Professionele Bachelor Toegepaste Informatica

JIDOKA.
face the future

JIDOKA RECRUITMENTTOOL

Michiel Slechten

Promotoren:

Paula Kozanecka
Greta Poelmans

JIDOKA
Hogeschool PXL Hasselt

Bachelorpaper Academiejaar 2017-2018

Professionele Bachelor Toegepaste Informatica

JIDOKA.
face the future

JIDOKA RECRUITMENTTOOL

Michiel Slechten

Promotoren:

Paula Kozanecka — JIDOKA
Greta Poelmans — Hogeschool PXL Hasselt

Bachelorpaper Academiejaar 2017-2018

Dankwoord

Graag zou ik eerst JIDOKA, in het bijzonder Paula Kozanecka, bedanken voor de kans om mijn stage in zo'n leerrijke omgeving uit te mogen voeren. Ook iedere collega zou ik graag bedanken voor de hulp en tips op ieder moment van mijn stage en voor de gemoedelijke werksfeer. Vervolgens ook een bedankje aan medestudent Toon Froeyen met wie ik deze stage heb uitgevoerd. Zijn hulp en geduld hebben ook enorm bijgedragen aan de totstandkoming van dit project.

Ook bedank ik graag mijn stagementor Greta Poelmans; op ieder moment stond zij klaar om mijn vragen te beantwoorden en steun te bieden waar deze nodig was.

Mijn ouders en mijn vriendin bedank ik ook, voor het geduld en de steun die ze mij boden tijdens het schrijven van mijn eindwerk.

Abstract

JIDOKA is een opkomend bedrijf dat voortdurend op zoek is naar nieuwe medewerkers om hun team te versterken. Veel recruitmentgerelateerde data wordt momenteel bijgehouden op verschillende platformen en services (Spreadsheets, Google Drive, JIRA), wat het recruitmentproces voor JIDOKA onnodig kan compliceren.

De opdracht omvat het opzetten van een online webapplicatie die het recruitmentproces binnen JIDOKA centraliseert en vereenvoudigt. Op deze manier kunnen vacatures snel en efficiënt ingevuld worden. Ook zal er een pool van interessante kandidaten bijgehouden en beheerd kunnen worden. Deze pool geeft JIDOKA de mogelijkheid kandidaten te filteren en hier analyses op uit te voeren (bv. het aantal sollicitanten per vacature). Deze webapplicatie zal overigens in verbinding staan met de JIDOKA-website, zodat sollicitanten onmiddellijk hun gegevens kunnen insturen voor een openstaande vacature en de recruitmentflow optimaal gevolgd kan worden. De applicatie behandelt deze flow volledig: van het insturen van gegevens en een CV, door een kandidaat bij de initiële sollicitatie, tot het bijhouden van besprekingen en feedback van gesprekken en het uiteindelijk verwelkomen van de kandidaat in het JIDOKA-team.

De frontend van de applicatie is ontwikkeld met Angular 4. De backend ontstaat in Spring Boot. Voor de opslag van data wordt een PostgreSQL docker server gebruikt. Mijn taak bestaat erin deze applicatie te helpen opstellen, maar ook constructief mee te werken aan de analyse die moet gebeuren om het project tot stand te brengen.

Het onderzoek, ten slotte, bestaat uit een studie tussen de verschillende analysetechnieken Story Mapping en Event Storming. De huidige analysetechniek om de requirements en flow van een project in kaart te brengen is Story Mapping. Vanuit een groeiende interesse in Event Storming worden de verschillen en de voor- en nadelen tussen deze twee in kaart gebracht.

Inhoudsopgave

Lijst van gebruikte figuren en tabellen

Lijst van gebruikte afkortingen en begrippen

Recruitmentproces	Volledige proces van het zoeken naar en aanwerven van nieuwe werknemers
Pool	Grote lijst van contactgegevens
Consultancy	Het geven van deskundig advies binnen een bepaald gebied
Agile	Verzameling van principes en methodes om een product op te leveren
Release	De uitgave of het opleveren van een product of een deel van een product
Retrospective	Terugblik op en evaluatie van uitgevoerde handelingen
HR	Human Resources; Management van werknemers
Flow	Doorlopen van een cyclus of proces in een bepaalde volgorde
Product Owner	De belangrijkste stakeholder; Bepaalt hoe het product moet functioneren
Frontend	Alles wat de gebruiker ziet van de applicatie
Backend	Het achterliggende deel dat ervoor zorgt dat de site werkt
Repository	Opslagplaats die software bevat
Story	Beschrijft hoe een functionaliteit of feature in een applicatie gebruikt hoort te worden door een gebruiker
Mergen	Samenvoegen
Pull Request	Aanvraag tot samenvoegen van code op de master branch
Pluralsight	Online leerplatform dat toegang biedt tot cursussen
Multipartfile	Representatie van een geüploade file
Requirement	Een vereiste staat van een functionaliteit voor de klant

Backlog	Verzameling van nog uit te voeren werk
Minimum Viable Product (MVP)	Deel van het product met het minst aantal functionaliteit en meeste waarde
Workspace	Plaats of opslag waar werk plaatsvindt.
Framework	Abstracte omgeving van softwarecomponenten en klassen dat gebruikt wordt bij het programmeren van applicaties
Backbone	Basis of leidraad van een systeem
In-depth	Diepgaand; Uitgebreid
Stakeholders	Alle belanghebbende personen die baat hebben bij het product
Template	Document dat al voorzien is van details die aangepast kunnen worden
Happy Path	Standaardscenario waarbij geen fouten of errors optreden
Exception	Fout die optreedt door een falen in het systeem
Hardcoded	Data of parameters in een applicatie die zo geprogrammeerd zijn dat ze niet aangepast kunnen worden zonder de applicatie te wijzigen
(Software) Library	Verzameling van code dat gebruikt wordt bij het programmeren van software

Inleiding

Tijdens mijn stageopdracht werk ik een applicatie uit die het volledige recruitmentproces van JIDOKA zal centraliseren en vereenvoudigen. Voor de aanvang van mijn stage werd informatie over kandidaten bijgehouden in verschillende vormen op verschillende platformen. Het proces dat een kandidaat doorloopt kan op die manier onoverzichtelijk worden. Deze stageopdracht biedt daarvoor een oplossing.

Deze stageopdracht zal een omgeving creëren waarin een pool van potentiële werknemers en kandidaten ontstaat.

Het onderzoek bij deze stage neemt de technieken Story Mapping en Event Storming onder de loep. Beide zijn technieken de gebruikt worden voor het analyseren van een nieuw of bestaand project. Story Mapping is een begrip dat voor de meeste wel bekend in de oren klinkt. Event Storming is een nieuw gegeven dat stilaan meer gebruikt wordt bij het realiseren van grote projecten maar het kent nog geen groot publiek. Aan de hand van een literatuurstudie en concrete cases hoop ik een onderzoek te kunnen vormen waarin het gebruik en verschil van beide technieken duidelijk wordt.

Mijn verwachtingen voor deze stage zijn hoog. JIDOKA is een bedrijf waarvan ik alleen maar goede zaken heb gehoord. Met een goed geëngageerd team en de visie volledig op de toekomst ben ik zeer gemotiveerd mezelf aan deze opdracht te zetten en zo veel mogelijk bij te leren.

I. Stageverslag

1 Bedrijfsvoorstelling

JIDOKA is een bedrijf dat zich specialiseert in zowel het uitwerken van softwarematige oplossingen als het ondersteunen of begeleiden van externe projecten. Dit door het bieden van consultancy aan bedrijven die klaar zijn voor de volgende stap, het realiseren van business ideeën in de vorm van uitstekend werkende software of het professioneel ondersteunen van bestaande projecten zodat een gewenst eindresultaat tot stand kan komen.

De factor die JIDOKA onderscheidt van de competitie is de menselijke factor die JIDOKA introduceert bij de uitvoer van deze processen. Het geven van vrijheid aan werknemers om hun eigen werkwijze te bepalen creëert een zeker bewustzijn dat ervoor zorgt dat er, met de focus op de wensen van de klant, een optimale output gecreëerd wordt.

JIDOKA werkt zijn projecten uit volgens het agile principe: tussentijdse releases, retrospectives en dagelijkse stand-ups zorgen ervoor dat zowel het team als de klant continu op de hoogte is van ontwikkelingen binnen het project.

De stageopdracht die uitgewerkt zal worden in dit eindwerk situeert zich volledig in de IT-afdeling met als uiteindelijke doel te functioneren in de HR-afdeling.

JIDOKA is gelegen in België waar het momenteel 2 vestigingen heeft: een kantoor in Hasselt en een in Mechelen.

2 Voorstelling stageopdracht

De vraag naar deze stageopdracht kwam vanuit de HR-afdeling. Het volledige recruitmentproces van JIDOKA werd voor aanvang van onze stage gespreid op verschillende platformen zoals bv. Google Spreadsheets/Documenten op de Google Drive. Dit zorgde vaak voor onoverzichtelijke en onhandige situaties waarbij verschillende documenten samen gepuzzeld moesten worden om informatie over kandidaten terug te vinden.

Mijn stageopdracht is het uitwerken van een applicatie die dit proces volledig zal centraliseren. De flow die een kandidaat doorloopt om een vacature in te vullen zal gebeuren via de nieuwe recruitmenttool. Op deze manier kan er veel efficiënter en overzichtelijker gewerkt worden bij het aanwerven van nieuwe talenten bij het JIDOKA-team.

De uiteindelijke klant van ons project zal dus JIDOKA zijn, zij zullen zelf intern gebruik maken van de verwezenlijkte applicatie. Specifieker zal de HR-afdeling het meeste baat hebben bij dit project, zij zijn dan ook de product owner.

Het uiteindelijke doel is het realiseren van een applicatie die in verbinding staat met de JIDOKA website. Vanuit deze applicatie zullen vacatures online gezet kunnen worden. Kandidaten kunnen op deze vacature solliciteren waarbij hun gegevens opgeslagen worden in de applicatie. De verdere werking van het recruitmentproces kan vervolgens volledig opgevolgd en uitgevoerd worden in deze omgeving.

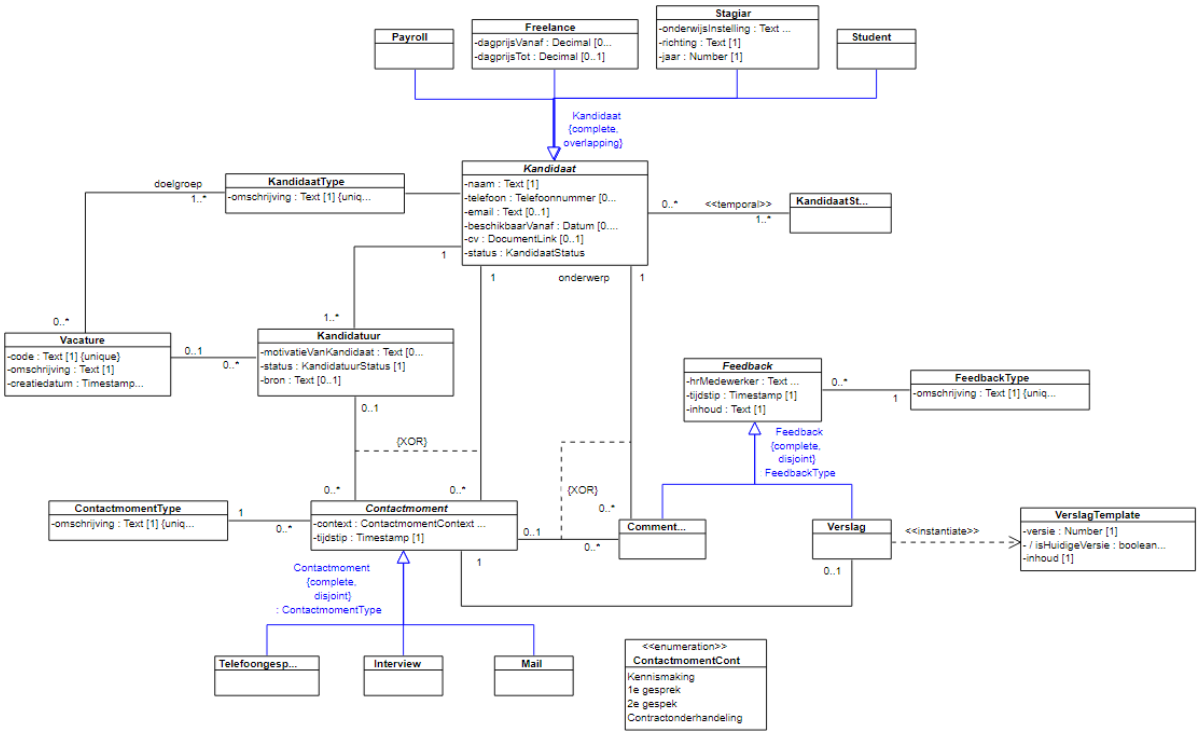
De frontend van mijn opdracht is opgebouwd in Angular 4. De backend in Spring Boot. De benodigde data wordt verwerkt in een PostgreSQL docker server.

