakt het mogelijk op bepaalde delen van de view juist wel of juist niet te renderen.

```

1  @using Microsoft.AspNetCore.Authorization
2  @using Microsoft.AspNetCore.Mvc.Localization
3  @inject IAuthorizationService AuthorizationService
4  @inject IViewLocalizer Localizer
5
6  @if (await AuthorizationService.AuthorizeAsync(User, "RequireAdminRole"))
7  {
8      <div class="navbar navbar-inverse navbar-default">
9          <div class="container">
10             <div class="navbar-header">
11                 <button type="button" class="navbar-toggle" data-toggle="collapse" data-target="#navbar-secondary">
12                     <span class="sr-only">Toggle navigation</span>
13                     <span class="icon-bar"></span>
14                     <span class="icon-bar"></span>
15                     <span class="icon-bar"></span>
16                 </button>
17             </div>
18             <div class="navbar-collapse collapse" id="navbar-secondary">
19                 <ul class="nav navbar-nav admin-navbar">
20                     <li><a asp-area="" asp-controller="User" asp-action="Index">@Localizer["Users"]</a></li>
21                     <li><a asp-area="" asp-controller="Machine" asp-action="Index">@Localizer["Machines"]</a></li>
22                     <li><a asp-area="" asp-controller="Department" asp-action="Index">@Localizer["Departments"]</a></li>
23                     <li><a asp-area="" asp-controller="TimeRegister" asp-action="Index">@Localizer["TimeRegistrations"]</a></li>
24                 </ul>
25             </div>
26         </div>
27     </div>
28 }

```

Autorisatie – view rendering

7.5.7 Gebruikers beheren

Omdat gebruikers en rollen onder het Identity framework vallen is het niet mogelijk om deze op dezelfde manier te beheren als entiteiten die we zelf hebben aangemaakt. Het identity framework heeft hiervoor een aantal helper classes die gebruikt moeten worden. Deze classes zijn via dependency injection beschikbaar in de controllers.

```

namespace ZwartOpWit.Controllers
{
    public class UserController : Controller
    {
        private readonly AppDbContext _context;
        private readonly UserManager<User> _userManager;
        private readonly RoleManager<IdentityRole> _roleManager;
        private readonly ILogger _logger;
        const int PageSize = 3;

        public UserController(AppDbContext context,
                             UserManager<User> userManager,
                             RoleManager<IdentityRole> roleManager,
                             ILoggerFactory loggerFactory)
        {
            context = context;
            _userManager = userManager;
            _logger = loggerFactory.CreateLogger<UserController>();
            _roleManager = roleManager;
        }
    }
}

```

Gebruikers beheren – dependency injection user en rolemanager

7.5.7.1 UserManager

De usermanager klasse wordt gebruikt om alle handelingen met gebruikers af te handelen zoals het aanmaken, wijzigen en verwijderen van gebruikers. Maar ook kijken of de gebruik al bestaat.

7.5.7.2 RoleManager

De rolemanager klasse wordt gebruikt om alle handelingen met rollen af te handelen zoals kijken welke user heeft welke rol maar ook het aanmaken, wijzigen en verwijderen van rollen.

7.5.8 Forgot password

7.5.8.1 Mail service

7.5.8.1.1 Introductie

Om e-mails te kunnen sturen met onze applicatie hebben we gekozen om gebruik te maken van een externe service. Dit wordt aangeraden vanuit de best practises van Microsoft. Wij hebben gekozen van om gebruik te maken van SendGrid. SendGrid is een mail services die alle mailing afhandelt. Deze service zorgt ervoor dat je mail correct aankomen en bijvoorbeeld niet in de spam filter terecht komen. Ook geeft een mail service inzicht over wat er met de mails gebeurt die via jou account gestuurd worden

7.5.8.1.2 Mail Account configureren

De sendgrid mail account verwacht een username (SendGridUser) en wachtwoord (SendGridKey) deze gaan we instellen door gebruik te maken van het de secret manager store in windows.

Gegevens worden met het volgende command toegevoegd aan de secret manager store: “dotnet user-secrets set SendGridUser MySendGridUserName”

Via het option pattern worden deze dan toegevoegd aan de startup.cs klasse. Hiervoor maken we gebruik van de klasse AuthMessageServiceOption die onze SendGridUser en SendGridKey meegeeft aan de startup.cs klasse

```
namespace ZwartOpWit.Models.Services
{
    public class AuthMessageSenderOptions
    {
        public string SendGridUser { get; set; }
        public string SendGridKey { get; set; }
    }
}
```

Forgot password – Options class

```
// This method gets called by the runtime. Use this method to add services to the container.
public void ConfigureServices(IServiceCollection services)
{
    // Add framework services.
    services.AddMvc();

    //Connection string ophalen (zie appsettings.json)
    var sqlConnectionString = Configuration.GetConnectionString("ConnectionString");

    //Add entity framework service
    services.AddEntityFramework().AddDbContext<AppDBContext>(options =>
        options.UseMySQL(
            sqlConnectionString,
            b => b.MigrationsAssembly("ZwartOpWit")
        )
    );

    //Add identity service
    services.AddIdentity<User, IdentityRole>(config =>
    {
        config.SignIn.RequireConfirmedEmail = true;
    })
    .AddEntityFrameworkStores<AppDBContext>()
    .AddDefaultTokenProviders();

    // Add application services.
    services.AddTransient<IEmailSender, AuthMessageSender>();
    services.AddTransient<ISmsSender, AuthMessageSender>();

    services.Configure<AuthMessageSenderOptions>(Configuration);
}
```

Forgot password - configuration

7.5.8.2 *Forgot password form*

Het “Forgot password” form vraagt het email adres van de gebruiker en stuurt dan een link met een activatie code naar het email van de gebruiker mocht het e-mail adres gekend zijn. De gebruiker kan deze link dan gebruiken om het wachtwoord opnieuw in te stellen.

7.6 Globalisation and localization

7.6.1 Introduction

Globlisation en localization is een belangrijk gegeven als je je project wil kunnen gebruiken in verschillende talen en omgevingen. Door hier rekening mee te houden in je programma kan zonder wijzingen in je code gemakkelijk afstemmen op de noden van de (toekomstige) gebruiker.

Globalisatie is een proces waarbij je applicaties maakt die verschillende culturen ondersteunen. Bij lokalisatie ga je rekening houden met een bepaalde cultuur, bijvoorbeeld een taal.

7.6.2 Gebruik in ons project

Om globalisation en localisation te activeren moeten er verschillende stappen ondernomen worden. Standaard bevat .net Core hier functionaliteit voor, maar deze moet wel geconfigureerd worden.

7.6.2.1 *Stap 1: Toevoegen van de service*

De eerste stap is het toevoegen van de configuratie voor de services in de startup.cs klasse. Hier moet aan de addMVC methode ook een methode addViewLocalization meegegeven worden met daarop de methode addDataAnnotationLocalization (Globalisation en Localization, Screenshot 1).

```
//Translation services
services
    .AddLocalization(options => options.ResourcesPath = "Resources")
    .AddMvc()
    .AddViewLocalization()
    .AddDataAnnotationsLocalization();
```

Globalisation en Localization, Screenshot 1: Toevoegen van de service

AddLocalization voegt de localization service toe aan de service container.

AddViewLocalization voegt ondersteuning toe voor localized files (resources die eindigen op .nl.resx, .en.resx, .fr.resx, ...)

AddDataAnnotations geeft ondersteuning voor de localized DataAnnotations in de views.
(@Localizer["DepartmentName"])

Door deze methodes de implementeren heb je de volledige translations service geactiveerd.

7.6.2.2 *Stap 2: Het configureren van Localizations*

De localizations waarvan je gebruikt maakt moeten ook gedefinieerd worden. Dit doe je in de startup.cs klasse in de methode configure.

Hier gaan we een lijst zetten van default (localizations) (Globalisation en Localization, Screenshot 2). In ons geval Engels en Nederlands (Globalisation en Localization, Screenshot 3).

```
public static string defaultCulture = "nl-BE";

public static List<CultureInfo> supportedCultures
{
    get
    {
        return new List<CultureInfo> {
            new CultureInfo("nl-BE"),
            new CultureInfo("en-US")
        };
    }
}
```

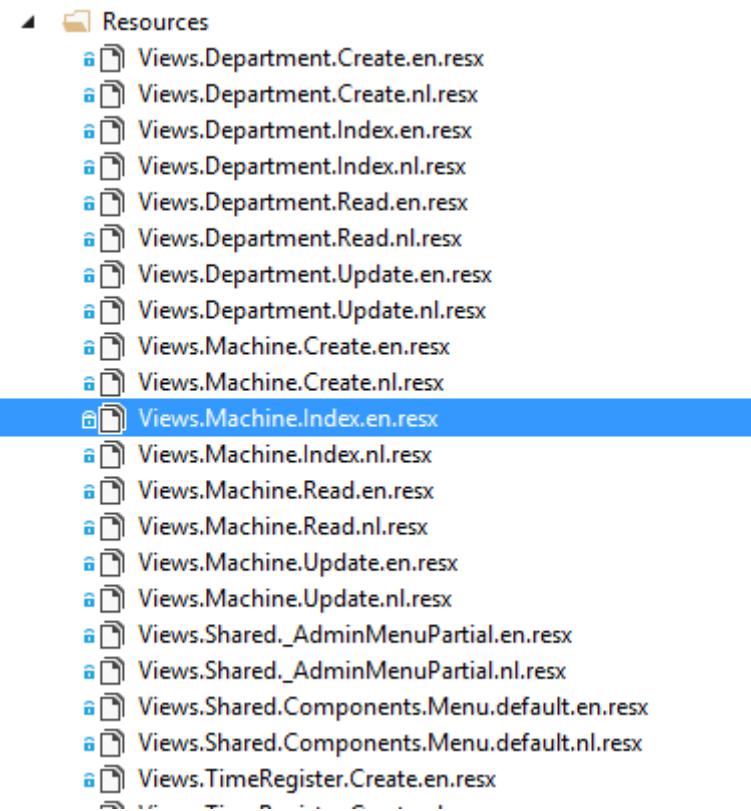
Globalisation en Localization, Screenshot 2: Het configureren van localizations

```
//Translations
app.UseRequestLocalization(new RequestLocalizationOptions
{
    DefaultRequestCulture = new RequestCulture(defaultCulture),
    // Formatting numbers, dates, etc.
    SupportedCultures = supportedCultures,
    // UI strings that we have localized.
    SupportedUIT Cultures = supportedCultures
});
```

Globalisation en Localization, Screenshot 3: Het configureren van localizations

7.6.2.3 Stap 3: Het aanmaken van resource files

De derde stap is het aanmaken van de resource files. Deze files worden aangemaakt per view per culture. De map waar deze staan hebben we eerder al geconfigureerd in stap 1. (Globalisation en Localization, Screenshot 4)



Globalisation en Localization, Screenshot 4: Resource mappen structuur

Een resource file wordt altijd aangemaakt met op het einde een culture info. In dat geval .nl, zo weet het systeem welke file bij welke taal hoort. Een resource file werkt op de basis van een key en een value. De key blijft in iedere culture hetzelfde, terwijl de value de eigenlijke vertaling is. (Globalisation en Localization, Screenshot 5)

Strings			
	Name	Value	Comment
▶	Add	Voeg toe	
	Back	Terug	
	Create	Aanmaken	
	DepartmentName	Departement naam	
*			

Globalisation en Localization, Screenshot 5: Resource file

7.6.2.4 Step 4: Select language, partial view

Omdat je natuurlijk eenvoudig van taal wil kunnen wisselen in het programma hebben we een partial view toegevoegd (Globalisation en Localisation, Screenshot 6). Deze gaat de culture info in een cookie opslaan (Globalisation en Localisation, Screenshot 7).

```

@using Microsoft.AspNetCore.Builder
@using Microsoft.AspNetCore.Http.Features
@using Microsoft.AspNetCore.Localization
@using Microsoft.AspNetCore.Mvc.Localization
@using Microsoft.Extensions.Options

@inject IViewLocalizer Localizer

@{
    var requestCulture = Context.Features.Get<IRequestCultureFeature>();
    var cultureItems = Startup.supportedCultures
        .Select(c => new SelectListItem { Value = c.Name, Text = c.DisplayName })
        .ToList();
}



- Language <span class="caret"></span>

@foreach (var cultureItem in cultureItems)
{
    <li><a href="#" asp-action="SetLanguage" asp-controller="Home" asp-route-returnUrl="@Context.Request.Path" asp-route-culture="@cultureItem.Value">@cultureItem.Text</a>
}