3 Uitwerking Stageopdracht

3.1 Start

Voor het starten van deze stageopdracht werd er toegang verleend tot enkele platformen waarmee JIDOKA werkt. Een van de belangrijkste hiervan is JIRA, een projectmanagementtool die toelaat om de workflow van een project volledig te managen en visualiseren[1]. Slack werd ook gebruikt, dit is een tool om de communicatie binnen het bedrijf te verzorgen[2]. Ten slotte kregen we ook toegang tot Bitbucket, waarschijnlijk een van de belangrijkste tools aangezien deze dient voor het onderhouden van de versiecontrole bij het uitwerken van deze applicatie[3].

Bij aanvang van de stage was het domeinmodel grotendeels al opgesteld, enkele story's waren op JIRA ook al uitgeschreven zodat onmiddellijk gestart kon worden met het programmeren van de applicatie. In figuur 1 staat het domeinmodel dat alle domeinen en domeinconcepten bevat die bij deze applicatie komen kijken.



Figuur 1 - Domeinmodel

De volledige code van dit project werd vervolgens geschreven in de IntelliJ IDEA-ontwikkelomgeving.

Na het opzetten van de ontwikkelomgeving konden we onmiddellijk starten met het coderen. Omdat Angular 4 en Spring Boot volledig nieuw waren was het eerst nodig om Pluralsight courses te volgen. Met hulp van deze cursussen en de kennis van collega's op Jidoka werd er relatief snel gestart met het schrijven van nieuwe functionaliteiten voor de applicatie, op een tempo van slechts één story op een week, tot drie à vier story's per week afgewerkt en goedgekeurd.

3.2 Eerste weken

Aanvankelijk waren er enkele struikelblokken, zoals het uploaden van files. Bij het insturen van zijn sollicitatie moet een kandidaat uiteraard de mogelijkheid hebben om zijn CV mee te verzenden. Zo'n file upload kon gerealiseerd worden door met een multipartfile te werken, wat ook aangeraden werd binnen JIDOKA. Op deze manier kon de inhoud van een CV opgeslagen worden als een bytearray type in de database.

Voor deze applicatie was het ook nodig een inlogsysteem te voorzien. Verschillende gebruikers moesten toegang kunnen krijgen tot de recruitmenttool via hun eigen profiel. Hiervoor is het OAuth2 framework[4] gebruikt. Aangezien dit een relatief nieuw framework is was hierover vrij weinig documentatie terug te vinden Een uitgebreidere uitleg over dit onderwerp is terug te vinden in het eindwerk van Toon Froeyen.

Aangezien als Software Manager de werking van GIT niet echt aan bod komt, was dit een enorm pijnpunt. Enkele problemen die we zo ondervonden met het werken van git was bv. Het coderen op de verkeerde branch of het mergen van verkeerde branches met elkaar, waardoor stukken code verloren gingen. Naast deze problemen is het verloop van de eerste weken relatief vlot gegaan. We konden al snel enkele functionaliteiten uitbouwen, zoals het toevoegen van een kandidaat.

Tijdens de dagelijkse stand-up konden we problemen signaleren en met wat hulp van de collega's kregen we die ook snel opgelost.

3.3 User Story's

Na de realisatie van de eerste functionaliteiten voor het project worden er nieuwe story's uitgewerkt. Deze story's worden opgesteld in de JIDOKA JIRA omgeving en telkens met een vaste template. Deze template bestaat uit enkele headers die de opbouw van zo'n story definiëren.

Figuur 2 - Aanmaken van een story

Zo begint een story met de rationale: in één zin moet de waarde van de nieuwe functionaliteit duidelijk zijn voor de klant. Het verwoorden van de rationale is vaak moeilijk, maar ook belangrijk: deze ene zin kan namelijk en feature maken of kraken voor de klant.

Figuur 3 - Rationale van een story

De context geeft een kort overzicht van waar deze feature zich bevindt in de applicatie. Dit zorgt ervoor dat de developers zeker niet in de war geraken en duidelijk weten waar deze story zich afspeelt. Ook de domeinconcepten die voorkomen bij het implementeren van de nieuwe functionaliteit worden in de context opgenomen. Zo kan de developer teruggrijpen naar het domeinmodel waar de relaties tussen verschillende concepten makkelijk terug te vinden zijn.

Figuur 4 - Context van een story

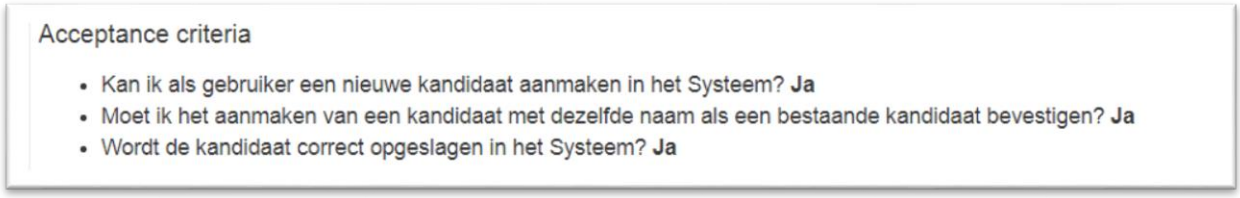
Vervolgens beschrijven we in de flow de interactie die zal plaatsvinden tussen het systeem en de gebruiker. Hier wordt altijd het happy path beschreven. De happy path is de opeenvolging van interacties tussen systeem en gebruiker waarbij er geen fouten opduiken. Dit is de optimale samenwerking tussen de twee zoals de functionaliteit hoort te werken.

Figuur 5 - Flow van een story

Na deze flow worden de exceptions opgenoemd. Dit zijn enkele problemen die kunnen optreden wanneer de gebruiker niet voldoet aan de eisen van de nieuwe functionaliteit, bv. door het leeg laten van een verplicht veld. De mogelijke fouten die een gebruiker kan maken worden uitgeschreven maar ook hoe het systeem hier op moet reageren, aan de hand van foutmeldingen of dergelijke.

Figuur 6 - Exceptions van een story

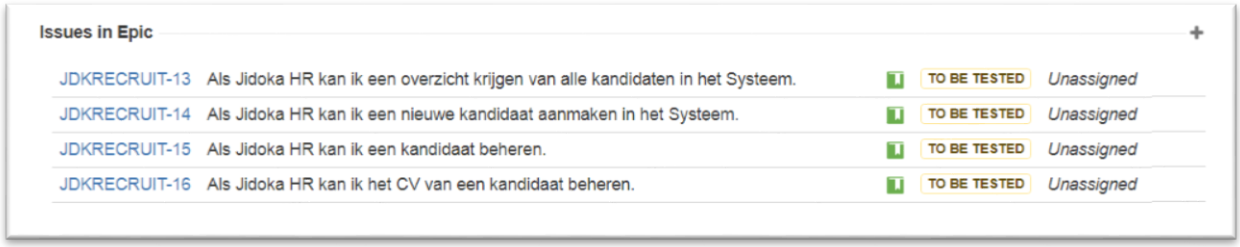
Uiteindelijk worden de acceptance criteria toegevoegd. Dit is een definitieve checklist waarmee de developer kan toetsen of zijn stuk code voldoet aan de eisen van de functionaliteit. De acceptance criteria worden altijd opgesteld d.m.v. ja-neevragen.



Figuur 7 - Acceptance criteria van een story

Na een review door een analist op JIDOKA en het verwerken van de nodige commentaren is de story klaar om geïmplementeerd te worden in de applicatie.

Een story op zich behandelt meestal maar een klein onderdeel van een groter deel van de applicatie. Om story's te bundelen volgens het onderdeel waarvan ze deel uitmaken kunnen we in JIRA epics aanmaken. Een Epic is eigenlijk een grote story opgesplitst in allerlei kleine story's. Een groot onderdeel kan het beheer van kandidaten zijn. Dit kan op zijn beurt opgesplitst worden in het tonen, aanmaken en aanpassen van een kandidaat.



Figuur 8 - Epic

3.4 Git Flow

Wanneer zo'n story omgezet wordt naar effectieve code wordt er gewerkt met git, via bitbucket. Git is een online repository waarmee de versiecontrole van de applicatie gevoerd wordt. De stabiele versie van de applicatie is altijd terug te vinden op de development branch.

Bij het starten van coderen van een nieuwe functionaliteit zullen we een branch aanmaken vanuit de development branch. Deze nieuwe branch vernoemen we naar het nummer van de story waaraan gewerkt wordt, bv. 'feature/JDKRECRUIT-15'. Op deze branch kan gecodeerd en getest worden zonder dat de applicatie direct aangepast wordt.

Bij het afwerken van deze functionaliteit maken we via bitbucket een pull request aan. Andere developers bekijken deze request en lezen de geschreven code na op foutjes. Er is hier ook ruimte om tips te geven waardoor de code optimaal opgesteld kan worden. Deze opmerkingen worden dan zo snel mogelijk terug verwerkt in de code tot deze uiteindelijk goedgekeurd wordt.

Eens de code goedgekeurd is kan de branch gemerged worden met de develop branch. Dit betekent dat de feature samengesmolten wordt met de stabiele versie van de applicatie. Onze applicatie is nu uitgebreid.

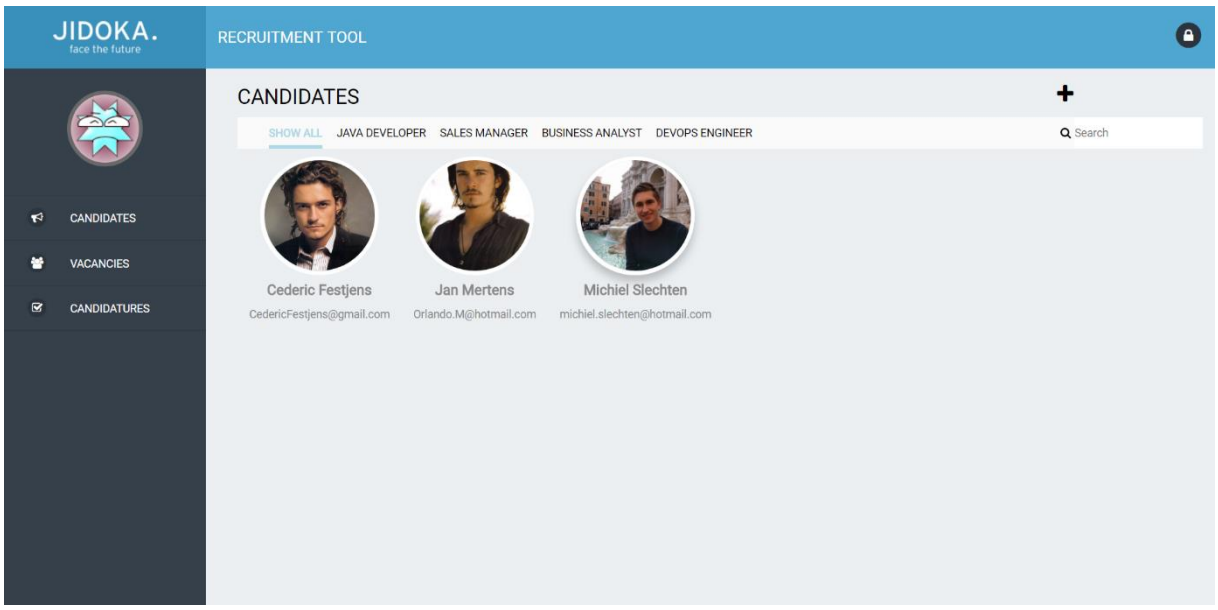
Wanneer de applicatie zich uiteindelijk bevindt in een staat waarop hij gereleaset kan worden naar het publiek of de klant, zal de applicatie op de master branch gezet worden. Op de master branch is de applicatie klaar voor productie.