```

Globalisation en Localization, Screenshot 6: Partial View

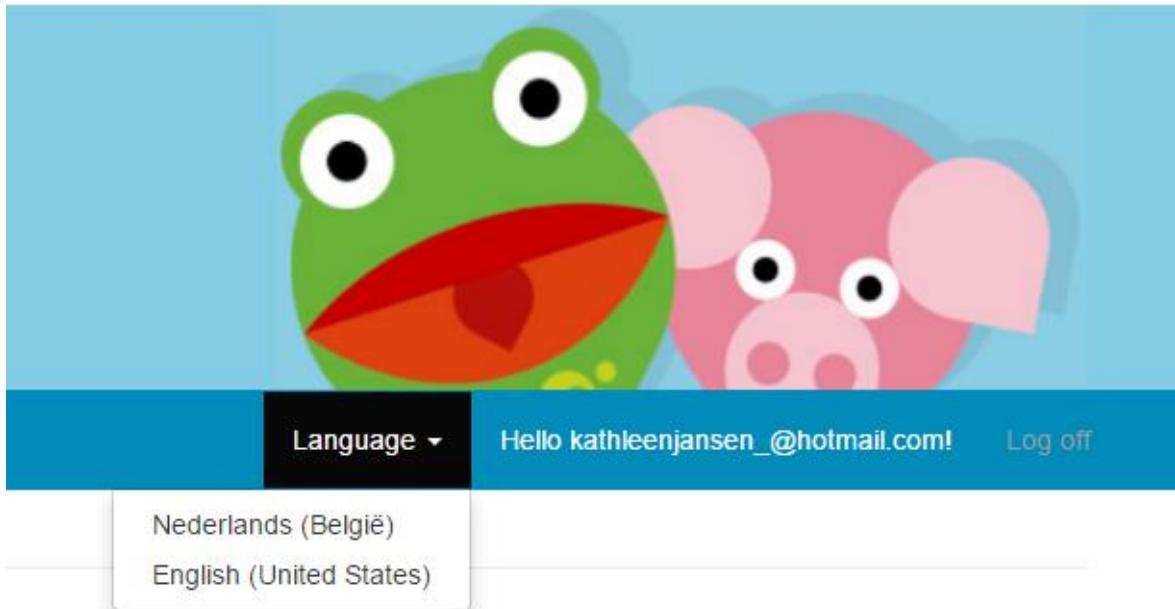
```

[HttpGet]
public IActionResult SetLanguage(string culture, string returnUrl)
{
    Response.Cookies.Append(
        CookieRequestCultureProvider.DefaultCookieName,
        CookieRequestCultureProvider.MakeCookieValue(new RequestCulture(culture)),
        new CookieOptions { Expires = DateTimeOffset.UtcNow.AddYears(1) }
    );

    return LocalRedirect(returnUrl);
}

```

Globalisation en Localization, Screenshot 7: Controller die de culture cookie zet.



Rendering van partial view

7.6.2.5 Stap 5: Localization toepassen in een view

Na dat alles geconfigureerd is, kunnen we translations gaan gebruiken. We doen dit in onze view aan de hand van de `@Localizer` annotatie. Deze annotatie handelt alles af rond vertalingen, als er een key gevonden wordt in de resource file (Globalisation en Localisation, Screenshot 8).

```
using Microsoft.AspNetCore.Mvc.Localization
[inject] IViewLocalizer Localizer
[model] ZwartOpWit.Models.ViewModels.DepartmentVM
{@
}



# @Localizer["Create"]


<form asp-action="Create" asp-controller="Department">
    <div>
        <div class="form-group">
            <label asp-for="department.Name">@Localizer["DepartmentName"]:</label>
            <input asp-for="department.Name" class="form-control" />
            <span asp-validation-for="department.Name" class="text-danger"></span>
        </div>
        <a class="btn btn-default" asp-area="" asp-controller="Department" asp-action="Index">@Localizer["Back"]</a>
        <input type="submit" value="@Localizer["Add"]" class="btn btn-default" />
    </div>
</form>
```

Globalisation en Localization, Screenshot 8: Implementatie Localizer

7.7 Components

7.7.1 Introduction

View Components zijn een feature die pas beschikbaar zijn sinds .net Core. Hierdoor zijn ze wellicht ook iets minder bekend. Ze zijn echter wel zeer handig en makkelijk in gebruik.

7.7.2 Components vs. Partial View

Hoewel components vergelijkbaar zijn met partial views zijn components een pak krachtiger. Zo gebruiken components geen model binding en hangen ze enkel af van de data die jij geeft als je ze oproept.

Een partial view kan geen business logica bevatten buiten de view om. Als je hier wel logica in wilt gebruiken ben je afhankelijk van de controller die de partial oproept.

Omwille van deze voordelen hebben we gekozen om gebruik te maken van een component bij onze menubalk. Dit was specifiek nodig omdat onze menubalk eigenlijk bestaat uit een lijst van machines en deze worden gehaald uit de database.

Kenmerken van een view component:

- Rendert een gedeelte in plaats van een hele respons
- Kan zowel parameters als business logica bevatten
- Bestaat altijd uit een view en een controller
- Doet meestal beroep op een layout page

Je kan view components makkelijk gebruiken in volgende situaties:

- Dynamische navigatie menu's
- Login panelen
- Tag Clouds
- Winkelwagentjes
- ...

Volgende screenshot (Component, Screenshot 1) geeft weer dat we gebruik maken van de component en niet van de controller. We doen dus een database actie buiten de controller om.

```
public class MenuViewComponent : ViewComponent
{
    private readonly AppDbContext _context;

    public MenuViewComponent(AppDbContext context)
    {
        _context = context;
    }

    public async Task<IViewComponentResult> InvokeAsync()
    {
        MenuVM indexVM = new MenuVM();

        indexVM.machineList = _context.Machines.ToList();

        return View(indexVM);
    }
}
```

Component, Screenshot 1: Component klasse

Daarnaast hebben we de view van onze component aangemaakt (Component, Screenshot 2). Om toch de structuur te volgen wordt er hier wel gecommuniceerd via een viewmodel (menuVM). Deze wordt opgevuld door de achterliggende component klasse (Component, Screenshot 1)

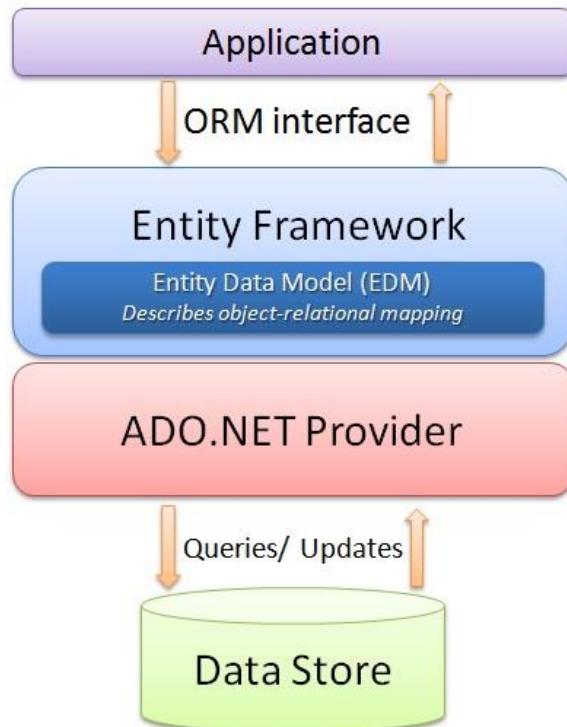
```
@using ZwartOpWit.Models
@model ZwartOpWit.Models.ViewModels.MenuVM
<div class="navbar navbar-inverse navbar-default">
    <div class="container">
        <div class="navbar-header">
            <button type="button" class="navbar-toggle" data-toggle="collapse" data-target=".navbar-collapse">
                <span class="sr-only">Toggle navigation</span>
                <span class="icon-bar"></span>
                <span class="icon-bar"></span>
                <span class="icon-bar"></span>
            </button>
        </div>
        <div class="navbar-collapse collapse">
            <ul class="nav navbar-nav">
                <li><a asp-area="" asp-controller="Home" asp-action="Index">Home</a></li>
                @if (Model != null)
                {
                    @foreach (MachineTypes machineType in Enum.GetValues(typeof(MachineTypes)))
                    {
                        <li class="dropdown">
                            <a href="#" class="dropdown-toggle" data-toggle="dropdown" role="button" aria-haspopup="true" aria-expanded="false">@machineType.ToString()<span class="fa fa-angle-down"></span>
                            <ul class="dropdown-menu">
                                <li><a asp-area="" asp-controller="Job" asp-action="Index" asp-route-filterMachineType="@machineType">Todo</a></li>
                                <li><a asp-area="" asp-controller="Job" asp-action="Index" asp-route-filterMachineType="@machineType">No machine</a></li>
                            <li><small>Machine Type: @machineType</small></li>
                            @if (Model.machineList.Where(x => x.Type == machineType).ToList().Count != 0)
                            {
                                <li role="separator" class="divider"></li>
                                @foreach (var machine in Model.machineList.Where(x => x.Type == machineType).ToList())
                                {
                                    <li><a asp-area="" asp-controller="Job" asp-action="Index" asp-route-machineId="@machine.Id">@machine.Name</a></li>
                                }
                                <li role="separator" class="divider"></li>
                            }
                            <li><a asp-area="" asp-controller="Job" asp-action="Import" asp-route-machineType="@machineType">Import</a></li>
                        </ul>
                    }
                }
            </ul>
            @await Html.PartialAsync("_LoginPartial")
        </div>
    
```

Component, Screenshot 2: View van de component

7.8 Entity Framework

7.8.1 Inleiding

Entity Framework is een open source ORM-framework. ORM staat voor Object Relational Mapping. Entity Framework is een set van technologieën in ADO.NET dat het ontwikkelen van data georiënteerde software applicaties ondersteund. Het maakt het developers gemakkelijk om het datamodel in code te gebruiken. Wij installeerde Entity Framework voor .net Core aan de hand van een Nuget Package.



7.8.2 Wat doet Entity Framework?

Entity Framework genereert objecten en entiteiten volgens de tabellen in de database en voorziet het mechanisme voor het doen van data manipulaties. Ook vergemakkelijkt EF het onderhouden van relaties tussen tabellen.

7.8.3 Voordelen

Entity Framework heeft als voordeel dat we het model beter kunnen voorstellen door gebruik te maken van relaties tussen de entiteiten. Ook maakt EF het mogelijk om logica in verband met data acces niet meer te schrijven op database niveau aan de hand van SQL, maar wel in programmeer code, in ons geval c# in combinatie met Linq. Dit geeft ook als voordeel dat de onderliggende data store makkelijker kan vervangen worden zonder te veel overhead, aangezien alle data acces logica zich op een hoger niveau bevindt.

7.9 CRUD

Voor het beheren van data maken we gebruik van CRUD. Dit staat voor de acties Create, Read, Update en Delete.

- Create : Het toevoegen van nieuwe gegevens
- Read: Het opvragen van gegevens
- Update: Het wijzigen van gegevens
- Delete: Het verwijderen van gegevens



We zullen de werking van de CRUD verder verduidelijken door een uitgewerkte CRUD te behandelen. Hieronder een overzicht van de CRUD Machines.

7.9.1 Index

De index pagina is de pagina waarvan alle CRUD acties beginnen. De index pagina geeft een overzicht van de records en laat toe om deze te filteren en te sorteren. Als er genoeg records zijn wordt er ook gebruik gemaakt van pagina's. Deze pagina's zullen dan een x-aantal records per pagina tonen om te voorkomen dat je het overzicht in de lijst verliest.

In de controller bestaat er voor de index pagina enkel een methode van het type get omdat er enkel data opgehaald wordt en geen data gemanipuleerd word. Deze methode handelt alle logica af.