Figuur 9 - Git Flow

3.5 Applicatie

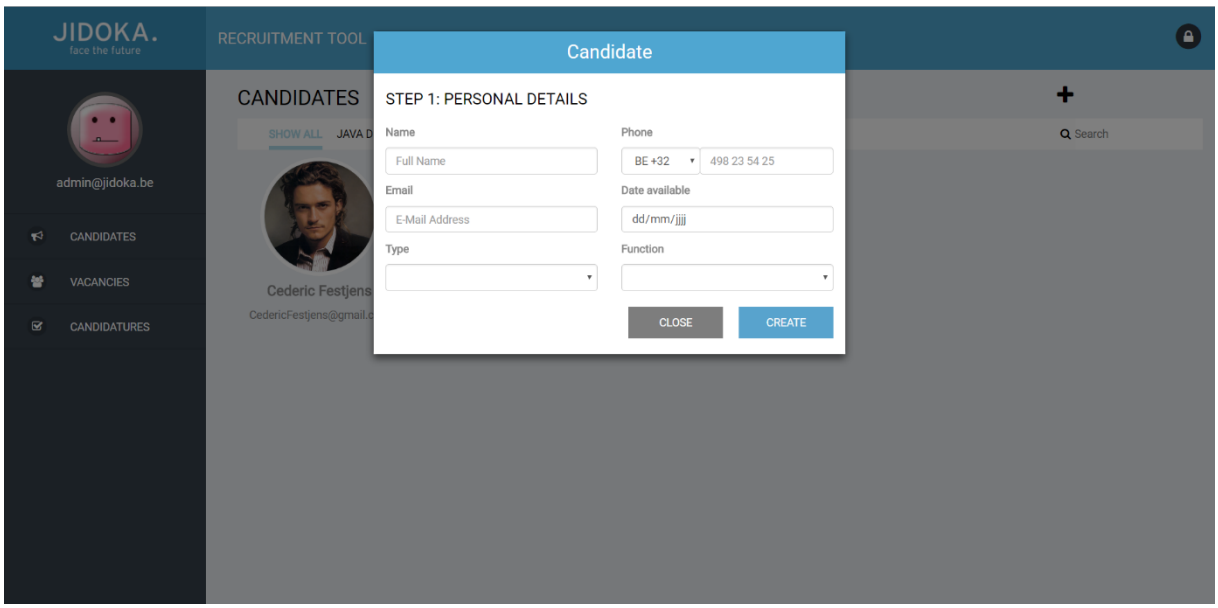
Tijdens de stageperiode werd er tot een resultaat bekomen dat enigszins gezien mag worden. De opdracht werd zo goed mogelijk uitgevoerd. Bij het afsluiten van de stage is de applicatie bruikbaar voor de HR-afdeling.

Na een klein loginscherm, waarvan de logingegevens wel nog hardcoded in de database staan, opent het overzicht van de kandidaten zich. Hierin worden alle kandidaten weergeven die bekend zijn in het systeem met hun foto, naam en e-mailadres. Het maakt niet uit of de kandidaten gesolliciteerd hebben voor een vacature. Op deze pagina kan er gefilterd worden op functie, bv. 'Java Developer', of kan er gezocht worden op naam van de kandidaat.



Figuur 10 - Overzicht kandidaten

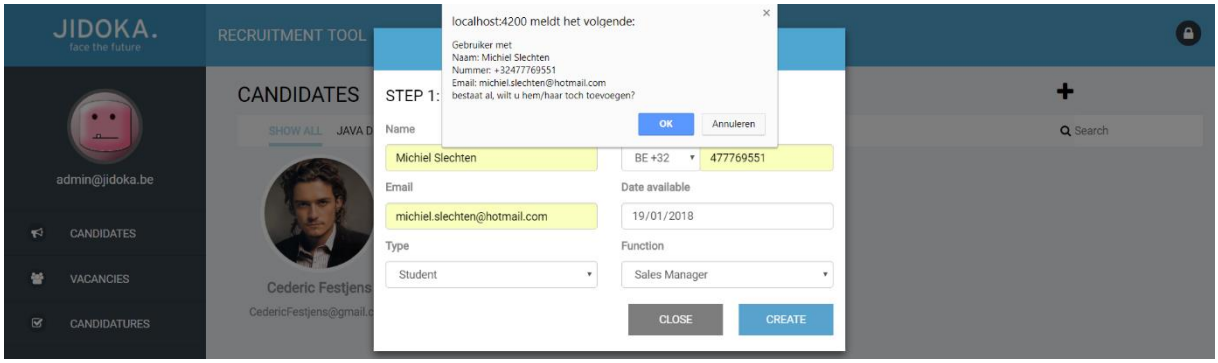
Er kunnen ook nieuwe kandidaten aangemaakt worden. Dit gebeurt door te klikken op de grote plus die zich rechtsboven op het scherm bevindt. De volgende pop-up zal vervolgens getoond worden:



Figuur 11 - Aanmaken kandidaat

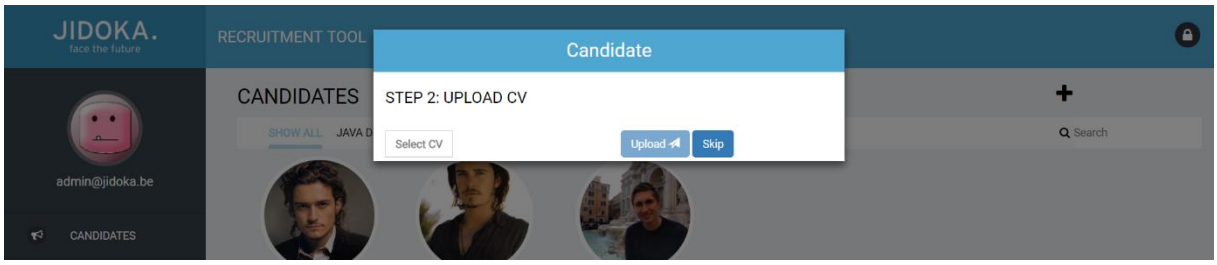
Om zo'n pop-up te tonen werd er gebruik gemaakt van Modals. Dit is een library voor Angular dat toelaat om dialoogschermen op te bouwen op een efficiënte maar zeker elegante manier[5].

In deze pop-up wordt gevraagd om de persoonlijke gegevens van de kandidaat in te geven. Elk van deze gegevens zijn verplicht, behalve de 'Date available', deze mag leeg gelaten worden. Wanneer één verplicht veld niet ingevuld wordt zal het niet mogelijk zijn om op de 'Create' knop te klikken. Indien er een kandidaat aangemaakt wordt die reeds bestaat in het systeem zal er een melding getoond worden die vraagt of de ingevoerde gebruiker toch opgeslagen moet worden.



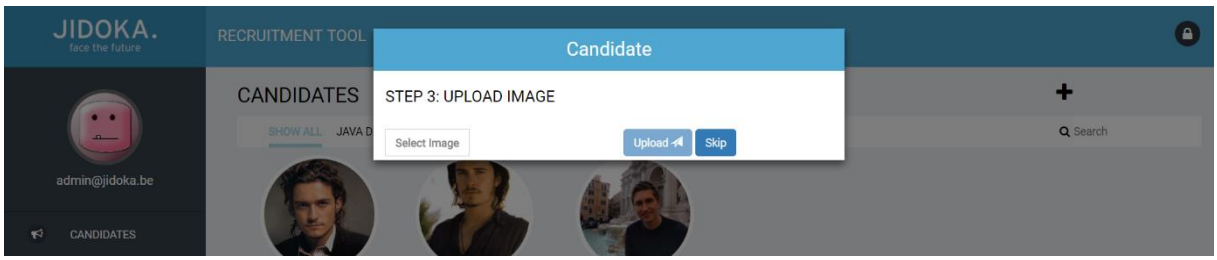
Figuur 12 - Melding bestaande kandidaat

Bij het klikken van 'Create' wordt er gegaan naar de volgende stap. Op stap twee zal de HR-werknemer een CV van de kandidaat kunnen uploaden. Dit is niet verplicht, maar zeker wel een handigheid. Indien er geen CV geüpload wordt, kan er later nog een toegevoegd worden via het detail van een kandidaat. Er kunnen enkel PDF bestanden opgeslagen worden in het systeem.



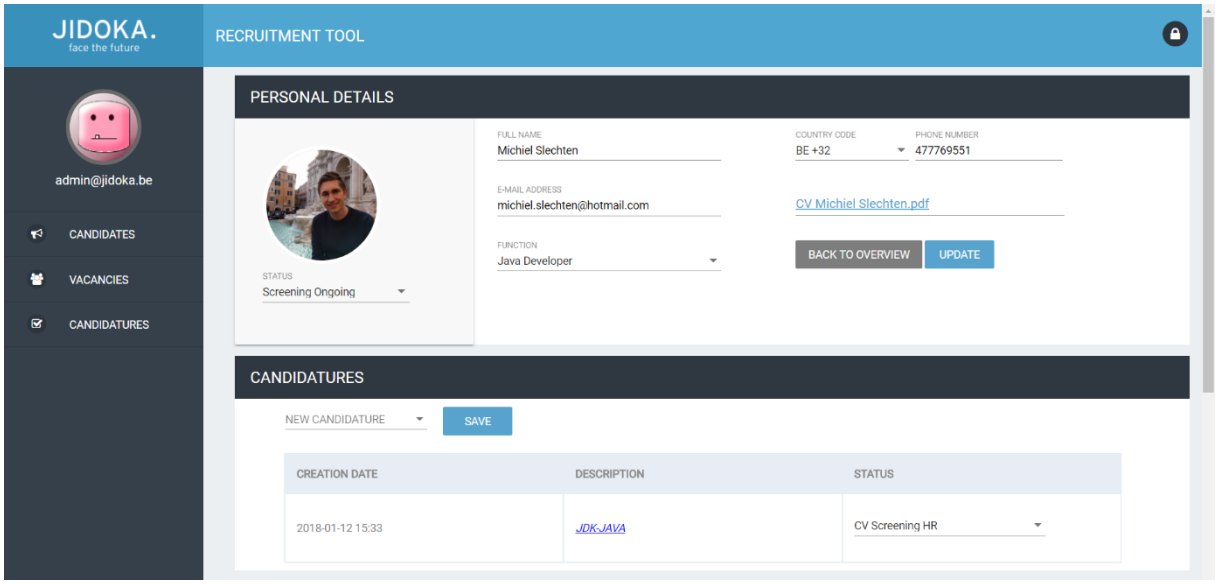
Figuur 13 - Upload CV

Als laatste stap voor het toevoegen van een kandidaat zal het systeem om een afbeelding van de kandidaat vragen. Dit is ook niet verplicht en een afbeelding kan later ook nog toegevoegd of gewijzigd worden. Enkel bestanden met een .jpg of .png extensie worden hier toegelaten.