Machines

Machines				
New machine				
Find by name: <input type="text"/> Search Back to full list				
Machine name	Calculation method	Type	Department	Action
Busch	None	Busch	Department 2	
Fold	Always5000Ticks	Fold	Department 3	
I-Stitch	None	Stitch	Department 3	

[Previous](#) [Next](#)

CRUD – Index overzicht

7.9.1.1 Filteren

Het filteren gebeurt door dynamisch like functies voor de doorzoekbare velden toe te voegen als er een waarde in het zoek veld is ingevuld.

```
var machines = _context.Machines.Include(m => m.Department).AsQueryable();

if (!string.IsNullOrEmpty(searchString))
{
    machines = machines.Where(s => s.Name.Contains(searchString));
}
```

CRUD – index filter toepassen

7.9.1.2 Rangschikken

Het rangschikken gebeurt door te kijken of er de vorige keer al op hetzelfde veld gesorteerd is. Als dit zo is keer de sortering dan om anders sorteert aflopend op het nieuwe veld.

```
machineListVm.nameSortParm = string.IsNullOrEmpty(sortOrder) ? "name_desc" : "";
machineListVm.calculationMethodSortParm = sortOrder == "calcMethod" ? "calcMethod_desc" : "calcMethod";
machineListVm.typeSortParm = sortOrder == "type" ? "type_desc" : "type";
machineListVm.departmentSortParm = sortOrder == "department" ? "department_desc" : "department";
```

CRUD – Index rangschikking bepalen

```
switch (sortOrder)
{
    case "name_desc":
        machines = machines.OrderByDescending(m => m.Name);
        break;
    case "calcMethod":
        machines = machines.OrderBy(m => m.CalculationMethod);
        break;
    case "calcMethod_desc":
        machines = machines.OrderByDescending(m => m.CalculationMethod);
        break;
    case "type":
        machines = machines.OrderBy(m => m.Type);
        break;
    case "type_desc":
        machines = machines.OrderByDescending(m => m.Type);
        break;
    case "department":
        machines = machines.OrderBy(m => m.Department.Name);
        break;
    case "department_desc":
        machines = machines.OrderByDescending(m => m.Department.Name);
        break;
    default:
        machines = machines.OrderBy(m => m.Name);
        break;
}
```

CRUD – index rangschikken doen

7.9.1.3 In pagina's verdelen

Voor het verdelen van de lijst in pagina's maken we gebruik van een helper klasse. Deze helper klasse paginatedList is een extensie op de gewone list. De paginated list bevat enkel extra attributen die nodig zijn voor het beheren van de pagina's. Het totaal aantal pagina's en de huidige pagina.

Er zijn ook extra methodes aangebracht om te kijken of er nog een vorige of volgende pagina is. Ook is er een constructor toegevoegd om de nieuwe attributen correct in te vullen.

```
public class PaginatedList<T> : List<T>
{
    public int PageIndex { get; private set; }
    public int TotalPages { get; private set; }

    public PaginatedList(List<T> items, int count, int pageIndex, int pageSize)
    {
        PageIndex = pageIndex;
        TotalPages = (int)Math.Ceiling(count / (double)pageSize);

        this.AddRange(items);
    }

    public PaginatedList()
    {
        PageIndex = 0;
        TotalPages = 0;
    }

    public bool HasPreviousPage
    {
        get
        {
            return (PageIndex > 1);
        }
    }

    public bool HasNextPage
    {
        get
        {
            return (PageIndex < TotalPages);
        }
    }

    public static async Task<PaginatedList<T>> CreateAsync(IQueryable<T> source, int pageIndex, int pageSize)
    {
        var count = await source.CountAsync();
        var items = await source.Skip((pageIndex - 1) * pageSize).Take(pageSize).ToListAsync();
        return new PaginatedList<T>(items, count, pageIndex, pageSize);
    }
}
```

CRUD – Index paginated list class

In de view wordt aan de hand van de paginatedList bepaald of er nog een volgende of vorige pagina is. Als de knop volgende of vorige geactiveerd wordt zal opnieuw de index methode aangeroepen worden met de correcte parameters en de volgende pagina geladen worden.

```
@{
    var prevDisabled = !Model.machineList.HasPreviousPage ? "disabled" : "";
    var nextDisabled = !Model.machineList.HasNextPage ? "disabled" : "";
}
<div>
```

CRUD – Index pagination determination has next, has previous page

```

<a asp-action="Index"
    asp-route-sortOrder="@ViewData["CurrentSort"]"
    asp-route-page="@{Model.machineList.PageIndex - 1}"
    asp-route-currentFilter="@ViewData["CurrentFilter"]"
    class="btn btn-default @prevDisabled">
    @Localizer["Previous"]
</a>
<a asp-action="Index"
    asp-route-sortOrder="@ViewData["CurrentSort"]"
    asp-route-page="@{Model.machineList.PageIndex + 1}"
    asp-route-currentFilter="@ViewData["CurrentFilter"]"
    class="btn btn-default @nextDisabled">
    @Localizer["Next"]
</a>

```

CRUD – Index pagination next previous

7.9.2 Create

De create bestaat uit 1 form en in de controller zijn er 2 methodes voor de create een HttpGet en HttpPost methode. Deze methodes zorgen ervoor dat de create pagina geopend kan worden en het form verwerkt kan worden.

In de HttpGet methode zit er logica om de lijst van departementen op te halen om zo de selectie box op het create form te vullen.

Create

Machine name:

Type:

Calculation method:

Department

Setup time:

Run time to 1000:

Run time to 1000 factor:

Run time from 1000:

Run time from 1000 factor:

CRUD – Create form

```
[HttpGet]
public ViewResult Create()
{
    MachineVM machineVM = new MachineVM();

    machineVM.selectDepartments = _context.Departments.Select(x => new SelectListItem() { Text = x.Name, Value = x.Id.ToString() }).ToList();

    return View(machineVM);
}
```

CRUD – Create get method

In de `HttpPost` methode wordt de nieuwe aangemaakt. Voor wegschrijven van het machine record naar de database gebruiken we entity framework.

```
[HttpPost]
public IActionResult Create(Machine machine)
{
    _context.Machines.Add(machine);
    _context.SaveChanges();
    return RedirectToAction("Index");
}
```

CRUD – Create post method

7.9.3 Read

De read pagina laat alle parameters van een machine zien. Het is niet mogelijk om hier parameters van de machine te wijzigen. Deze pagina is handig als gebruikers geen rechten mogen hebben om de entiteit aan te passen maar wel om deze te zien.

Read

Machine name	Busch
Type	Busch
Calculation method	None
Department	Department 2
Setup time	2
Run time to 1000	3
Run time to 1000 factor	0
Run time from 1000	2
Run time from 1000 factor	3

[Back](#)

CRUD – Read view

Om de department naam weer te geven maken we gebruik van een include hiermee wordt het gerelateerde department record ook in de query opgehaald.

```
[HttpGet]
public ViewResult Read(int id)
{
    MachineVM machineVM = new MachineVM();
    machineVM.machine = _context.Machines.Include(m => m.Department).FirstOrDefault(x => x.Id == id);
    return View(machineVM);
}
```

CRUD – Read method

7.9.4 Update

Bij de update wordt er net als bij de create gebruik gemaakt van een `HttpGet` en `HttpPost` methode. De `HttpGet` methode zal het formulier vullen maar t.o.v. de create wordt nu ook het geselecteerde record opgehaald. Ook hier wordt opnieuw de data opgehaald om de selectieboxen te vullen.

Update

Machine name:

Type:

Calculation method:

Department:

Setup time:

Run time to 1000:

Run time to 1000 factor:

RuntimeFrom1000:

RuntimeFrom1000Factor:

CRUD – Update from

```
[HttpGet]
public ViewResult Update(int id)
{
    MachineVM machineVM = new MachineVM();
    machineVM.machine = _context.Machines.FirstOrDefault(x => x.Id == id);
    machineVM.selectDepartments = _context.Departments.Select(x => new SelectListItem() { Text = x.Name, Value = x.Id.ToString() }).ToList();
    return View(machineVM);
}
```

CRUD – Update get method

In de `HttpPost` methode wordt het formulier verwerkt en de wijziging bewaard in de database. Hier gebruiken we Entity Framework voor.

```
[HttpPost]
public IActionResult Update(Machine machine)
{
    _context.Entry(machine).State = Microsoft.EntityFrameworkCore.EntityState.Modified;
    _context.SaveChanges();
    return RedirectToAction("Index");
}
```

CRUD – Update post method

7.9.5 Delete

De delete bestaat uit 1 methode als deze van de index pagina wordt opgeroepen word het huidig record verwijderd. Om dit te kunnen doen maken we gebruik van entity framework. Voor de delete wordt uitgevoerd wordt er eerst gekeken of de machine wel bestaat.

```
[HttpGet]
public IActionResult Delete(int id)
{
    var original = _context.Machines.FirstOrDefault(e => e.Id == id);
    if (original != null)
    {
        _context.Machines.Remove(original);
        _context.SaveChanges();
    }
    return RedirectToAction("Index");
}
```

CRUD – delete method

7.10 Patterns

7.10.1 Inleiding

Design patterns zijn stukken code die te herbruiken zijn om gekende problemen op te lossen. In ons project hebben wij er enkele gebruikt omdat dit aan de ene kant gemakkelijk was en aan de andere kant omdat dit werd verplicht door .net core wegens best practices regels.

7.10.2 Options pattern

Het options pattern is een pattern dat gebruikt wordt om via een klasse een groep van generaliseerde settings weer te geven. Deze klassen kunnen dan bijvoorbeeld vertaald worden naar JSON. Voor het options pattern is het belangrijk dat de klassen een parameterloze constructor hebben. Deze opties zijn dan via dependency injection beschikbaar in andere klassen.

Dit pattern is toegepast voor het configureren van de mail service.

```
namespace ZwartOpwit.Models.Services
{
    public class AuthMessageSenderOptions
    {
        public string SendGridUser { get; set; }
        public string SendGridKey { get; set; }
    }
}
```

Options pattern – Options klasse

```
//Configure mail service
services.Configure<AuthMessageSenderOptions>(myOptions =>
{
    myOptions.SendGridKey = "GridKey";
    myOptions.SendGridUser = "GridUser";
});
```

Options pattern – Options klasse vullen

7.10.3 Simple factory pattern (Calculation method)

Het simple factory pattern is een pattern waar aan de hand van een enum een bepaalde method op een klasse aangeroepen wordt. Wij hebben dit gebruikt om de calculatie methodes aan de machines te koppelen. Op de machine wordt een waarde van de enum gekozen die daarna overeenkomt met een methode in de calculatie logica. Alle klassen die in de factory gebruikt worden implementeren een bepaalde interface.

```
namespace ZwartOpWit.Helpers.CalculationMethods
{
    interface ICalculationMethod
    {
        TimeSpan calculate(JobLine jobLine);
    }
}
```

Simple factory pattern – Interface

```
namespace ZwartOpWit.Helpers.CalculationMethods
{
    public class StichMainCalculationMethod : ICalculationMethod
    {
        public TimeSpan calculate(JobLine jobLine)
        {
            Machine machine = jobLine.Machine;

            int stations = jobLine.Job.PageQuantity / 4;
            int quantity = jobLine.Job.Quantity;

            double instelMachine = machine.SetupTime + (stations * machine.SetupTimeStationFactor);
            double startDraai1000 = machine.RunTimeTo1000Speed + (stations * machine.RunTimeTo1000SpeedStationFactor);
            double doorDraai1000 = machine.RunTimeFrom1000Speed + (stations * machine.RunTimeFrom1000SpeedStationFactor);
            double quantityDoorDraai = quantity - 1000;
            double centiTime = instelMachine + startDraai1000 + (doorDraai1000 * quantityDoorDraai / 1000);

            int hour = 0;
            int minute = 0;
            int second = 0;

            if (centiTime >= 1)
            {
                hour = 1;
                centiTime--;
            }

            double centiMinute = centiTime % 1;
            minute = (int)Math.Round(centiMinute * 60);

            double centiSecond = (centiMinute * 60) - minute;
            second = (int)Math.Round(centiSecond * 60);

            TimeSpan planned = new TimeSpan(hour, minute, second);
            return planned;
        }
    }
}
```

Simple factory pattern – Calculation method implementing the interface

```
public void calculateTime(Machine machine)
{
    ICalculationMethod calcMethod = null;

    if (machine != null)
    {
        this.Machine = machine;

        switch (machine.CalculationMethod)
        {
            case CalculationMethodTypes.StichMain:
                calcMethod = new StichMainCalculationMethod();
                break;
            case CalculationMethodTypes.Always1Hour:
                calcMethod = new Always1Hour();
                break;
            case CalculationMethodTypes.Always5000Ticks:
                calcMethod = new Always5000Ticks();

                break;
        }

        if (calcMethod != null)
        {
            CalculatedTime = calcMethod.calculacte(this);
        }
    }
}
```

Simple factory pattern – Factory

7.11 CSV Import

De csv bestanden die vanuit filemaker doorgestuurd worden kunnen via de code hieronder geupload worden. Via de Interface IformFile wordt het bestand naar de, in dit geval map ‘uploads’ gestuurd.

```
[HttpGet]
public IActionResult Import(MachineTypes machineType)
{
    JobImportVM jobImportVM = new JobImportVM();
    jobImportVM.machineType = machineType;

    return View(jobImportVM);
}
```

Het bestand wordt daarna opgeroepen via StreamReader, gecombineerd met FileStream, door gebruik te maken van het eerst gekozen pad. Via split worden dan de juiste gegevens in Job en JobLine gezet en aan de context toegevoegd. Als alle gegevens verzameld zijn wordt de context naar de database geschreven.

```
using (StreamReader reader = new StreamReader(new FileStream(Path.Combine(uploads, files.FileName), FileMode.Open)))
{
    currentLine = reader.ReadLine();

    while (!reader.EndOfStream)
    {
        currentLine = reader.ReadLine();
        lines.Add(currentLine);
    }
}

foreach (string line in lines)
{
    splitArray = line.Split(';');

    //Check if job exists with current job number
    job = _context.Jobs.FirstOrDefault(z => z.JobNumber == splitArray[1]);

    //Create Job & JobLine
    if (job == null)
    {
        //Create Job
        job = new Job();

        job.DeliveryDate = DateTime.ParseExact(splitArray[4], "dd/MM/yyyy", null);
        job.JobNumber = splitArray[1];

        double aantal = double.Parse(splitArray[2]);
        job.Quantity = Convert.ToInt16(aantal);

        job.PaperBw = splitArray[8];
        job.Cover = 0;
        job.PaperCover = "no cover";
        job.Height = 297;
        job.Width = 210;
        job.PageQuantity = int.Parse(splitArray[6].Remove(2));

        _context.Jobs.Add(job);
    }

    //Create JobLine
    jobLine = new JobLine();

    jobLine.JobId = job.Id;
    jobLine.Job = job;
    jobLine.Sequence = 1;
    jobLine.MachineType = machineType;
    jobLine.Completed = false;

    _context.JobLines.Add(jobLine);
}
```

8 Werking programma

8.1 Users beheren

8.1.1 Introductie

Als admin-gebruiker kan je gebruikers beheren. Gebruikers kunnen inloggen op het systeem. Er bestaan 2 rollen. Een admin-rol en een gewone gebruiker-rol.

De optie gebruikers beheren is enkel zichtbaar in de admin navigatiebalk. Deze is enkel als je een admin-user bent beschikbaar.

8.1.2 Index pagina

Nadat je op de knop “Gebruikers” in de admin navigatiebalk hebt geklikt, kom je op de index pagina van “Gebruikers”. Deze geeft een lijst weer van alle huidige gebruikers.

Hier heb je verschillende mogelijkheden en opties. Voor het beheren van de gebruikers werken we met CRUD (zie hoofdstuk CRUD). Verder heb je de mogelijkheid om een gebruiker te filteren door gebruik te maken van “Zoeken”. Ook zijn er paginaknoppen voorzien voor als de lijst te uitgebreid wordt.



Screenshot: Navigatiebalk voor admin – ga naar Gebruikers

Gebruikers Machines Departments Tijds registraties

b
zwartopwit.be
DRUKWERK VAN A TOT XXL



Home Stitch+ Typo+ Fold+ Score+ Busch+ Language - Hello kathleenjansen_@hotmail.com! Log off

Gebruikers

Nieuwe gebruiker

Zoeken Zoeken | Terug naar volledige lijst

Email

Actie

George@test.com



JamesietheGreat@miauwkes.com



kathleenjansen_@hotmail.com



Vorige

Volgende

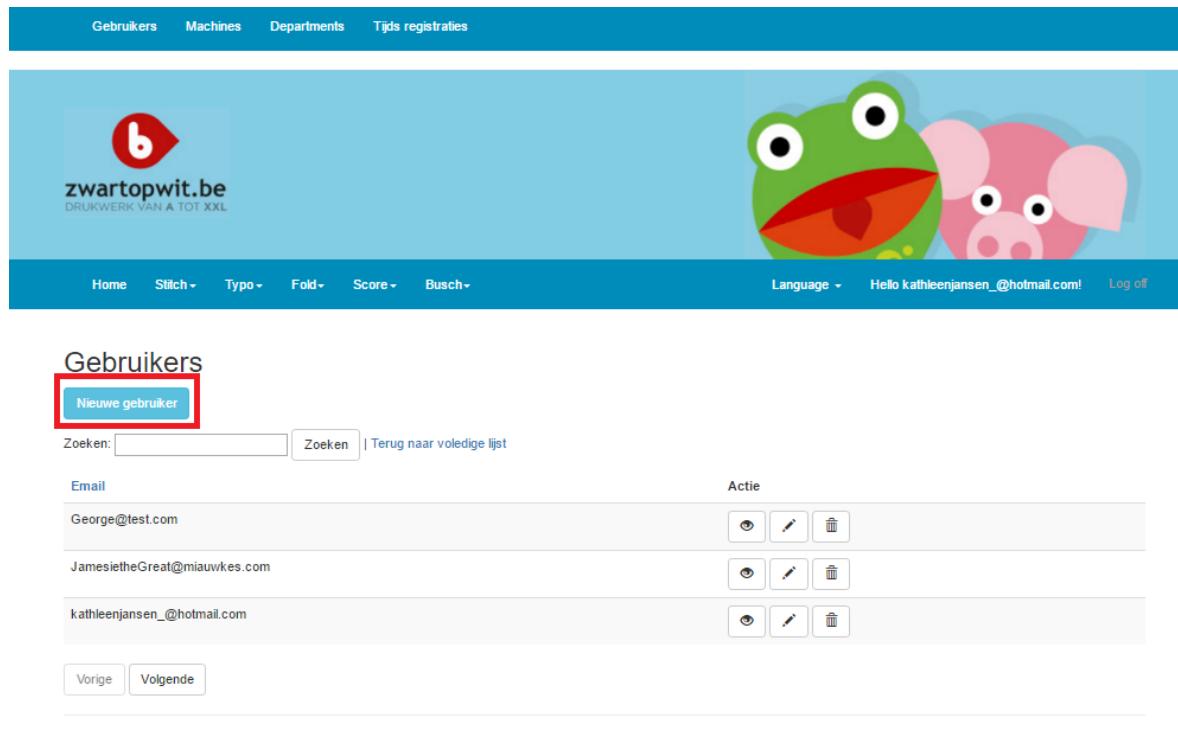
© 2017 - ZwartOpWit

Screenshot Gebruikers – Index pagina

8.1.3 User CRUD

8.1.3.1 Create

Als we op de knop “Nieuwe gebruiker” klikken komen we op de create pagina voor een nieuwe gebruiker. Hier kunnen we een nieuwe gebruiker aanmaken.



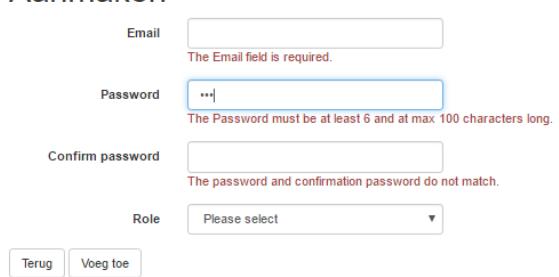
The screenshot shows the zwartOpwit.be website interface. At the top, there's a navigation bar with links for 'Gebruikers', 'Machines', 'Departments', and 'Tijds registraties'. Below the header, there's a logo for 'zwartOpwit.be' with the tagline 'DRUKWERK VAN A TOT XXL'. To the right of the logo is a cartoon illustration of two characters. The main content area is titled 'Gebruikers'. A red box highlights the 'Nieuwe gebruiker' button. Below it, there's a search bar with 'Zoeken:' and a 'Zoeken' button, followed by a link 'Terug naar volledige lijst'. The main table lists users with columns for 'Email' and 'Actie'. Each row contains three icons: a magnifying glass, a pencil, and a trash can. At the bottom of the table are 'Vorige' and 'Volgende' buttons. The footer contains the copyright notice '© 2017 - ZwartOpWit'.

Screenshot: Index pagina – Nieuwe gebruiker aanmaken

Een gebruiker bestaat uit een e-mailadres (username), een paswoord en een rol. Om typfouten tegen te gaan wordt er 2x naar het paswoord gevraagd. Uit rol kan men kiezen tussen admin of employee. Alle velden zijn verplicht in te vullen, anders krijg je een foutmelding. Ook wordt er gevalideerd op de lengte van de ingevoerde tekst.

Meer informatie over het gebruik van rollen en identity vind je terug in hoofdstuk 8.6 Identity.

Aanmaken



The screenshot shows the 'Aanmaken' (Create) form. It has four input fields: 'Email' (with error message 'The Email field is required.'), 'Password' (with error message 'The Password must be at least 6 and at max 100 characters long.'), 'Confirm password' (with error message 'The password and confirmation password do not match.'), and a 'Role' dropdown menu ('Please select'). At the bottom are 'Terug' and 'Voeg toe' buttons. The footer contains the copyright notice '© 2017 - ZwartOpWit'.

Screenshot: Gebruikers – Nieuwe gebruiker aanmaken + validatie

8.1.3.2 Read

Door op het oog – icoontje te klikken in de gebruikers lijst, krijg je een weergave van de details van de gekozen gebruiker.

Gebruikers

The screenshot shows a user management interface. At the top left is a button labeled 'Nieuwe gebruiker'. Below it is a search bar with the placeholder 'Zoeken:' and a 'Zoeken' button. To the right of the search bar is a link 'Terug naar volledige lijst'. The main area contains a table with three rows of user data. Each row includes an 'Email' column (containing email addresses like 'George@test.com', 'JamesietheGreat@miauwkes.com', and 'kathleenjansen_@hotmail.com') and an 'Actie' (Action) column with three icons: a magnifying glass, a pencil, and a trash bin. The first magnifying glass icon in the 'Actie' column of the first row is highlighted with a red box. At the bottom left are navigation buttons for 'Vorige' and 'Volgende'.

© 2017 - ZwartOpWit

Screenshot: Gebruikers - Read

Deze worden weergegeven in een overzichtelijke lijst. Het paswoord van de gebruiker wordt natuurlijk niet getoond. Indien je wil terugkeren naar de vorige pagina klik je op “terug”.

Lezen

The screenshot shows a user detail view. At the top left is an 'Email' label followed by the email address 'kathleenjansen_@hotmail.com'. Below this is a 'Terug' (Back) button.

Screenshot: Gebruikers – Read detailvoorstelling in lijst

8.1.3.3 Update

Als je op het potlood icoontje klikt, kan je de gegevens van de gekozen gebruiker wijzigen.

Gebruikers

Nieuwe gebruiker

Zoeken: Zoeken | Terug naar volledige lijst

Email	Actie
George@test.com	<input type="radio"/> <input type="button" value="Edit"/> <input type="button" value="Delete"/>
JamesietheGreat@miauwkes.com	<input type="radio"/> <input type="button" value="Edit"/> <input type="button" value="Delete"/>
kathleenjansen_@hotmail.com	<input type="radio"/> <input type="button" value="Edit"/> <input type="button" value="Delete"/>

Vorige

© 2017 - ZwartOpWit

Screenshot: Gebruikers - Update

Deze heeft dezelfde invoervelden als de Create/Aanmaken pagina. De reeds bestaande gegevens zijn opgehaald en worden ineens ingevuld. Indien de gebruiker klaar is met updaten klikt hij onderaan op wijzigen en komt hij terug op de index pagina.

Wijzigen

Email

Role

© 2017 - ZwartOpWit

Screenshot: Gebruikers – Wijzigen detail pagina

8.1.3.4 Delete

Om een departement te verwijderen druk je op het vuilbak-icoontje op de index pagina. Het departement wordt dan verwijderd.

Gebruikers

The screenshot shows a user management interface. At the top left is a blue button labeled "Nieuwe gebruiker". Below it is a search bar with the placeholder "Zoeken:" containing the letter "a", and a "Zoeken" button. To the right of the search bar is a link "Terug naar volledige lijst". The main area contains a table with three rows of user data:

Email	Actie
George@test.com	
JamesietheGreat@miauwkes.com	
kathleenjansen_@hotmail.com	

At the bottom left are "Vorige" and "Volgende" navigation buttons. A copyright notice "© 2017 - ZwartOpWit" is at the bottom center.

Screenshot: Gebruikers – Verwijderen

8.1.4 Zoeken

Je kan een gebruiker gemakkelijk terugvinden door de naam of een gedeelte hiervan in te geven in de zoekbalk. Deze is niet hoofdletter gevoelig. Wil je daarna terugkeren naar de volledige lijst druk je op “Terug naar volledige lijst”.