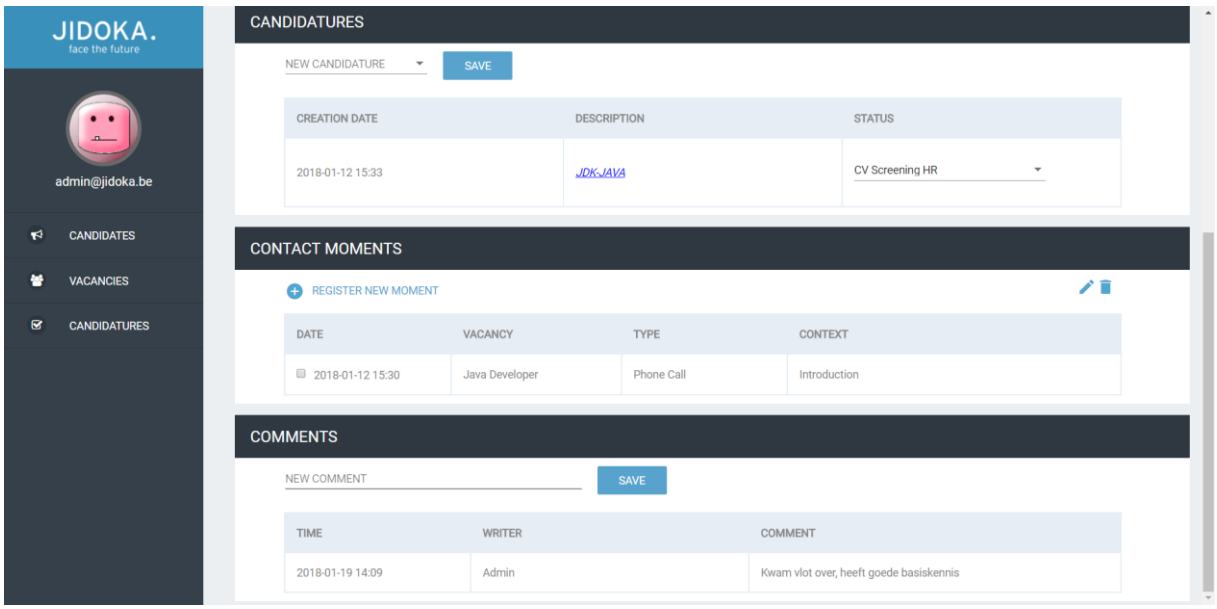
Figuur 14 - Upload image

Na het voltooien van deze stappen werd er succesvol een nieuwe kandidaat aangemaakt, deze is nu zichtbaar in het overzicht. Wanneer er geklikt wordt op een kandidaat in het overzicht wordt het detail van deze kandidaat weergeven. Op dit detail vinden we alle gegevens over de kandidaat terug, maar kunnen we deze ook aanpassen. De afbeelding kan gewijzigd worden door het klikken op de huidige afbeelding, dit geldt ook voor de CV.



Figuur 15 - Detail kandidaat

Onder de persoonlijke gegevens bevindt zich een tabel waarin de kandidaturen voor deze kandidaat beschreven worden. Een kandidatuur is wanneer de kandidaat solliciteert voor een vacature. Door middel van een keuzelijst kan de gebruiker ook een nieuwe kandidatuur toevoegen aan een kandidaat. In deze keuzelijst worden alle vacatures getoond die open zijn. Een vacature heeft namelijk meerdere statussen: Open, On Hold of Closed. Bij het toevoegen van zo'n kandidatuur wordt ook deze getoond in de tabel, dit samen met de creatiedatum, code van de vacature en de status van de kandidatuur.



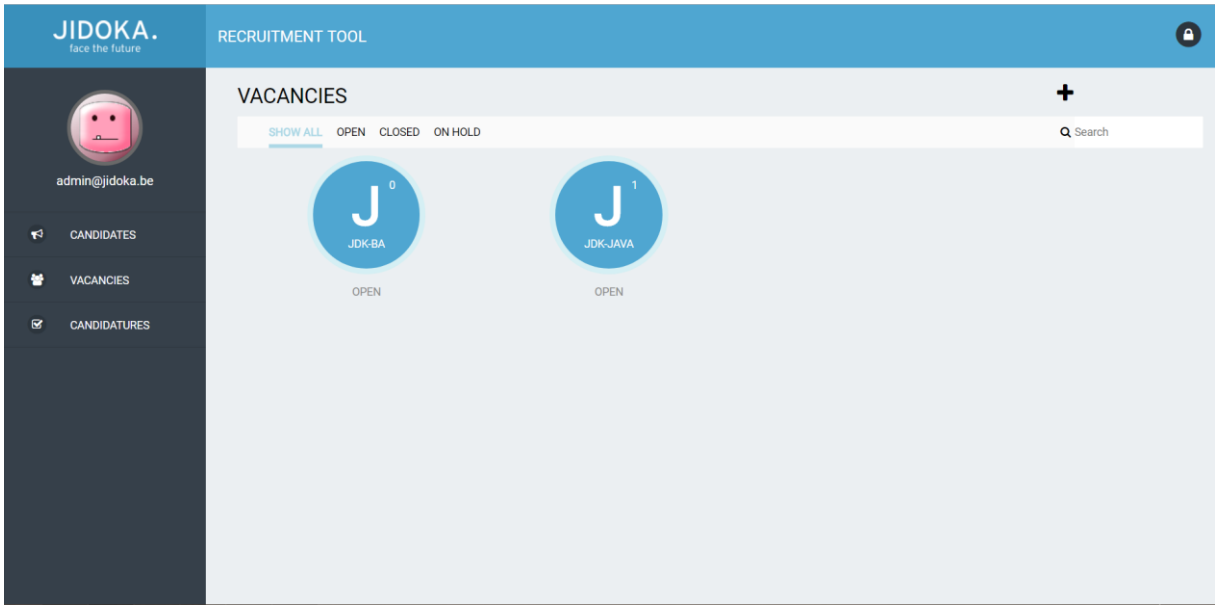
Figuur 16 - Detail Kandidaat

Onder het overzicht van de kandidaturen bevinden zich nog twee secties: contactmomenten en comments.

De contactmomenten zijn alle momenten waarbij de HR-medewerker contact heeft gehad met de kandidaat. Dit kan mondeling, telefonisch als op e-mail zijn. Door het registreren van zo'n momenten kan bijgehouden worden hoeveel contact er al is geweest met een kandidaat en hoelang deze al aan het solliciteren is.

De comments spreken voor zich. Onder deze sectie kunnen alle medewerkers van JIDOKA commentaar geven op een kandidaat. Indien ze een opmerking hebben over, bv. vorige werkervaring met de betreffende kandidaat, kan deze hier geschreven worden. Zo kan de HR-medewerking meningen van collega's verzamelen over een kandidaat.

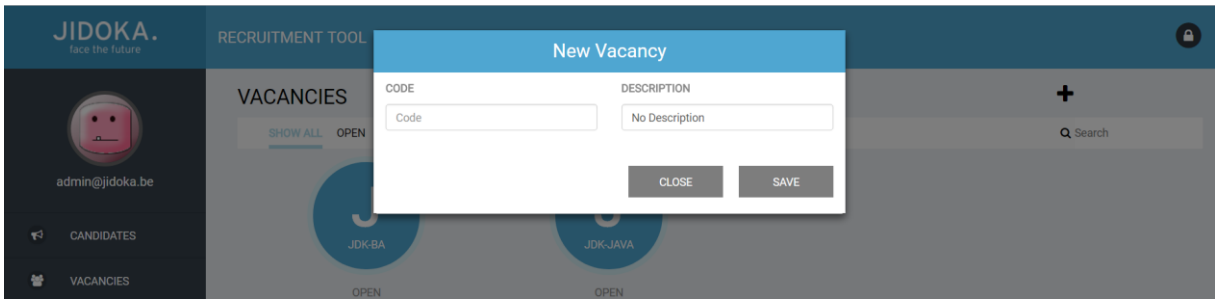
Via het menu aan de linkerkant van de applicatie zal er vervolgens genavigeerd kunnen worden naar het overzicht van de vacatures. Deze worden volgens dezelfde stijlgids als de kandidaten weergeven in cirkels.



Figuur 17 - Overzicht vacatures

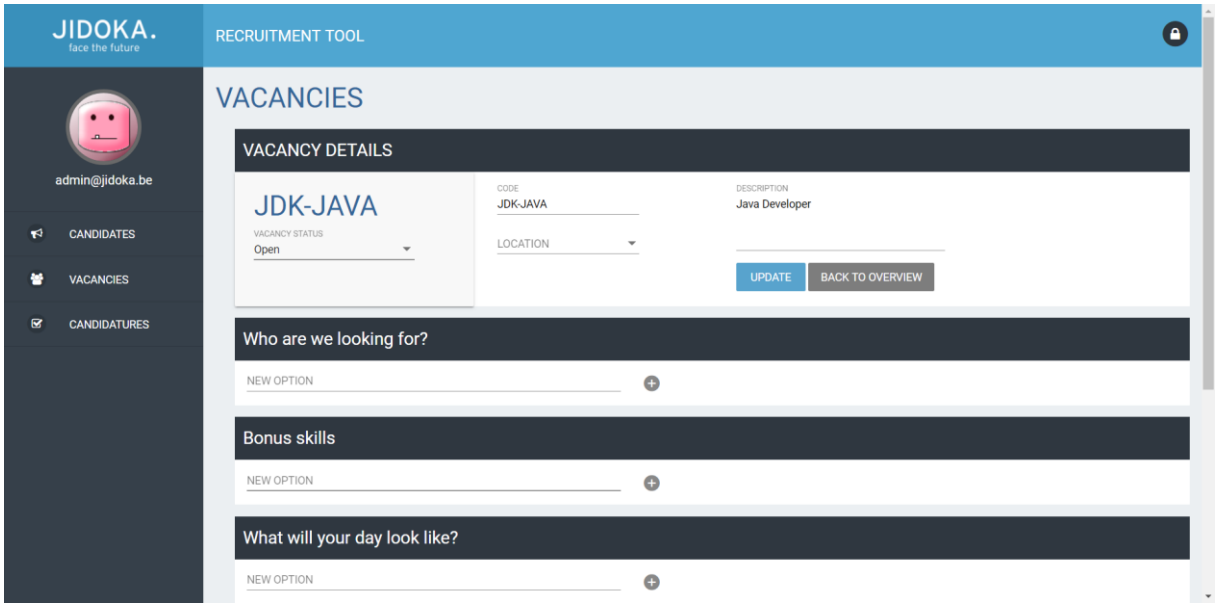
Elke vacature heeft zijn eigen unieke code die de vacature definieert. De beginletter van deze code, zowel als de volledige code, worden weergeven in het overzicht. Ook bevat de cirkel een klein cijfertje, deze stelt het aantal kandidaten voor die gesolliciteerd hebben voor betreffende vacature.

Zoals bij het overzicht van kandidaten kunnen nieuwe vacatures aangemaakt worden via de grote plus aan de rechter bovenkant. Bij het aanklikken van deze plus wordt de pop-up die instaat voor het aanmaken van een vacature aangeroepen.



Figuur 18 - Aanmaken vacature

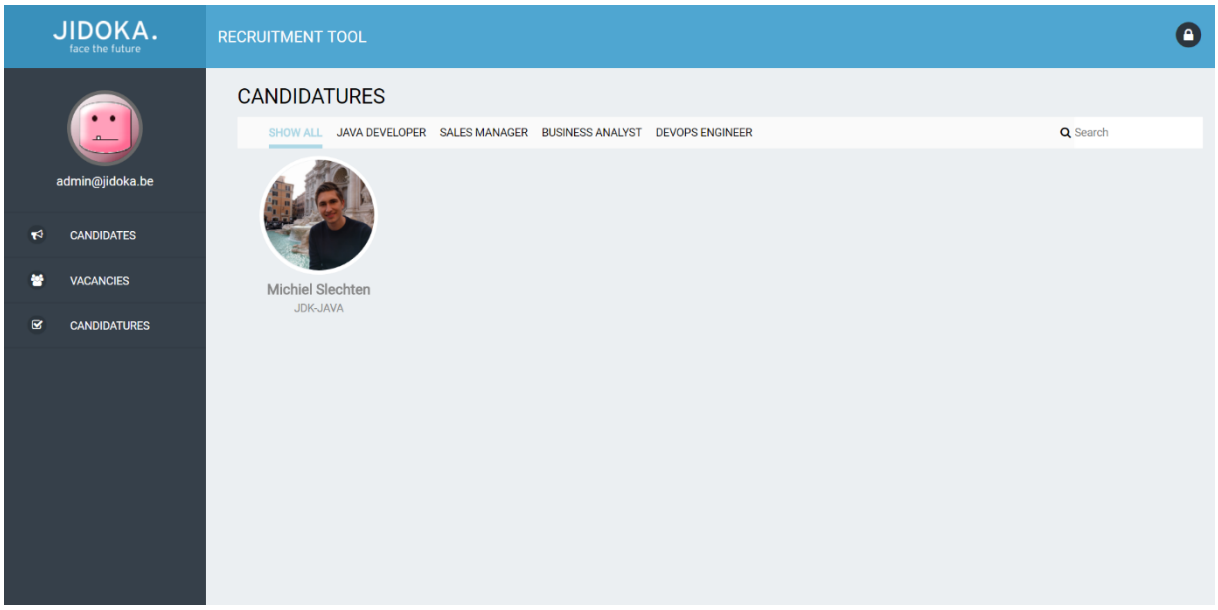
Het aanmaken van een nieuwe vacature is zeer simpel. Er wordt initieel enkel een code en een beschrijving gevraagd. Voor de code bestaat nog geen vaste norm, JIDOKA is vrij om eender welke code te gebruiken die zij geschikt vinden, bv. 'JDK-JAVA' voor een Java Developer vacature.