Gebruikers

The screenshot shows the same user management interface as the previous one, but with a red box highlighting the search input field "Zoeken:" which contains the letter "a". A red arrow points from the text "Screenshot: Gebruikers – Filteren" to this search field. The rest of the interface is identical to the first screenshot.

Screenshot: Gebruikers – Filteren

8.1.5 Navigeren

Omdat de mogelijkheid bestaat dat de pagina veel gebruikers bevat, hebben we navigatieknoppen voorzien. Met de knoppen “Vorige” en “Volgende” kan je gemakkelijk naar de volgende pagina. De knop “Vorige” is niet klikbaar op de eerste pagina, de knop “Volgende” is niet klikbaar op de laatste pagina. Het aantal gebruikers dat wordt weergegeven is momenteel ingesteld op 3. Deze kan nog gemakkelijk aangepast worden in de code.

Gebruikers

Nieuwe gebruiker

Zoeken: Zoeken | Terug naar volledige lijst

Email	Actie
George@test.com	
JamesietheGreat@miauwkes.com	
kathleenjansen_@hotmail.com	

Vorige Volgende

Screenshot: Gebruikers – Navigeren door index pagina

8.2 Departementen beheren

8.2.1 Introductie

Als admin-gebruiker kan je departementen beheren. Machines behoren tot een departement.

De optie departementen beheren is enkel zichtbaar in de admin navigatiebalk. Dus enkel als je een admin-user bent is deze beschikbaar.

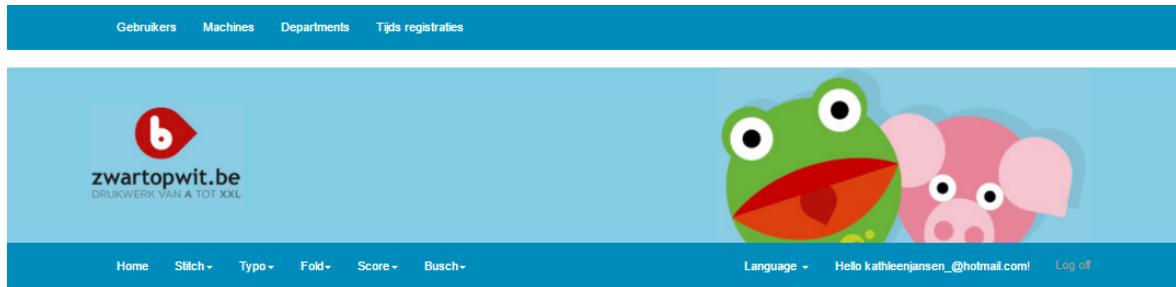
The screenshot shows the top navigation bar of a web application. The tabs are "Gebruikers", "Machines", "Departments" (which has a red arrow pointing to it), and "Tijds registraties". Below the tabs, there is a logo for "zwartopwit.be" and a cartoon illustration of a green frog and a pink elephant. At the bottom of the bar, there are links for "Home", "Slecht+", "Typo", "Fold", "Score", "Busch", "Language", "Hello kathleenjansen_@hotmail.com!", and "Log off".

Screenshot: Navigatiebalk voor admin – Ga naar Departments

8.2.2 Index pagina

Nadat je op de knop “Departments” in de admin navigatiebalk hebt geklikt, kom je op de index pagina van “Departments”. Deze geeft een lijst weer van alle huidige departementen.

Hier heb je verschillende mogelijkheden en opties. Voor het beheren van de departments werken we met CRUD (zie hoofdstuk CRUD). Verder heb je de mogelijkheid om een department te filteren door gebruik te maken van “Zoeken”. Ook zijn er paginaknoppen voorzien voor als de lijst te uitgebreid wordt.



Departementen

[Nieuw departement](#)

Zoeken: [Zoeken](#) | Terug naar voelde lijst

Departement naam

Actie

Department 1



Department 2



Department 3



[Vorige](#) [Volgende](#)

© 2017 - ZwartOpWit

Screenshot: Departementen – Index pagina

8.2.3 Department CRUD

8.2.3.1 Create

Als we op de knop “Nieuw departement” klikken komen we op de create pagina voor een departement. Hier kunnen we een nieuw departement aanmaken.

Departementen

[Nieuw departement](#)

Zoeken: [Zoeken](#) | Terug naar voelde lijst

Departement naam

Actie

Department 1



Department 2



Department 3



[Vorige](#) [Volgende](#)

© 2017 - ZwartOpWit

Screenshot: Departementen – Nieuw departement toevoegen

Een departement bestaat enkel uit een naam. Deze is verplicht in te vullen, anders krijg je een foutmelding. Deze moet ook minimum 3 karakters bevatten.

Aanmaken

Departement naam:

© 2017 - ZwartOpWit

Aanmaken

Departement naam:

The Name field is required.

© 2017 - ZwartOpWit

Screenshot: Departementen – Aanmaken + validatie

8.2.3.2 Read

Door op het oog – icoontje te klikken in de departementen lijst, krijg je een weergave van de details van het departement.

Departementen

Departement naam		Actie
Department 1		
Department 2		
Department 3		

© 2017 - ZwartOpWit

Screenshot: Departementen - Read

Deze worden weergegeven in een overzichtelijke lijst. Indien je wil terugkeren naar de vorige pagina klik je op “terug”.

Lezen

Department name

Department 1

© 2017 - ZwartOpWit

Screenshot: Departementen – Read detail voorstelling

8.2.3.3 Update

Als je op het potlood icoontje klikt kan je de gegevens van het gekozen departement wijzigen.

Departementen

Nieuw departement

Zoeken: Zoeken | Terug naar voledige lijst

Departement naam	Actie
Department 1	
Department 2	
Department 3	

Vorige

© 2017 - ZwartOpWit

Screenshot: Departementen - Update

Deze heeft dezelfde invoervelden als de Create/Aanmaken pagina. De reeds bestaande gegevens zijn opgehaald en worden ineens ingevuld. Indien de user klaar is met updaten klikt hij onderaan op wijzigen en komt hij terug op de index pagina.

Wijzigen

Departement naam:

Department 1

Terug

Wijzigen

© 2017 - ZwartOpWit

Screenshot: Departementen – Wijzigen detail voorstelling

8.2.3.4 Delete

Om een departement te verwijderen druk je op het vuilbak-icoontje op de index pagina. Het departement wordt dan verwijderd.

Departementen

Nieuw departement

Zoeken: Zoeken | Terug naar voledige lijst

Departement naam	Actie
Department 1	
Department 2	
Department 3	

Vorige

© 2017 - ZwartOpWit

Screenshot: Departementen - Verwijderen

8.2.4 Zoeken

Je kan een departement gemakkelijk terugvinden door de naam of een gedeelte hiervan in te geven in de zoekbalk. Deze is niet hoofdletter gevoelig. Wil je daarna terugkeren naar de volledige lijst druk je op “Terug naar volledige lijst”.

The screenshot shows a search interface for departments. At the top, there is a blue button labeled 'Nieuw departement'. Below it is a search bar containing the number '2', with a red box and arrow highlighting the search button next to it. To the right of the search bar is a link 'Terug naar volledige lijst'. The main area displays a table with one row for 'Department 2'. The table has columns for 'Departement naam' and 'Actie'. Under 'Actie' are three icons: a magnifying glass, a pencil, and a trash can. At the bottom left are navigation buttons for 'Vorige' and 'Volgende'.

Screenshot: Departementen - zoeken

8.2.5 Navigeren

Omdat de mogelijkheid bestaat dat de pagina veel departementen bevat, hebben we navigatieknoppen voorzien. Met de knoppen “Vorige” en “Volgende” kan je gemakkelijk naar de volgende pagina. De knop “Vorige” is niet klikbaar op de eerste pagina, de knop “Volgende” is niet klikbaar op de laatste pagina. Het aantal departementen dat wordt weergegeven is momenteel ingesteld op 3. Deze kan nog gemakkelijk aangepast worden in de code.

The screenshot shows the index page for departments. It features a search bar at the top with a red box around the 'Zoeken' button. The main content area contains a table with three rows, each representing a department: 'Department 1', 'Department 2', and 'Department 3'. Each row includes an 'Actie' column with three icons. At the bottom, there are navigation buttons for 'Vorige' and 'Volgende', which are also highlighted with a red box.

Screenshot: Departementen – Navigeren door index pagina

8.3 Machines beheren

8.3.1 Introductie

Als admin-gebruiker kan je machines beheren. Met machines bedoelen we de afwerkingsmachines van Zwart op Wit. Vaak heeft een drukwerk nog afwerking nodig. Dit kan gaan van nietjes, vouwen, rillen,... Alles om het vouwwerk tot in de details af te werken.

De optie machines beheren is enkel zichtbaar in de admin navigatiebalk. Dus enkel als je een admin-user bent is deze beschikbaar.



Screenshot: Navigatiebalk voor admin – Ga naar machines beheren

8.3.2 Index pagina

Nadat je op de knop “Machines” in de admin navigatiebalk hebt geklikt, kom je op de index pagina van “Machines”. Deze geeft een lijst weer van alle huidige machines.

Hier heb je verschillende mogelijkheden en opties. Voor het beheren van de machines werken we met CRUD (zie hoofdstuk CRUD). Verder heb je de mogelijkheid om een machine te filteren door gebruik te maken van “Zoeken”. Ook zijn er paginaknoppen voorzien voor als de lijst te uitgebreid wordt.

Machine naam	Calculatie methode	Type	Departement	Actie
Busch Test	StichMain	Stich	Department 2	
Fold	Always5000Ticks	Fold	Department 3	
I-Stitch	None	Fold	Department 3	

Machines

[Nieuwe machine](#)

Zoeken: [Zoeken](#) | Terug naar volledige lijst

Vorige [Volgende](#)

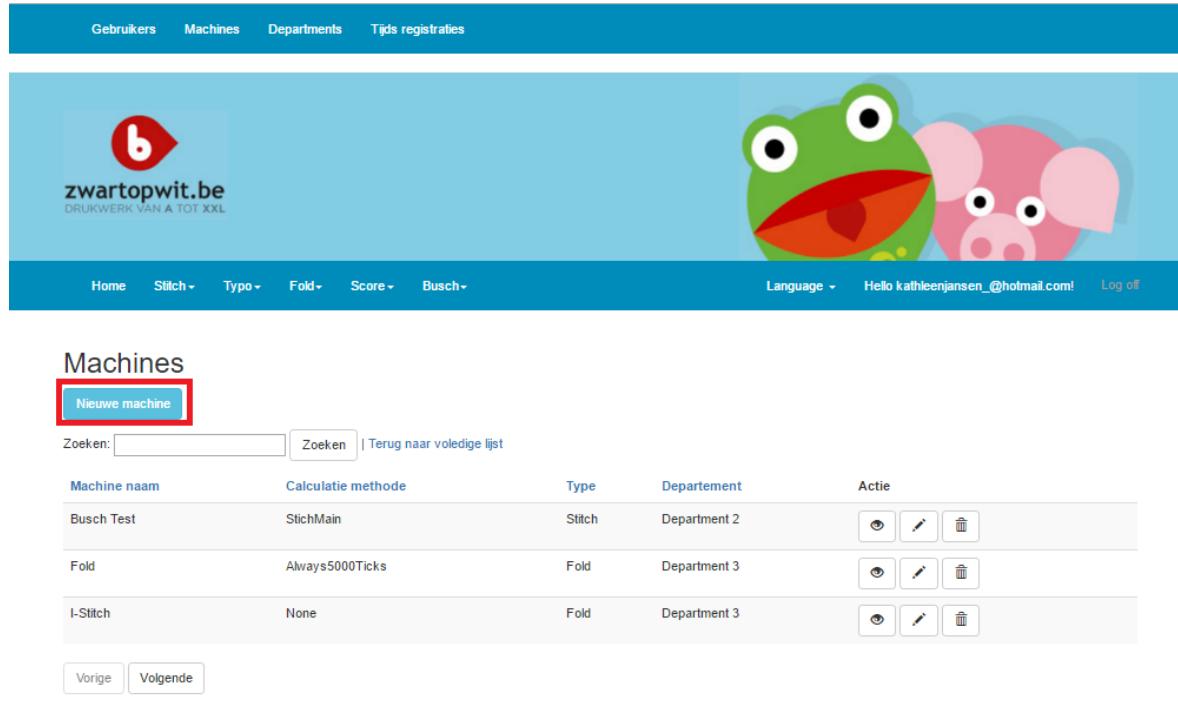
© 2017 - ZwartOpWit

Screenshot: Machines – Index pagina

8.3.3 Machine CRUD

8.3.3.1 Create

Als we op de knop “Nieuwe machine” klikken komen we op de create pagina voor een machine. Hier kunnen we een nieuwe machine aanmaken.



The screenshot shows the ZwartOpWit website interface. At the top, there is a navigation bar with links for 'Gebruikers', 'Machines', 'Departments', and 'Tijds registraties'. Below the navigation bar is a header featuring the ZwartOpWit logo (a stylized 'b' inside a red circle) and the text 'zwartopwit.be DRUKWERK VAN A TOT XXL'. To the right of the header is a cartoon illustration of a green frog and a pink elephant. The main content area has a blue header with the text 'Machines'. Below this, a button labeled 'Nieuwe machine' is highlighted with a red box. There is also a search bar with the placeholder 'Zoeken:' and a 'Zoeken' button. A link 'Terug naar volledige lijst' is also present. The main table lists three existing machines: 'Busch Test', 'Fold', and 'I-Stitch', each with columns for 'Machine naam', 'Calculatie methode', 'Type', 'Departement', and 'Actie' (with icons for edit and delete). At the bottom of the page are 'Vorige' and 'Volgende' navigation buttons, and a copyright notice '© 2017 - ZwartOpWit'.

Screenshot: Machines – Nieuwe machine aanmaken

Aanmaken

Machine naam:

Type:

Calculatiemethode:

Departement

Setup time:

Run time tot 1000:

Run time tot 1000 factor:

Run time vanaf 1000:

Run time vanaf 1000 factor:

[Terug](#)

[Voeg toe](#)

© 2017 - ZwartOpWit

Screenshot: Machines: Nieuwe machine aanmaken detail

Als we een machine willen aanmaken moeten we de nodige gegevens invullen. De machine krijgt een naam, type, calculatiemethode, departement en de nodige gegevens die de calculatiemethode nodig heeft voor de berekening van de doorlooptijd van de job.

De setup tijd is de tijd die nodig is om de machine in te stellen.

Om de berekening van de doorlooptijd te doen zijn volgende velden nodig:

- De Run time tot 1000
- De run time to 1000 factor
- Run time vanaf 1000
- Run time vanaf 1000 factor

Er zit ook validatie op de velden. Indien deze niet of verkeerd zijn ingevuld geeft deze een foutmelding. Een voorbeeld van een foutieve ingave is bijvoorbeeld een letter gebruiker waar een

getal moet komen, of dat de opgegeven naam niet lang genoeg is. Indien alles wel correct is, wordt deze machine met succes aangemaakt. Hij zal nu beschikbaar zijn op de indexpagina en is ook opgeslagen in de database.

Aanmaken

Machine naam:

The field Name must be a string with a minimum length of 3 and a maximum length of 255.

Type:



Calculatie methode:



Departement



Setup time:

The SetupTime field is required.

Run time tot 1000:

The RunTimeTo1000Speed field is required.

Run time tot 1000 factor:

The RunTimeTo1000SpeedStationFactor field is required.

Run time vanaf 1000:

The field RunTimeFrom1000Speed must be a number.

Run time vanaf 1000 factor:

The RunTimeFrom1000SpeedStationFactor field is required.

Screenshot: Machines - De user krijgt foutmeldingen indien hij een foute ingave doet

8.3.3.2 Read

Door op het oog – icoontje te klikken in de machine lijst, krijg je een weergave van de details van de machine.

Machines

Machines				
Machine naam	Calculatie methode	Type	Departement	Actie
Busch Test	StichMain	Stitch	Department 2	
Fold	Always5000Ticks	Fold	Department 3	
I-Stitch	None	Fold	Department 3	

Vorige Volgende

© 2017 - ZwartOpWit

Screenshot: Machines - Druk op het oogje om de machine details te bekijken

Deze worden weergegeven in een overzichtelijke lijst. Indien je wil terugkeren naar de vorige pagina klik je op “terug”.

Lezen

Machine naam	Busch Test
Type	Stitch
Calculatie methode	StichMain
Departement	Department 2
Setup time	2
Run time tot 1000	3
Run time tot 1000 factor	0
Run time vanaf 1000	2
Run time vanaf 1000 factor	3

Terug

© 2017 - ZwartOpWit

Screenshot: Machines – Detail pagina (read)

8.3.3.3 Update

Als je op het potlood icoontje klikt kan je de gegevens van de gekozen machine wijzigen.

Machines

Machines				
Machine naam	Calculatie methode	Type	Departement	Actie
Busch Test	StichMain	Stitch	Department 2	
Fold	Always5000Ticks	Fold	Department 3	
I-Stitch	None	Fold	Department 3	

[Nieuwe machine](#)

Zoeken: [Zoeken](#) | [Terug naar volledige lijst](#)

[Vorige](#) [Volgende](#)

© 2017 - ZwartOpWit

Screenshot: Machines - Druk op het oogje om de machine te wijzigen

Deze heeft dezelfde invoervelden als de Create/Aanmaken pagina. De reeds bestaande gegevens zijn opgehaald en worden ineens ingevuld. Indien de user klaar is met updaten klikt hij onderaan op wijzigen en komt hij terug op de index pagina.



Wijzigen

Machine naam:

Type:

Calculation method:

Departement:

Setup time:

Run time tot 1000:

Run time tot 1000 factor:

RuntimeFrom1000:

RuntimeFrom1000Factor:

[Terug](#) [Wijzigen](#)

© 2017 - ZwartOpWit

Screenshot: Machines – Wijzigen detail pagina

8.3.3.4 Delete

Om een machine te verwijderen druk je op het vuilbak-icoontje op de index pagina. De machine wordt dan verwijderd.

Machines

Nieuwe machine

Zoeken: Zoeken | Terug naar volledige lijst

Machine naam	Calculatie methode	Type	Departement	Actie
Busch Test	StichMain	Stitch	Department 2	
Fold	Always5000Ticks	Fold	Department 3	
I-Stitch	None	Fold	Department 3	

Vorige Volgende

© 2017 - ZwartOpWit

Screenshot: Machines - verwijderen

8.3.4 Zoeken

Je kan een machine makkelijk terugvinden door de naam of een gedeelte hiervan in te geven in de zoekbalk. Deze is niet hoofdletter gevoelig. Wil je daarna terugkeren naar de volledige lijst druk je op "Terug naar volledige lijst".

Machines

Nieuwe machine

Zoeken: Zoeken Terug naar volledige lijst

Machine naam	Calculatie methode	Type	Departement	Actie
Busch Test	StichMain	Stitch	Department 2	
Score test	None	Score	Department 2	
Test Machine Busch busch	None	Busch	Department 3	

Vorige Volgende

© 2017 - ZwartOpWit

Screenshot: Machines - zoeken

8.3.5 Navigeren

Omdat een pagina al snel veel machines kan bevatten hebben we navigatieknoppen voorzien. Met de knoppen "Vorige" en "Volgende" kan je makkelijk naar de volgende pagina. De knop "Vorige" is niet klikbaar op de eerste pagina, de knop "Volgende" is niet klikbaar op de laatste pagina. Het aantal machines dat wordt weergegeven is momenteel ingesteld op 3. Deze kan nog makkelijk aangepast worden in de code.

Machines

Machines				
Nieuwe machine				
Zoeken:		Zoeken	Terug naar volledige lijst	
Machine naam	Calculatie methode	Type	Departement	Actie
Busch Test	StichMain	Stitch	Department 2	
Fold	Always5000Ticks	Fold	Department 3	
I-Stitch	None	Fold	Department 3	

Vorige Volgende

© 2017 - ZwartOpWit

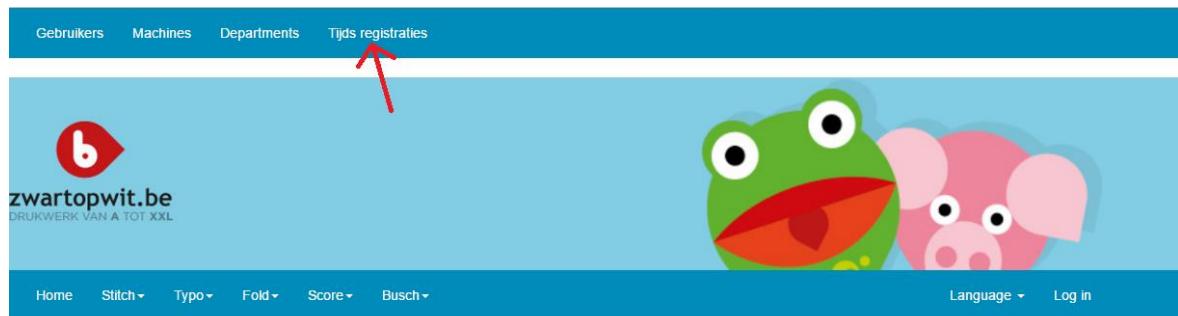
Screenshot: Machines - nавигация по index страницы

8.4 Time registers beheren

8.4.1 Introductie

Als admin-gebruiker kan je time registers (Tijds registratie) beheren.

De optie time registers beheren is enkel zichtbaar in de admin navigatiebalk. Dus enkel als je een admin-user bent is deze beschikbaar.



Screenshot: Navigatiebalk voor admin – Ga naar time registers

8.4.2 Index pagina

Nadat je op de knop “Time registers” in de admin navigatiebalk hebt geklikt, kom je op de index pagina van “Time registers”. Deze geeft een lijst weer van alle huidige departementen.

Hier heb je verschillende mogelijkheden en opties. Voor het beheren van de time registers werken we met CRUD (zie hoofdstuk CRUD). Verder heb je de mogelijkheid om time registers te filteren door gebruik te maken van “Zoeken”. Ook zijn er paginaknopen voorzien voor als de lijst te uitgebreid wordt.

Tijd registraties

[Nieuwe tijds registratie](#)

Zoeken: [Zoeken](#) | [Terug naar voledige lijst](#)

Job id	Gebruiksnaam	Stop tijd	Start tijd	Acties
98		1/05/2017 11:51:25	1/05/2017 11:51:27	
98		1/05/2017 11:15:17	1/05/2017 11:15:19	
98		7/05/2017 14:36:04	7/05/2017 14:36:09	

[Vorige](#) [Volgende](#)

© 2017 - ZwartOpWit

Screenshot: Time registers – Index pagina

8.4.3 Time register CRUD

8.4.3.1 Create

Als we op de knop “Nieuwe tijds registratie” klikken komen we op de create pagina voor een tijds registratie. Hier kunnen we een nieuw tijds registratie aanmaken.

Tijd registraties

[Nieuwe tijds registratie](#)

Zoeken: [Zoeken](#) | [Terug naar voledige lijst](#)

Job id	Gebruiksnaam	Stop tijd	Start tijd	Acties
98		1/05/2017 11:51:25	1/05/2017 11:51:27	
98		1/05/2017 11:15:17	1/05/2017 11:15:19	
98		7/05/2017 14:36:04	7/05/2017 14:36:09	

[Vorige](#) [Volgende](#)

Screenshot: Tijdsregistraties – Nieuwe tijds registratie toevoegen

Een tijdsregistratie bestaat uit een link naar een job, user, start en einddatum. Alle velden zijn verplicht behalve einddatum. Deze kan later nog worden meegegeven.

Aanmaken

Job id:

Gebruikersnaam:

Start tijd:

The Start field is required.

Stop tijd:

The Stop field is required.

[Terug](#)

[Voeg toe](#)

Screenshot: Tijdregistraties – Aanmaken + validatie

8.4.3.2 Read

Door op het oog – icoontje te klikken in de tijdsregistratie lijst, krijg je een weergave van de details van een tijdsregistratie.

Tijd registraties

Tijd registraties				
Nieuwe tijds registratie	Zoeken	Zoeken	Terug naar volledige lijst	
Job id	Gebruiknaam	Stop tijd	Start tijd	Acties
98		1/05/2017 11:51:25	1/05/2017 11:51:27	
98		1/05/2017 11:15:17	1/05/2017 11:15:19	
98		7/05/2017 14:36:04	7/05/2017 14:36:09	

[Vorige](#) [Volgende](#)

Screenshot: Tijdsregistraties - Read

Deze worden weergegeven in een overzichtelijke lijst. Indien je wil terugkeren naar de vorige pagina klik je op “terug”.

Lezen

Job Id	98
Gebruikersnaam	
Start tijd	7/05/2017 14:36:04
StopTime	7/05/2017 14:36:09

[Terug](#)

Screenshot: Tijdsregistraties – Read detail voorstelling

8.4.3.3 Update

Als je op het potlood icoontje klikt, kan je de gegevens van de gekozen tijdsregistratie wijzigen.