Figuur 19 - Detail van een vacature

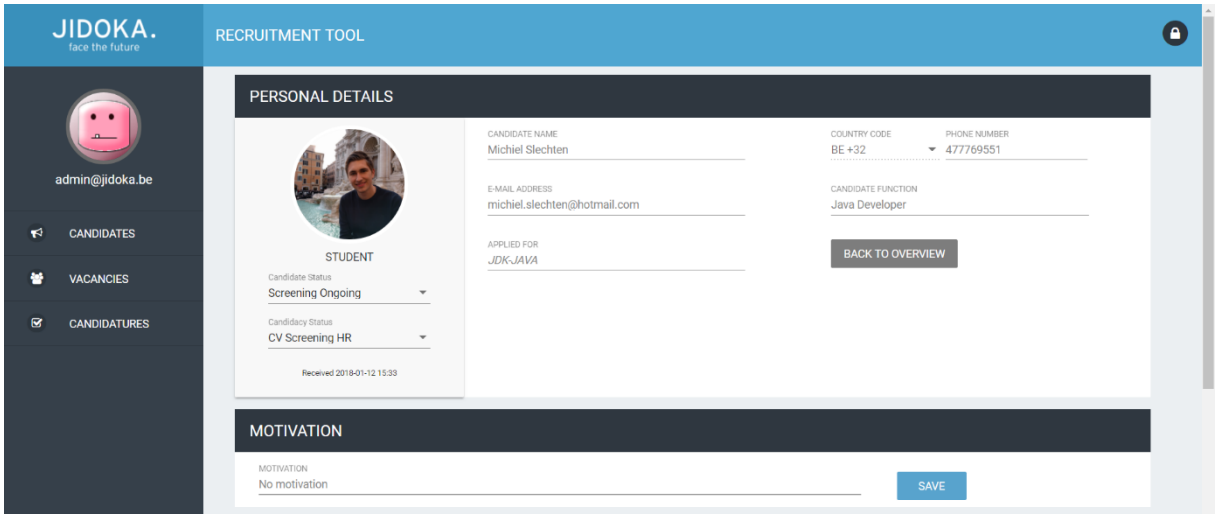
Bij het openen van het detail van een vacature kunnen de basisgegevens zoals code, omschrijving, status maar ook locatie aangepast worden. De mogelijke locaties zijn tegenwoord Hasselt, Mechelen of Client-Side, dit laatste is op locatie bij een klant. Ook wordt er de mogelijkheid gegeven om de vacature aan te vullen met meer details zoals bv. vereiste eigenschappen van de gezochte kandidaat.

Ten Slotte kan er via het menu genavigeerd worden naar het overzicht van de kandidaturen. Deze pagina ligt ook in de stijllijn van de kandidaten en vacatures.



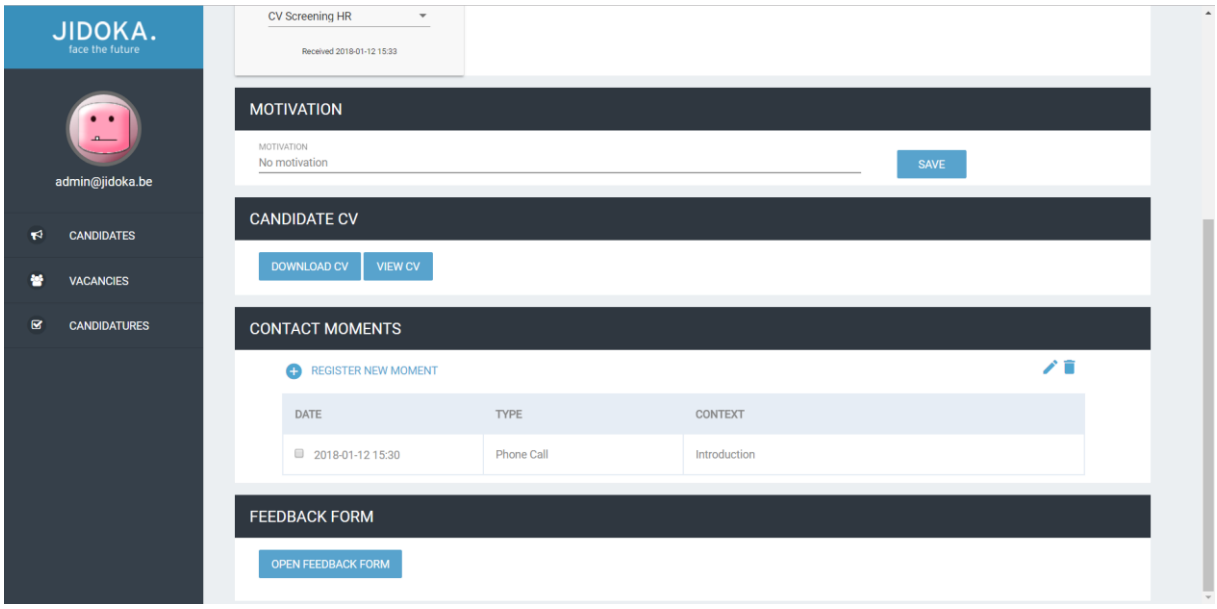
Figuur 20 - Overzicht kandidaturen

Hier worden alle kandidaten weergegeven die gesolliciteerd hebben voor een vacature. De foto, naam van de kandidaat en code van de vacature wordt weergeven. Er kan ook weer gefilterd worden op functie van de kandidaat en men kan zoeken naar de naam van een kandidaat. Via deze pagina kunnen er geen nieuwe kandidaturen aangemaakt worden, deze kunnen enkel aangemaakt worden via het detail van een kandidaat. We kunnen wel het detail van een kandidatuur openen.



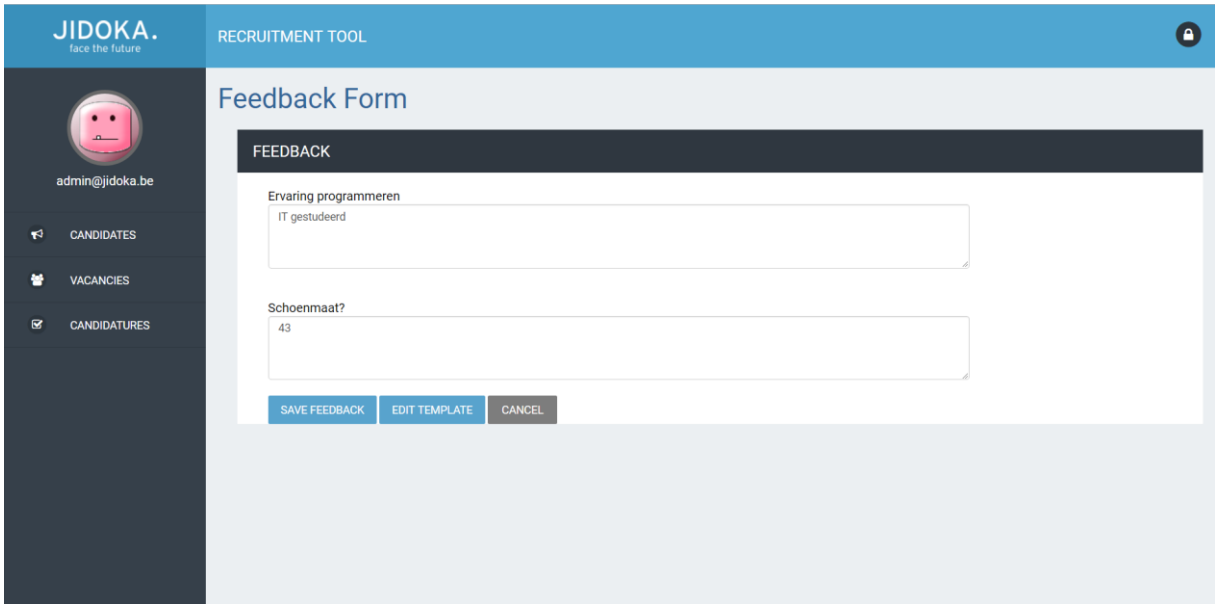
Figuur 21 - Detail kandidatuur

Op het detail vinden we de algemene gegevens van de kandidaat terug, enkel kunnen deze hier niet aangepast worden. Eronder bevindt zicht de motivatie die de kandidaat opgegeven heeft bij het solliciteren voor een geselecteerde vacature.



Vervolgens, wanneer we naar beneden scrollen, kunnen we het CV van de kandidaat downloaden of online bekijken. Zoals bij het detail van een kandidaat kunnen er ook weer contactmomenten geregistreerd worden voor een kandidatuur om de voortgang van en communicatie tussen een kandidaat op te meten.

Uiteindelijk kunnen we een pagina openen met feedback. Deze pagina bevat een template van vragen die elke kandidaat gevraagd kan worden. Op deze manier is er een universele vragenlijst maar kunnen de antwoorden ook makkelijk opgeslagen worden. Dit ziet er ongeveer zo uit:



Figuur 22 - Feedback form

Dit bevat in grote lijnen het gebruik van de gerealiseerde applicatie binnen onze stageperiode.

3.6 Persoonlijke reflectie

Ik kan zeggen dat ik bij het uitvoeren van deze stageopdracht enorm veel heb bijgeleerd, vooral op technisch vlak. Voor het uitwerken van mijn opdracht heb ik proactief deelgenomen aan het coderen van het project. Iets dat als SWM'er niet direct evident is.

De eerste weken had ik het dan ook behoorlijk moeilijk. Ik voelde dat ik een beetje achter mijn medestudent Toon aanhuppelde en begon mezelf hopeloos te onderschatten. Toen heb ik gewoon voor mezelf de tijd moeten nemen om de nieuwe technologieën (Angular 4 en Spring Boot) aan te leren op mijn eigen tempo. De ondersteuning die collega's me boden bij dit proces was ongelooflijk en snel kon ik ook meedraaien bij het schrijven en voltooien van mijn eigen story's.

Bij de aanvang van het project beschouwde ik mijn functioneren dan ook totaal niet als optimaal, met de kennis die ik heb opgedaan als Software Manager op de PXL kon ik wel direct aan de slag met het schrijven van nieuwe story's. Enkel het programmeren van deze story's was een enorm struikelblok ik het begin door het ontbreken van mijn Java kennis. Na enkele weken van bijwerken voelde ik wel dat ik er ongelooflijk op vooruitgegaan was. Op het einde van de stageperiode voelde ik mij ook echt trots op mijn verwezenlijkte kennis, ik kon al zonder problemen volledig zelfstandig coderen aan de applicatie.

Dat ik mee moest helpen met de technische kant van de opdracht is voor mij achteraf gezien een gigantische meerwaarde in deze stage. Aangezien ik ook de analyse van het project deed, zat ik eigenlijk in een 'best of both worlds' -scenario.