Tijd registraties

Nieuwe tijds registratie					
Zoeken:		Zoeken	Terug naar volledige lijst		
Job id	Gebruiksnaam	Stop tijd	Start tijd	Acties	
98		1/05/2017 11:51:25	1/05/2017 11:51:27		
98		1/05/2017 11:15:17	1/05/2017 11:15:19		
98		7/05/2017 14:36:04	7/05/2017 14:36:09		

[Vorige](#) [Volgende](#)

Screenshot: tijdsregistraties - Update

Deze heeft dezelfde invoervelden als de Create/Aanmaken pagina. De reeds bestaande gegevens zijn opgehaald en worden ineens ingevuld. Indien de user klaar is met updaten klikt hij onderaan op wijzigen en komt hij terug op de index pagina.

Wijzigen

Job id:

Gebruikersnaam:

Start tijd:

Stop tijd:

Screenshot: Tijdsregistaties – Wijzigen detail voorstelling

8.4.3.4 Delete

Om een tijdsregistratie te verwijderen druk je op het vuilbak-icoontje op de index pagina. De tijdsregistratie wordt dan verwijderd.

Tijd registraties

Nieuwe tijds registratie				
Zoeken:		Zoeken	Terug naar volledige lijst	
Job id	Gebruikersnaam	Stop tijd	Start tijd	Acties
98		1/05/2017 11:51:25	1/05/2017 11:51:27	
98		1/05/2017 11:15:17	1/05/2017 11:15:19	
98		7/05/2017 14:36:04	7/05/2017 14:36:09	

Screenshot: Tijdsregistraties - Verwijderen

8.4.4 Zoeken

Je kan een tijdsregistratie gemakkelijk terugvinden door de naam of een gedeelte hiervan in te geven in de zoekbalk. Deze is niet hoofdletter gevoelig. Wil je daarna terugkeren naar de volledige lijst druk je op “Terug naar volledige lijst”.

Tijd registraties

Nieuwe tijds registratie

Zoeken: 8 Zoeken Terug naar volledige lijst

Job id	Gebruiksnaam	Stop tijd	Start tijd	Acties
98		1/05/2017 11:15:17	1/05/2017 11:15:19	
98		7/05/2017 14:36:04	7/05/2017 14:36:09	
98		1/05/2017 11:51:25	1/05/2017 11:51:27	

Vorige Volgende

Screenshot: Tijdsregistraties - zoeken

8.4.5 Navigeren

Omdat de mogelijkheid bestaat dat de pagina veel tijdsregistraties bevat, hebben we navigatieknoppen voorzien. Met de knoppen “Vorige” en “Volgende” kan je gemakkelijk naar de volgende pagina. De knop “Vorige” is niet klikbaar op de eerste pagina, de knop “Volgende” is niet klikbaar op de laatste pagina. Het aantal tijdregistraties dat wordt weergegeven is momenteel ingesteld op 3. Deze kan nog gemakkelijk aangepast worden in de code.

Nieuwe tijds registratie

Zoeken: | Zoeken | Terug naar volledige lijst

Job id	Gebruiksnaam	Stop tijd	Start tijd	Acties
98		1/05/2017 11:15:17	1/05/2017 11:15:19	
98		7/05/2017 14:36:04	7/05/2017 14:36:09	
98		1/05/2017 11:51:25	1/05/2017 11:51:27	

Vorige Volgende

Screenshot: Tijdsregistraties – Navigeren door index pagina

8.5 Job planning

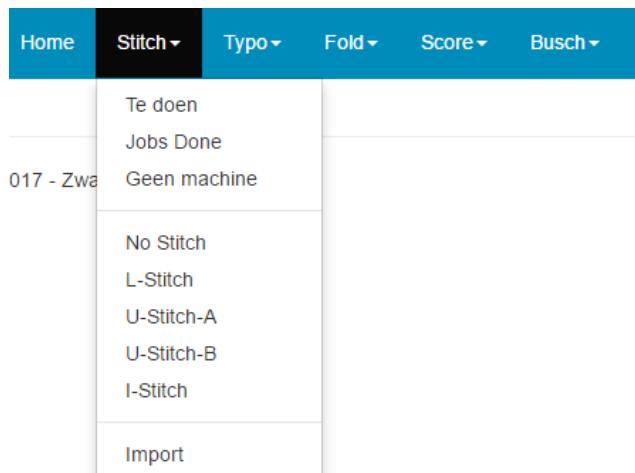
8.5.1 Jobs menu items

Er zijn verschillende pagina's voorzien om job lijnen weer te geven. Deze pagina's worden dynamisch opgebouwd vanuit de controller. Het menu bestaat uit de 5 machine types:

- Stitch
- Type
- Fold
- Score
- Busch

Voor ieder machine type wordt dan de volgende menu structuur gehandhaafd

- To do
- Jobs Done
- No machine
 - Machine 1
 - Machine 2
 -
- Import



Menu voorbeeld stitch

8.5.1.1 To do

Deze pagina gaat voor het machine type dat geselecteerd is alle ingeplande jobs tonen.

Leverdatum:	22-05-2017	Filter datum			
Zoeken op naam:		Zoek	Terug naar volledige lijst		
Jobs voor machine Stitch					
Job nummer	Hoeveelheid	Pagina's	Papier inside	Papieren cover	Machine
851311-01 / Id = 119	100	0	maco silk medium	no cover	U-Stitch-B
852255-02 / Id = 132	500	0	maco silk 250 + MAT cover	no cover	U-Stitch-B
852295-01 / Id = 131	1500	0	maco silk light	no cover	U-Stitch-B
852386-02 / Id = 125	250	0	maco silk 250 + MAT cover	no cover	U-Stitch-B

[Vorige](#) [Volgende](#)
Totale doorlooptijd = 00:32:02

Jobs to do overzicht

8.5.1.2 Jobs done

Deze pagina toont alle jobs voor het machinetype waar het menu onder valt die al in de status "completed" zijn.

Leverdatum: Filter datum

Zoeken op naam: Zoek | Terug naar volledige lijst

Jobs completed Stitch

Job nummer	Hoeveelheid	Pagina's	Papier inside	Papieren cover	Machine	Tijd	Out	Wijzig
851431-01 / Id = 146	1000	0	offset medium	no cover	No Stitch	00:00:00	<input type="button" value=""/>	<input type="button" value=""/>
852386-05 / Id = 122	500	0	maco silk light	no cover	U-Stitch-B	00:00:00	<input type="button" value=""/>	<input type="button" value=""/>

© 2017 - ZwartOpWit

Jobs completed overzicht

8.5.1.3 No machine

Deze pagina toont alle jobs die zijn opgeladen voor het machine type onder het gekozen menu, waarvan die nog niet zijn toegewezen aan een machine.

Leverdatum: Filter datum

Zoeken op naam: Zoek | Terug naar volledige lijst

Jobs Todo Stitch

Job nummer	Hoeveelheid	Pagina's	Papier inside	Papieren cover	No Stitch	L-Stitch	U-Stitch-A	U-Stitch-B	I-Stitch	Out	Wijzig
851311-01 / Id = 119	100	0	maco silk medium	no cover	<input type="radio"/>						
851311-01 / Id = 152	100	0	maco silk medium	no cover	<input type="radio"/>						

Jobs zonder machine

8.5.1.4 Machine detail

Dit zijn de detail pagina's per machine en deze laten alle jobs zien die gereed staan voor een bepaalde machine. Vanuit deze pagina kan ook een job gestart of gestopt worden.

Leverdatum: 22-05-2017

Zoeken op naam: | Terug naar volledige lijst

Jobs voor machine Stitch

Job nummer	Hoeveelheid	Pagina's	Papier inside	Papieren cover	Machine	Tijd	Start Job	Out	Wijzig
851311-01 / Id = 119	100	0	maco silk medium	no cover	U-Stitch-B	00:31:59	<input type="button" value=""/>	<input type="button" value=""/>	<input type="button" value=""/>
852255-02 / Id = 132	500	0	maco silk 250 + MAT cover	no cover	U-Stitch-B	00:00:00	<input type="button" value=""/>	<input type="button" value=""/>	<input type="button" value=""/>
852295-01 / Id = 131	1500	0	maco silk light	no cover	U-Stitch-B	00:00:00	<input type="button" value=""/>	<input type="button" value=""/>	<input type="button" value=""/>
852386-02 / Id = 125	250	0	maco silk 250 + MAT cover	no cover	U-Stitch-B	00:00:00	<input type="button" value=""/>	<input type="button" value=""/>	<input type="button" value=""/>

Totale doorloopijd = 00:32:02

© 2017 - ZwartOpWit

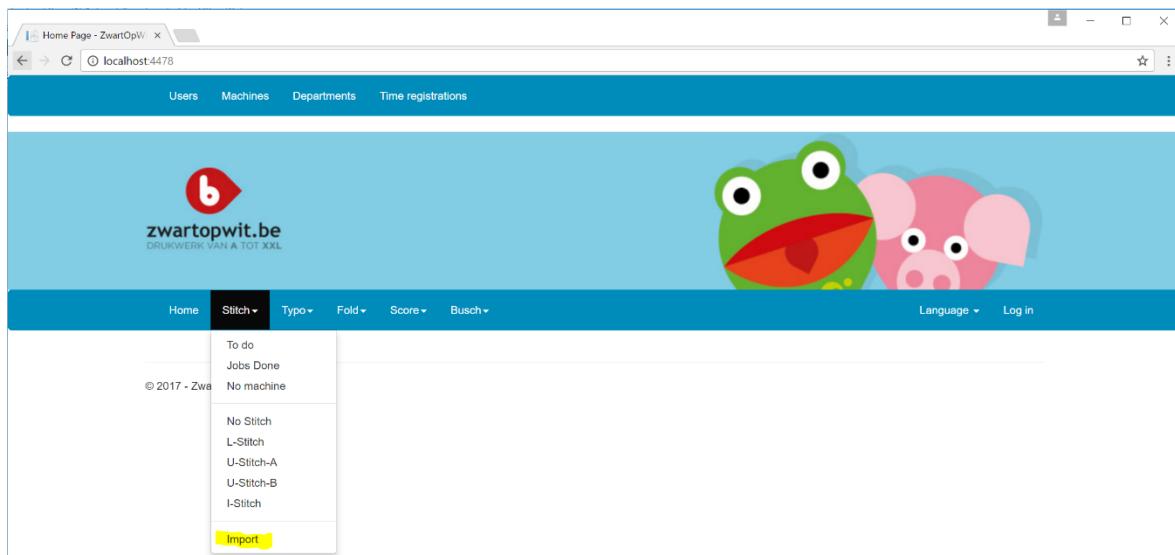
Machine detail overzicht

8.5.1.5 Import

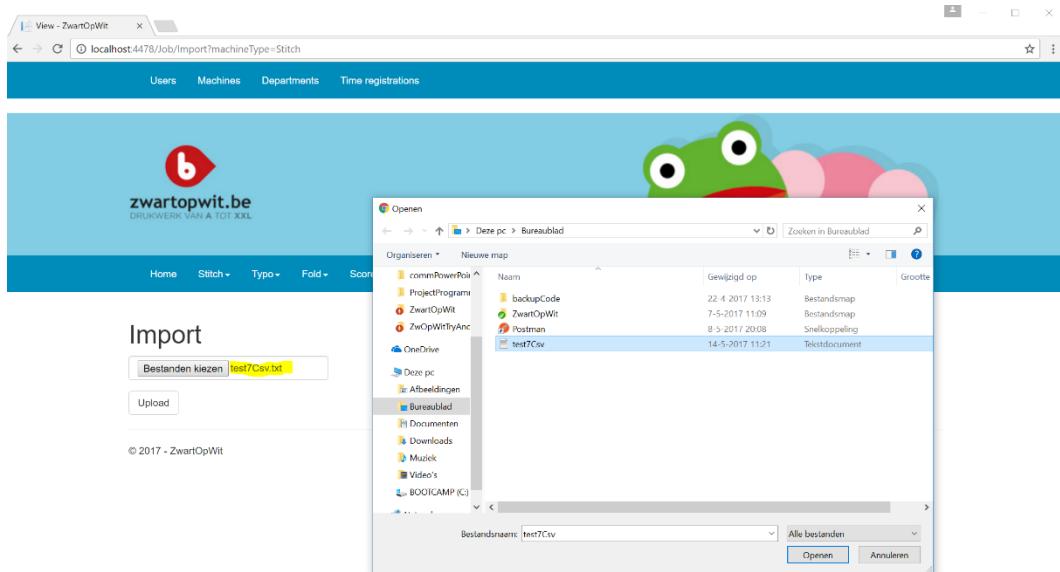
Via het programma Filemaker kan een csv-file gemaakt worden met de gegevens van de jobs. Deze plaats je best op het bureaublad. Via Import kan je het bestand opzoeken en zo halen we de jobs binnen en kopiëren we de gegevens naar de Database. De scheiding tussen jobs en joblines gebeurt hier.