De volledige kennis die me zo goed bijgebracht is op Hogeschool PXL op vlak van testing en analyse heb ik vlijtig kunnen toepassen, maar ook op technisch vlak heb ik mezelf helemaal kunnen ontplooien bij JIDOKA.

II. Onderzoekstopic

1 Inleiding

Voor de onderzoeksvraag tijdens mijn stage zal ik Story Mapping vergelijken met Event Storming. Beide zijn technieken die helpen bij het analyseren van functionaliteiten en requirements om tot een geheel project of product te komen.

JIDOKA past op dit moment al Story Mapping toe bij projecten, maar is naast deze techniek ook nieuwsgierig naar Event Storming. In dit onderzoek zal ik voor hen deze 2 technieken met elkaar vergelijken.

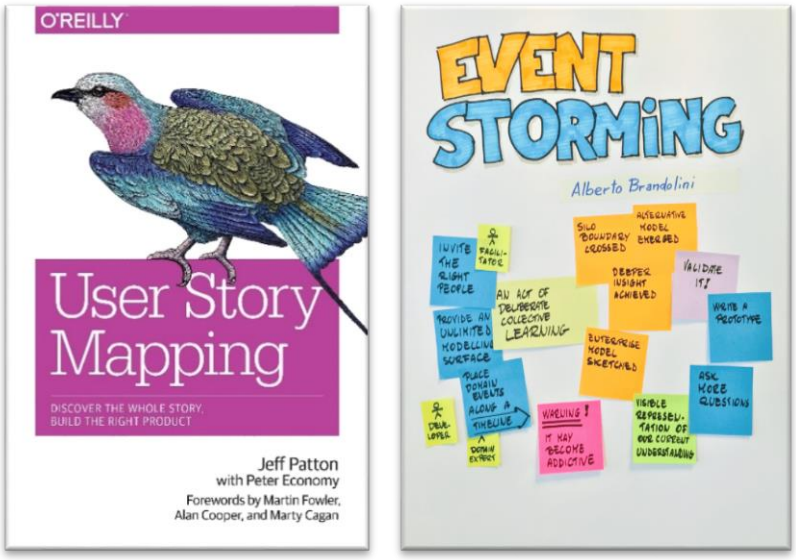
2 Methode van onderzoek

2.1 Aanpak

Dit onderzoek is grotendeels gebaseerd op twee boeken. Beide boeken zijn zowat de basis voor desbetreffende analysetechnieken. Ook wordt er een studie gedaan die zich baseert op verscheidene artikelen van het internet.

Het eerste boek betreft User Story Mapping, geschreven door Jeff Patton. Gepubliceerd door O'Reilly Media op 1 oktober 2015 met ISBN-code: 9781491904909

Het tweede boek is Event Storming geschreven door Alberto Brandolini. Het boek is voorlopig enkel terug te vinden op Leanpub: een platform dat toelaat boeken digitaal en publiekelijk te schrijven en te verkopen. Het heeft dus nog geen uitgeverij of ISBN-code. Het boek is ook nog niet afgewerkt.



Figuur 23 - User Story Mapping & Event Storming boeken

Aan de hand van de inhoud van deze twee boeken wordt een beeld geschetst van de gebruikswijze en toepassing van beide technieken. Ook heb ik enorm geluk met mijn stagebedrijf JIDOKA. Samen met een paar collega's die interesse hebben in dit onderzoek zullen we enkele bookreading sessions organiseren zodat we onderling kunnen discussiëren over de inhoud van beide boeken. Vervolgens kan er een vergelijking gemaakt worden.

3 Literatuurstudie

3.1 Artikel 1: User Story Mapping

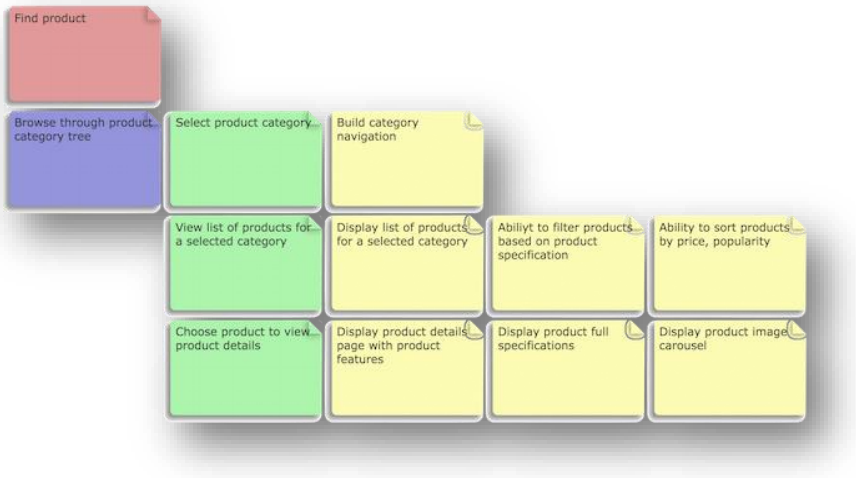
Het eerste artikel betreft User Story Mapping en is uitgebracht door ThoughtWorks, een multinationaal bedrijf dat zich specialiseert in softwareanalyse en de uitwerking van grote projecten.[6]

Een van de belangrijke zaken voor het opbouwen van een project is de identificatie van de 'requirements'. Vaak is het moeilijk om deze te vinden en zeker om ze te prioriteren. Story Mapping is een workshop waarbij een team betrokken is om visueel een product backlog op te stellen in plaats van een saai document. Deze visuele voorstelling wordt meestal voorgesteld door plaknotities op een muur.

Story Mapping is het uitwerken van de requirements in de vorm van een boomstructuur. Het geheel is opgebouwd uit doelstellingen. Doelstellingen kunnen tot stand komen door activiteiten te voltooien. Deze activiteiten raken voltooid door gebruikers die een opdracht uitvoeren.

Doelstelling > Activiteit > Opdracht > Story's.

Een voorbeeld hiervan is het uitwerken van een webwinkelapplicatie zijn. Een doelstelling is dat er een product gevonden kan worden. Dit wordt verwezenlijkt door het doorzoeken van producten, in een categorie, waarin de gebruiker kan filteren of zoeken. De plaknotities zouden er zo kunnen uitzien:



Figuur 24 - Voorbeeld van User Story Mapping

Een groot voordeel van Story Mapping is dat, doordat al deze functionaliteiten visueel voorgesteld worden, de mensen die aan het project werken een beter algemeen begrip hebben van wat er nu eigenlijk gerealiseerd moet worden. Story Mapping nodigt meer uit om te discussiëren en dieper in te gaan op functionaliteiten dan een saai uitgeschreven document. Het vormt *'shared understanding'.* De opbouw van deze 'shared understanding' betekent dat zowel de klant als de werknemer hetzelfde idee hebben. Om tot een goed resultaat te komen is het belangrijk dat alle betrokken partijen zich op dezelfde lijn bevinden.

Omdat alles effectief besproken wordt kan er ook gemakkelijk een prioriteit geplakt worden op de verschillende doelstellingen en activiteiten. Er vormt zich een algemeen besef over hoe belangrijk

bepaalde functionaliteiten zijn tegenover andere. Zo kunnen de verschillende doelstellingen gemakkelijk ingedeeld worden in releases en MVP's om een bedrijf te assisteren in het behalen van deadlines en releases. Hierdoor wordt een 2-dimensionale backlog gecreëerd in plaats van de traditionele (1-dimensionale) backlog.

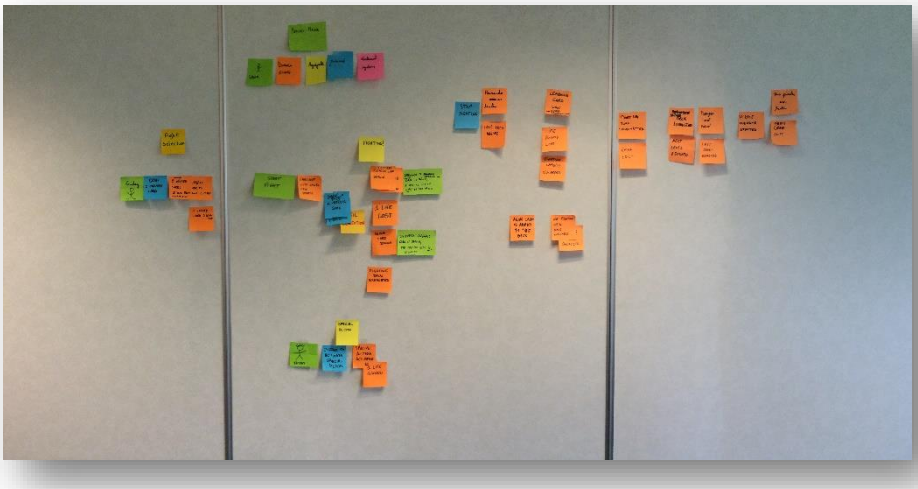
3.2 Artikel 2: Event Storming

Het volgende topic is Event Storming in een artikel van TechBeacon, een platform dat zich baseert op het verzamelen en verspreiden van informatie gebruikt in de bedrijfswereld. [7]

Event Storming is een snelle 'group modeling'-techniek die handig is voor het versnellen van development teams. Het is een techniek die niet enkel moet dienen als softwareanalyse maar ook toegepast kan worden op elk technisch domein. Event Storming legt zich vooral toe op de participatie van alle betrokken stakeholders en teams zodat het gehele proces binnen een organisatie voor iedereen duidelijk is.

Een belangrijk keyword in Event Storming is een 'Domain Event'. Een Domain Event is een proces of een actie binnen een organisatie dat toedraagt tot het realiseren van een flow of product. Dit wordt omschreven als een werkwoord in de verleden tijd.

Tijdens een Event Storming workshop wordt het proces van de organisatie volledig in kaart gebracht, en van daaruit worden probleemgebieden en kandidaten voor automatisatie aangeduid. In dit proces zal elk Domain Event opgenomen moeten worden. Elk Domain Event wordt apart genomen samen met zijn gebruikers, verwacht resultaat en trigger die dit event in gang moet zetten. Op deze manier worden de belangrijkste aspecten van deze Domain Events zichtbaar en kunnen scenario's uitgedacht worden. Uiteindelijk zullen er relaties gelegd worden tussen deze verschillende Events en ontstaat er een model. Dit zou zo voorgesteld kunnen worden:



Figuur 25 - Voorbeeld Event Storming

Een groot voordeel van deze techniek is dat er al zeer snel een visueel duidelijk model ontstaat waarop de applicatie gebaseerd zal kunnen worden. Ook is dit model zeer toegankelijk en verstaanbaar voor iedereen binnen het team, aangezien het samen opgebouwd is. Event Storming creëert een groot en simpel overzicht van het project.

3.3 Artikel 3: Event Storming

Omdat over het algemeen Story Mapping een veel bekendere techniek is zal ik met het 3e artikel op het internet proberen Event Storming wat bereikbaarder te maken. Dit artikel komt van de Washington Technologie Association, een non-profit organisatie die zich bezighoudt met het verzamelen van gegevens en informatie voor de technologie community. [8]

Volgens Washington Technology is Event Storming een goede techniek voor het onthullen van een groot en ingewikkeld business probleem. Het wordt er gebruikt om een gigantische service op te delen in kleinere microservices. Door middel van Event Storming kunnen deze microservices gedefinieerd worden op enkele uren i.p.v. enkele weken. Dit omdat na een sessie het team een begrip heeft van de processen en de onderlinge business domeinen.

Wat nodig is voor zo'n sessie, zoals ook besproken in Artikel 2, zijn de volgende attributen:

- Plaknotities in minstens 3 verschillende kleuren
- Een grote rol papier
- Een stapel stiften
- Een bord met de legende van de verschillende kleuren plaknotities

Verwijder ten slotte alle stoelen zo ver mogelijk van de muur waarop de sessie gaat plaatsvinden en plak het papier over de muur die de workspace moet voorstellen. Het is de bedoeling dat iedereen zo vlot mogelijk mee kan participeren. Begin met het plaatsen van domain events (iets wat gebeurt in het domein) met plaknotities op de muur. Vul de domain events vervolgens aan met plaknotities in andere kleuren: externe factoren en problemen of conflicten.