Per machinetype wordt er een andere file aangeleverd. Het machine type wordt automatisch ingevuld afhankelijk van het menu waaruit de import gestart wordt.

Voor een latere uitbreiding wordt hier al de opsplitsing gemaakt tussen jobs en joblines. Een job kan bijvoorbeeld een "stitch" deel en een "fold" deel omvatten. Deze acties behoren tot dezelfde job daarom de opsplitsing.



Selecteer het juiste bestand en druk op Upload.



8.5.2 Jobs starten en stoppen

Vanuit het machine detail scherm en het to do scherm is het mogelijk om jobs te starten en te stoppen. Als er een job gestart word wordt er achterliggend een tijdsregistratie record aangemaakt waarop de begindatum ingevuld is.

Als de medewerker voor stoppen kiest gaat de medewerker moeten kiezen of de job gedaan is of niet. Er zijn verschillende scenario's waardoor 1 persoon de job niet kan afmaken.

8.5.3 Job flow

De job starten gebeurt door te klikken op start job.

Job nummer	Hoeveelheid	Pagina's	Papier inside	Papieren cover	Machine	Tijd	Start Job	Out	Wijzig
851311-01 / Id = 119	100	0	maco silk medium	no cover	U-Stitch-B	00:31:59	<input type="button" value="Start"/>	<input type="button" value="Out"/>	<input type="button" value="Wijzig"/>
851311-01 / Id = 152	100	0	maco silk medium	no cover	No Stitch	00:00:00	<input type="button" value="Start"/>	<input type="button" value="Out"/>	<input type="button" value="Wijzig"/>

Start job vanuit jobs to do

Als de job gestart is komt de medewerker op het scherm uit om de job te stoppen

Job gestart

© 2017 - ZwartOpWit

Stop job

Als er op stop wordt geklikt krijgt de medewerker de vraag of de job klaar is indien ja wordt de job lijn op completed geplaatst anders blijft deze open staan. Het stoppen zorgt ervoor dat de tijdsregistratie een einddatum krijgt.

Job klaar?

© 2017 - ZwartOpWit

Stop job opties

9 Algemeen Besluit

Als algemeen besluit willen we graag concluderen dat we heel wat hebben bijgeleerd. Enerzijds zijn we toch wel trots op het product dat we hebben gerealiseerd na heel wat zwoegen en zweten.

We hebben geleerd dat een goede analyse belangrijk is en dat het op termijn toch wel heel tijdbesparend zou geweest zijn. Iets wat we jammer genoeg iets te laat bij achter kwamen. Dit is iets wat we zeker willen meenemen naar een toekomstig project.

Ook communicatie is belangrijk. Met twee aan één project werken is niet altijd even simpel. Je moet samen conclusies maken en zien dat je op dezelfde golflengte zit. We zijn hier als een sterk team aan de slag gegaan, dat elkaar ondersteunde en aanvulde waar nodig.

Het uitwerken van een dergelijk project vraagt organisatie en inspanning. We kwamen tot de conclusie dat het eigenlijk nooit af is, en dat je je verder in kan blijven verdiepen.

Algemeen kunnen we besluiten dat we fier zijn op onze planningstool. Deze vergt nog wel wat werk en inzet, maar geeft een mooie basis om in de toekomst verder uit te breiden. Dit rekening houdend met zijn specifieke noden en wensen.

10 Bijlagen

1. Plan van aanpak

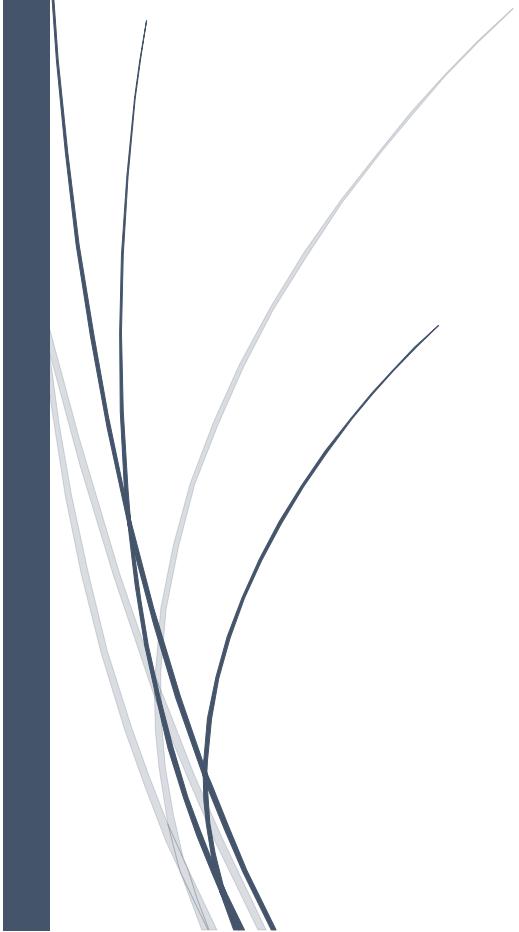


18-11-2016

Plan van Aanpak

Tool voor planning en afwerking drukwerk

www.zwartopwit.be



Kathleen Jansen & Jo Brunau

Inleiding

Dit plan van aanpak is opgemaakt in het kader van de implementatie van een nieuwe tool voor planning van afwerking van drukwerk bij zwartopwit.be / drukkerij Bulckens.

Probleemstelling

Bestellingen van drukwerk komen binnen via de website. Nadat deze bestellingen zijn ontvangen worden zij verwerkt. De vertaling van het bestelde pakket drukwerk naar de werkvlloer toe, is echter in de huidige omgeving erg complex georganiseerd.

Op een vlottere en lean-gerichte manier te werk kunnen gaan, is er nood aan een programma dat de planning en verwerking van de verschillende drukwerken doorheen de afwerking van de drukkerij op elk moment kan volgen.

Business and Functional Requirements

- Systeem moet op basis van externe Filemaker database gegevens importeren in een eigen database.
- Gegevens in de database moeten kunnen bijgewerkt worden.
- Systeem moet a.d.h.v. de gegevens in de database geschatte werktijden kunnen berekenen.
- De applicatie moet beschikken over een veilig identificatieproces.
- Er moet een onderscheid gemaakt kunnen worden tussen medewerker en verantwoordelijke.
- Nieuwe gebruikers moeten kunnen worden aangemaakt.
- Gebruikers moeten planplaats kunnen kiezen.
- Systeem moet drukwerken aan de juiste planplaats toewijzen.
- Gebruikers moeten af te werken drukwerk kunnen kiezen volgens levertijd en bestemming.
- Systeem moet werktijden registreren.
- Systeem moet onderbrekingen registreren.
- Gebruikers moeten opmerkingen kunnen toevoegen aan elke tijdsregistratie.
- Systeem moet alle verzamelde gegevens naar de database schrijven en linken aan de gekozen drukwerken
- Systeem moet in realtime kunnen tonen welke gebruikers, welk drukwerken hebben opgestart.
- Systeem moet queries kunnen uitvoeren op de database om nacalculaties uit te voeren volgens werknemer en drukwerk

Design Requirements

- De applicatie zal in eerste instantie standalone draaien. Later kan nog bekijken worden of verweven in de Filemaker database tot de mogelijkheid behoort.
- De applicatie dient geschreven te worden in .NET
- Het resultaat is een web-applicatie.

- De database wordt een MySQL database.

Technologie

Doordat er in het huidige IT-landschap al gebruik wordt gemaakt van Linux based services (Ruby) en er een MySQL database beschikbaar is de keuze gevallen om technologieën in deze lijn te gebruiken. De data die door de planning tool gebruikt kan teruggevonden worden in Filemaker.

- Asp.net Core c#
- MySQL
- Filemaker
- Razor

Kostprijs

Ontwikkelingskosten:

- Deze planningstool wordt uitgewerkt a.d.h.v. een schoolproject waardoor deze 0 euro bedraagt.

Onderhoudskosten

- Om de planningstool te laten draaien worden de kosten geschat op 0 euro. Dit omdat we draaien op de bestaande infrastructuur gaan er geen extra licentie en hardware kosten zijn.
- Op termijn kan er een investering nodig zijn in extra hardware naarmate de applicatie groeit.

Planning

Er is gekozen om Agile te gaan werken. Dit omwille van volgende redenen:

- Flexibel opleveren
- Makkelijk omgaan met scope uitbreiding
- Alles wordt in hapklare delen opgeleverd voor einduser

Jaar	2016							2017																		
Week	47	48	49	50	51	52	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Analyse																										
Sprint 1																										
Sprint 2																										
Sprint 3																										
Sprint 4																										
Afronding																										

Verwachte sprintplanning:

Analyse:

- Het analyseren van de planningtool + uitwerking userstories

- Uitwerken van proof of concept (testen of het systeem goed kan communiceren met andere systemen in het IT-landschap)

Sprint 1:

- Uitwerken van basis van de user-interface
- Ophalen van data uit filemaker
- Uitwerken van form-templates
- Authenticatie en autorisatie (login)

Sprint 2:

- Uitwerken van de planningtoolschermen
- Uitwerken van planning engine

Sprint 3:

- Te bepalen na analyse

Sprint 4:

- Te bepalen na analyse

Afronding:

- Bugfixing
- Detail afhandeling

UAT

User Acceptance Testing. Voor alles live gaat in productie moeten de key users uitgebreid de tool testen en goedkeuring geven. Alle testen worden beschreven in een te bepalen testing tool. Na een testcycli wordt er een document met de resultaten verdeeld.

Op te leveren:

- End-to-end test-scenario's
- Go-live approval

Training

Training bestaat uit het verzamelen/produceren van trainingsmateriaal en het geven van de effectieve training aan users. Enkel key users worden getrained. Deze worden verwacht hun kennis over te dragen naar de eindgebruikers (Train the trainer).

Op te leveren:

- Trainingsmateriaal
- Getrainde key users

11 Bronnen

WARMER, J. en KLEPPE, A., Praktisch UML 4^e Editie, 2^e druk, Pearson Education Benelux, Amsterdam, 2008, 253 pagina's

Microsoft (2017), MVC ASP.NET, www.asp.net/mvc

W3schools (2017), ASP.NET Razor, https://www.w3schools.com/asp/razor_intro.asp

Damienbod (2016), ASP.NET Core with MySQL and Entity Framework Core,
<https://damienbod.com/2016/08/26/asp-net-core-1-0-with-mysql-and-entity-framework-core/>

Microsoft Docs (2017), View Components, <https://docs.microsoft.com/en-us/aspnet/core/mvc/views/view-components>

Code Project/Rahul Rajat Singh, R.R.S. (2012), An Introduction to Entity Framework,
<https://www.codeproject.com/Articles/363040/An-Introduction-to-Entity-Framework-for-Absolute-B>

Microsoft (2017), Introduction to Entity Framework, [https://msdn.microsoft.com/en-us/library/aa937723\(v=vs.113\).aspx](https://msdn.microsoft.com/en-us/library/aa937723(v=vs.113).aspx)

Microsoft (2017), Introduction to Identity, <https://docs.microsoft.com/en-us/aspnet/core/security/authentication/identity>

VAN DE PERRE L., Source Control met Git (2016), 16 slides

Options pattern <https://andrewlock.net/how-to-use-the-ioptions-pattern-for-configuration-in-asp-net-core-rc2/>

Microsoft Docs (2017), Globalisation and Localization, <https://docs.microsoft.com/en-us/aspnet/core/fundamentals/localization>

GitHub (2017), Github, <https://github.com>

Bootstrap (2017), Bootstrap, <http://getbootstrap.com/>

.net core policies (2017), Turotorial: Policy-based Authorization in ASP.NET Core,
<https://stormpath.com/blog/tutorial-policy-based-authorization-asp-net-core>

Roles setup (2017), Roles Setup <http://stackoverflow.com/questions/39934201/asp-net-core-identity-add-custom-user-roles-on-application-startup>

Option pattern (2017), Configurations in .net Core, <https://docs.microsoft.com/en-us/aspnet/core/fundamentals/configuration#using-options-and-configuration-objects>