Op het einde van de sessie zal je overblijven met 3 opties: een oplossing, de ontdekking dat het probleem ergens anders ligt of dat het probleem niet op te lossen is.

Het voordeel van deze sessie is dat een project een stuk versneld wordt. In plaats van veel middelen te spenderen aan het zoeken van een oplossing ontdek je soms met deze 'korte' sessie dat je probleem gewoonweg niet eens op te lossen is. Maar ook alle participanten leren meer over het betreffende domein. Dit begrip bevordert het vinden van oplossingen. Uiteindelijk creëert Event Storming een grote map die de volledige business flow en zijn processen blootlegt.

3.4 Vergelijking

Het is duidelijk dat beide technieken zwaar focussen op een *'shared understanding'* binnen een team. Zowel Story Mapping als Event Storming slaat op het betrekken van alle betrokkenen personen bij het opstellen van de analyse van het project.

Het verschil ligt bij de flow die ontstaat. Bij Story Mapping wordt het project wordt uitgewerkt op zo'n manier dat er een verhaal over verteld kan worden. Event Storming pakt elk aspect van een applicatie apart aan en werkt dit volledig uit, om dan later relatie en hechtingen op te stellen die alles tot een geheel moeten brengen.

Een ander verschil is de uitkomst van beide technieken. Story Mapping deelt de functionaliteiten mooi in een backlog in en het helpt ons met het identificeren van de onderdelen die nodig zijn om ons project van een goede flow te voorzien. Indien gewenst biedt Story Mapping ook een overzicht van mogelijke releases of een MVP. Bij Event Storming ligt de focus vooral op het identificeren of oplossen van een probleem. De uitkomst van deze techniek ligt eerder bij het zoeken van een domein waarop focus gelegd moet worden omdat dit domein onduidelijk is of er moeilijk een oplossing te bekomen is.

Ook in de sessie zelf is een licht verschil. User Story Mapping kan met enkele stakeholders of domain experts uitgevoerd worden, terwijl bij Event Storming het volledige team met alle stakeholders geacht wordt deel te nemen.

3.5 Persoonlijke reflectie

Zelf ben ik veel meer vertrouwd met Story Mapping, dit is een veel bekendere techniek en eentje die ik in het IT-Project ook al heb toegepast. Het is een gemakkelijke techniek waarbij je simpelweg alle requirements van een applicatie op kan schrijven en gemakkelijk een samenhang kan terugvinden. Event Storming daarentegen is iets waar ikzelf nog niet veel van had opgevangen, eigenlijk had ik er nog nooit van gehoord. Ik vond ook zeer weinig artikels terug met een duidelijke uitleg over Event Storming. Het lijkt me ook een techniek waar enorm veel bij komt kijken en waar je technisch al behoorlijk wat ervaring voor moet hebben. Zeker omdat je geen flow uitwerkt, maar verschillende domeinen apart van elkaar. Ik ben benieuwd er meer over te weten te komen.

4 Onderzoek

4.1 User Story Mapping

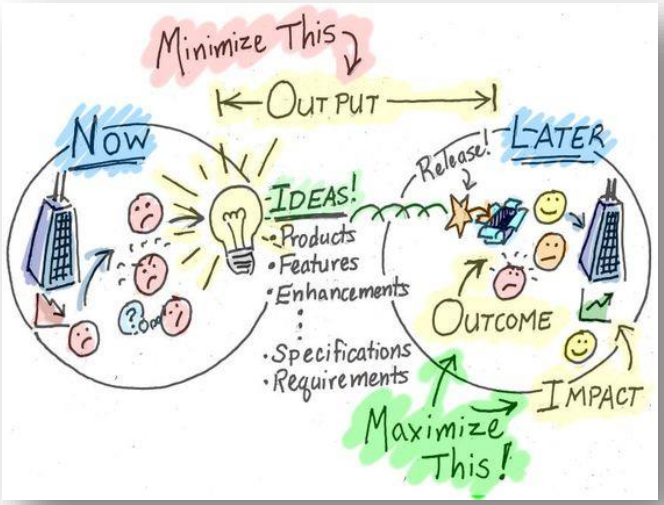
3.1.1 Inleiding

Dit onderzoek begint bij het boek User Story Mapping. Het is geschreven door Jeff Patton, een analist die zelf al 20 jaar actief is in het vak en talloze ervaringen heeft met analyse en projectmanagement. Hij licht in zijn boek Story Mapping toe a.d.h.v. voorbeelden uit persoonlijke ervaringen en projecten, en dit op een ludieke en aangename manier.

3.1.2 User Story Mapping in werking

Tijdens het voorwoord haalt Patton al steevast aan dat een van de belangrijkste zaken bij het gebruik van Story Mapping de understanding is tussen klant, analist en de programmeurs. Een belangrijke eigenschap van Story Mapping is het aanzetten tot conversatie tussen story's: "*Shared documents aren't shared understanding!"[9].* Volgens Patton horen story's ook niet perfect uitgewerkt of gedocumenteerd te worden, het is belangrijker dat je er over kan discussiëren dan dat je een perfect document neerschrijft. Tijdens deze discussie moeten er ideeën ontstaan die het product helpen evolueren naar een product dat de wereld kan veranderen, voor de eindgebruiker tenminste.

Alles tussen een idee en de uitwerking ervan noemen we 'output'. Output is wat we bouwen. Al het werk van de developers en de analyse dat gebeurt om dit werk zo efficiënt mogelijk te laten verlopen is het proces van de output. Maar de output is niet het belangrijkste! Het is de outcome die we willen: het finale resultaat of product. Volgens Patton moeten we tijdens het realiseren van een product streven naar zo min mogelijk output maar een zo optimaal mogelijke outcome. Dit door te focussen op een zeer specifieke doelgroep en kleinere releases die simpel maar effectief zijn.



Figuur 26 - Minimize Output & Maximize Outcome

Maar wat is Story Mapping nu eigenlijk? Patton haalt als grote voorbeeld het verhaal van Gary Levitt aan. Gary was iemand die een markt zag voor een webapplicatie die gebruikt kon worden om bands en optredens te promoten. Gary zag deze applicatie heel groots maar bij de aanvang van het project ondervond hij dat de ontwikkeling van zijn applicatie zoveel tijd en geld zou kosten dat het niet meer haalbaar voor hem zou zijn. Dit is waar hij de hulp van Jeff Patton inriep.

Aan de hand van kleine plaknotities werden alle functionaliteiten en gebruikers, die Gary in zijn applicatie wou voorzien, opgeschreven. Vervolgens werden deze functionaliteiten gelegd in de prioriteit die Gary voor ogen had. Zo werd er over elke plaknotitie nagedacht over het nut van de functionaliteit en konden gaten in het verhaal snel gevonden en opgevuld worden. Eens alle plaknotities gefilterd en in volgorde lagen, zag de vloer er ongeveer zo uit [9]:

Figuur 27 - User Story Mapping: Gary's Story

Er ontstond een flow in het verhaal van de applicatie van Gary. Het was nu duidelijk voor hem op welke manier zijn applicatie ging werken, maar ook wat de cruciale delen waren die de applicatie ondersteunden. De gele lijn stelt hier de backbone voor: Het zijn alle plaknotities met grote delen van de applicatie, bijvoorbeeld "Manage Concert". Onder dit onderdeel worden dan notities met details of kleinere stappen gelegd zoals b.v. "Create Concert", "Edit Concert",... Deze worden naargelang prioriteit top-down onder het grote onderdeel gesorteerd. Het belangrijkste van dit mappen is dat Gary gaten kon vinden in zijn verhaal en uitgebreid kon nadenken over zijn gebruikers en hun doelen. Op deze manier kan de applicatie zo efficiënt mogelijk opgebouwd worden.

Het voordeel van Story Mapping is dat er ook makkelijk een understanding kan ontstaan tussen verschillende teams. Het uitmappen van activiteiten en de details ervan legt verbanden tussen de benodigde teams die elkaar ondersteunen om tot een geheel te komen.

Figuur 28 - Discussing User Story Mapping

3.1.3 Waarom User Story Mapping?

Een groot voordeel van User Story Mapping is de prioritering van de verschillende story's. Aangezien onder de backbone de verschillende story's top-down gesorteerd worden naargelang prioriteit kan er makkelijk achterhaald worden welke story's een baseline of een minimum product vormen voor de applicatie. Op deze manier kan de map opgedeeld worden in verschillende releases. Naargelang het project vordert kan het voorvallen dat een story niet meer relevant is, deze kan dan altijd zonder problemen verschoven worden naar een latere release of verwijderd worden uit de map.

Een ander handig gegeven van User Story Mapping is dat bottlenecks of gebreken makkelijk opgespoord kunnen worden. Bij het navertellen van het uitgemapte project komen er soms gaten aan het licht. Story Mapping laat ons toe deze gebreken op een simpele manier op te vullen.

Ten Slotte legt User Story Mapping de interacties tussen gebruiker en systeem volledig bloot. Alle stakeholders, en vooral de developers, weten zo hoe de applicatie afgestemd moet worden om deze interacties zo efficiënt mogelijk te laten verlopen.

3.1.4 Wanneer User Story Mapping?

User Story Mapping is toepasbaar op elk project en is het meest efficiënt bij de start van een project. Het helpt namelijk om een project te vertalen naar een backlog. Het is ook aan te raden wanneer 'the bigger picture' zoek is in het verhaal van de klant: Story Mapping is dan een zeer efficiënte manier om ideeën naar bruikbare story's om te vormen. Ook om een startpunt of een baseline voor een project te vinden kan Story Mapping een wereld van verschil maken.

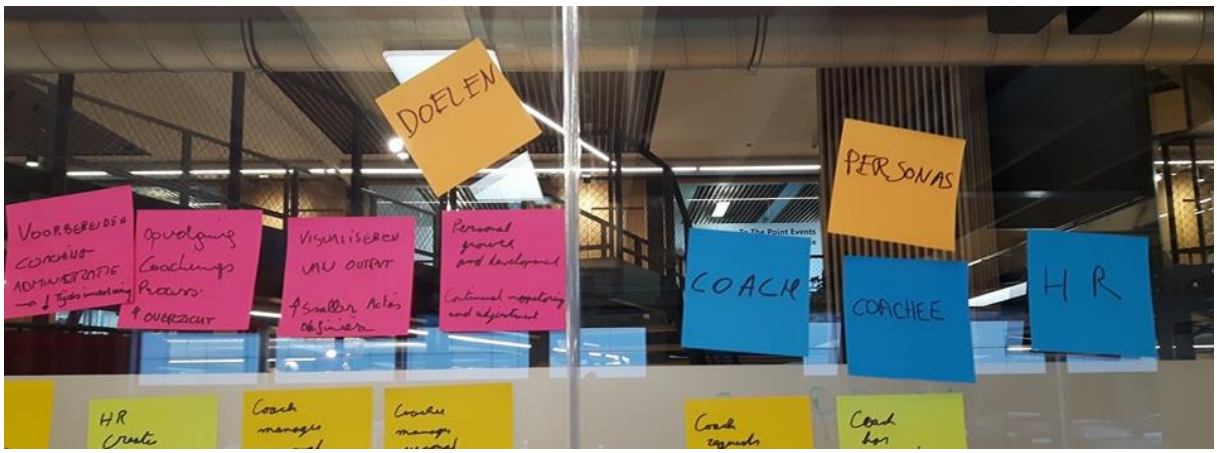
In principe zou Story Mapping zelfs meerdere keren binnen eenzelfde project moeten gebeuren. Door deze oefening regelmatig te herhalen wordt de Story Map actueel gehouden met de realiteit op de markt en de wijzigende noden van de klant en gebruikers.

3.1.4 Proof of Concept

Op een dinsdagavond werd er op JIDOKA een sessie georganiseerd om, met enkele collega's en geïnteresseerden, Story Mapping in de praktijk uit te voeren.

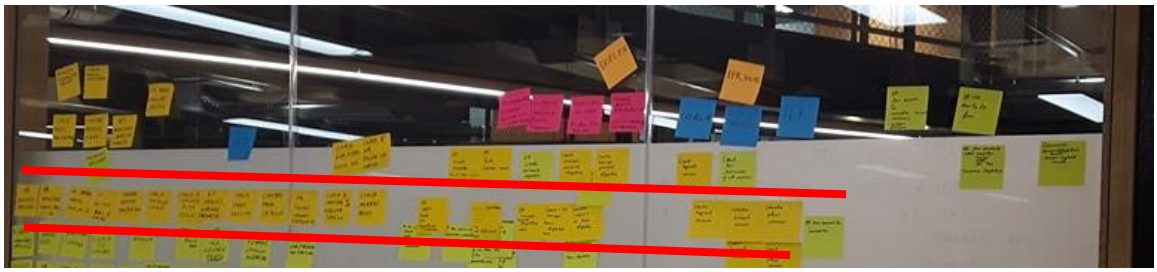
Omdat we met een grote groep waren werd er besloten twee groepen te vormen. Een groep ging Story Mapping toepassen op een onderdeel binnen JOYN, een onderneming die een digitale klantenkaart op de markt brengt. De andere groep ging dieper in op de HR-tool, een applicatie binnen JIDOKA die de vooruitgang op performance en de coaching van werknemers opvolgt. Ikzelf heb me aangesloten bij de HR-tool aangezien dit meer aansluit bij JIDOKA.

Er werd begonnen met het definiëren van actors of personen. Dit zijn de betrokken personen die te maken zullen hebben met de applicatie. Vervolgens werden de doelen van de HR-Tool in kaart gebracht. Met deze doelen moet gedurende heel de sessie rekening gehouden worden, functionaliteiten of acties die niet behoren tot minstens één van deze doelen moeten namelijk niet opgenomen worden in onze Story Map.



Figuur 29 - Doelen en personen

Nadat de doelen en personen duidelijk waren konden we aan de sessie beginnen. We startte met het opstellen van onze backbone, de basis voor de applicatie. Paula Kozanecka, de product owner van de HR-tool, vertelde in grote lijnen de flow die de HR-Tool zou moeten doorlopen. Iedereen stond al klaar met een pen en notities in de hand, ongeduldig te wachten om er aan te beginnen. Dit bleek niet zo evident toen we allemaal hetzelfde begonnen op te schrijven. Terwijl Paula het verhaal deed heeft dus één persoon die flow opgeschreven en samen met de groep omgezet in onderdelen die samen het verhaal van de HR-Tool voorstellen. Belangrijk was dat hier nog geen details in opgenomen werden, deze komen later aan bod bij het uitdiepen van de grote onderdelen. Toen Paula haar verhaal had gedaan konden we enkele onderdelen opsplitsen in meerdere onderdelen. Sommige werden ook volledig geschrapt omdat ze niet behoren tot het doel of omdat ze onder te verdelen waren in een ander onderdeel. Uiteindelijk werd er ook nog behoorlijk geschoven aan de volgorde van de onderdelen.



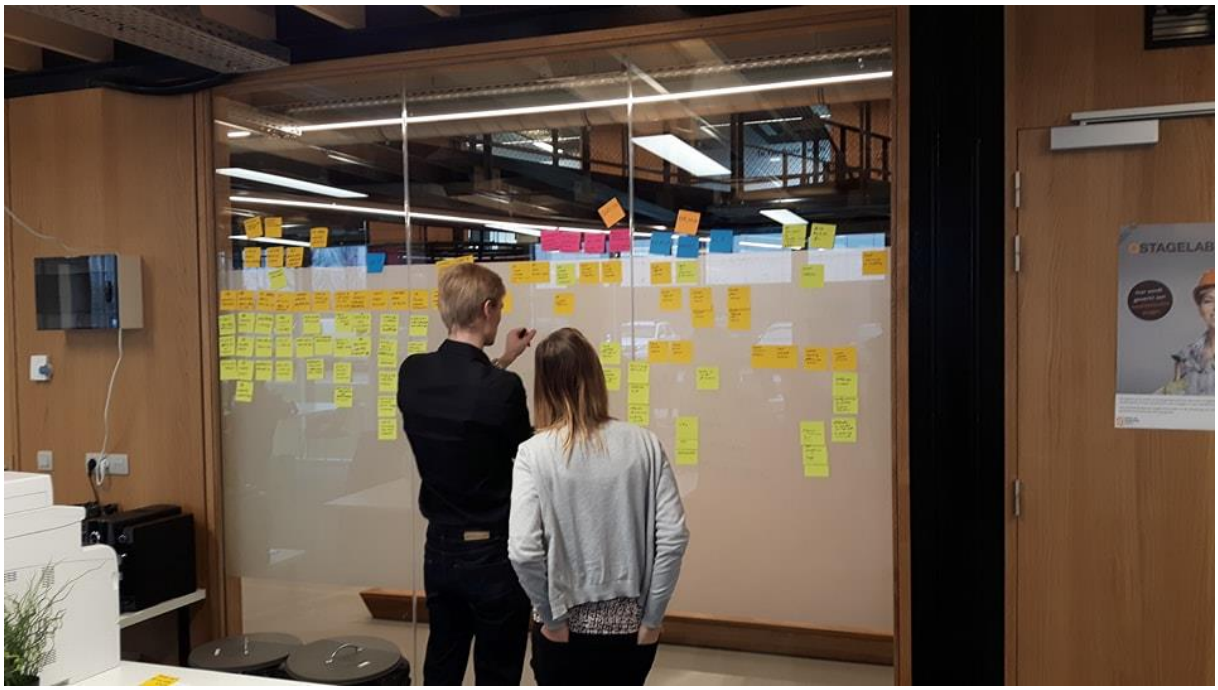
Figuur 30 - Story Mapping backbone

Toen we uiteindelijk de hoofdactiviteiten in een flow van links naar rechts hadden geschikt konden we beginnen met het uitdiepen van deze activiteiten. Van elke hoofdactiviteit plakten we notities, van boven naar onder, onder de hoofdactiviteit. Deze notities bevatten de acties die uitgevoerd moeten worden om tot de hoofdactiviteit te bekomen, in de meeste gevallen waren dit CRUD-operaties voor ons.

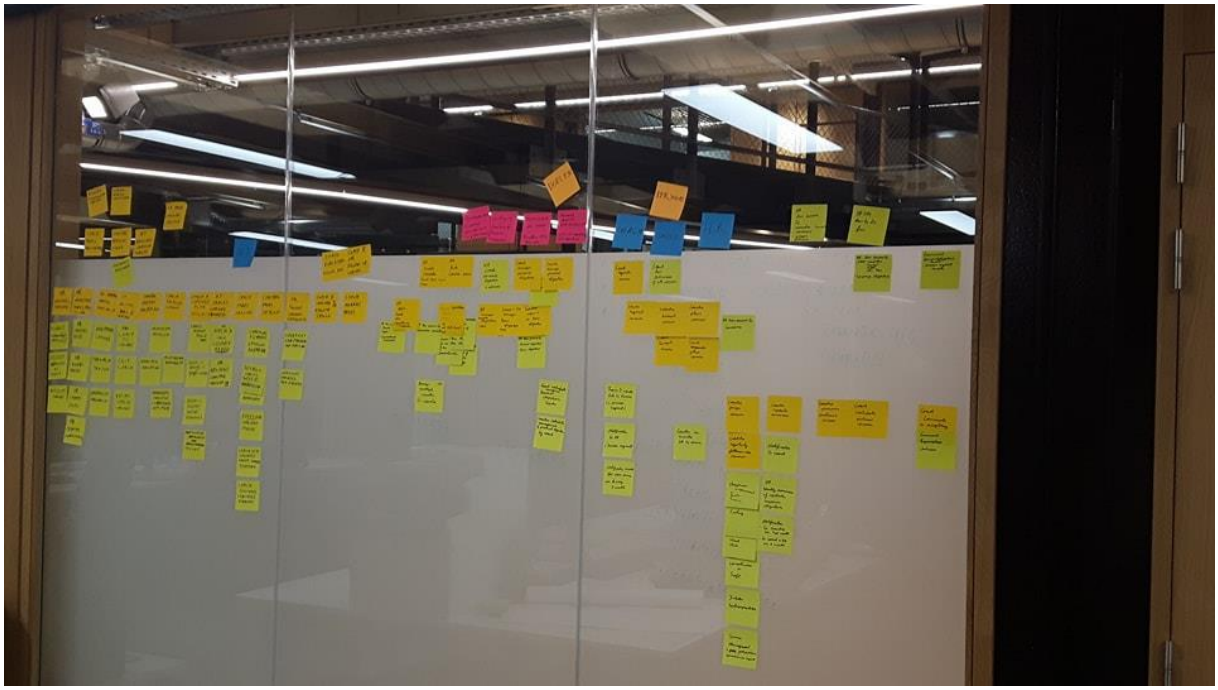
Hoe meer notities er opgeplakt werden, des te meer vragen er ontstonden over de verbanden en volgorde van alle acties. Samen met Paula moest er dan naar een oplossing gezocht worden die deze acties in de juiste positie plaatste, en waarbij soms nieuwe acties ontstonden ter ondersteuning. Zoals Maarten, een developer bij JIDOKA, al mooi opmerkte: “Bij elke notitie die er opgeschreven wordt voel ik de applicatie uit elkaar getrokken worden”. Een groot verschil dat ons opviel in vergelijking met Event Storming is dat hier veel dieper gegaan wordt en er meer discussie plaatsvindt over de functie en plaatsing van een actie.

Figuur 31 - Story Mapping werking 1

Voor een groot deel is de HR-tool uitgewerkt volgens User Story Mapping. Aangezien dit proces merkbaar meer tijd in beslag neemt dan Event Storming kwam de volledige HR-tool helaas niet aan bod. In de loop van de week werd deze wel aangevuld en verder ingedeeld in releases. Deze Story Map wordt ondertussen gebruikt als basis voor de story’s voor de HR-Tool.



Figuur 32 - Story Mapping werking 2



Figuur 33 - Story Mapping resultaat

4.2 Event Storming

3.2.1 Inleiding

Een grote bron van informatie over Event Storming komt voor dit onderzoek uit het gelijknamige boek geschreven door Alberto Brandolini[10]. Belangrijk te noteren is dat het boek nog niet volledig afgewerkt is. Toch is het voldoende uitgeschreven om een uitleg te kunnen formuleren over het Event Storming gebeuren.

Het basisidee achter Event Storming wordt al snel duidelijk. Event Storming is een techniek die vooral draait rond workshops en de volle participatie van alle stakeholders die betrokken zijn bij een project. De duur van zo'n workshop is afhankelijk van de grootte en complexiteit van het project of domein, maar neemt volgens Brandolini meestal wel een volle dag in beslag. Om deze workshop goed te laten draaien is de locatie waarin deze plaats vindt een belangrijke factor. Een ruimte met enkel een tafel en wat stoelen? Totaal niet hoe de analyse van een project moet gebeuren. Schuif die tafel en alle stoelen aan de kant, zorg voor een grote rol papier die op de muur gehangen wordt, pak een stapel plaknotities en een hoop stiften: het is tijd voor Event Storming.

3.2.2 Event Storming in werking

Event Storming begint heel simpel. Voor het starten van een Event Storming sessie worden er altijd een of twee moderators aangewezen die overzien dat alles vlot verloopt. Ook stellen de moderators de deelnemers vragen om ze over bepaalde domeinen na te doen denken of na te gaan of de deelnemers de plaatsing van een bepaald event wel begrijpen.

Per plakbriefje beschrijft de deelnemer een 'key event': dit is een gebeurtenis die plaats vindt in de applicatie. Dit key event wordt genoteerd aan de hand van een werkwoord in de verleden tijd op een liefst oranje plaknotitie. Een voorbeeld hiervan zou "HR received CV" kunnen zijn. Best begint de workshop met het plaatsen van een key event in het midden van de workspace. Zo kunnen er andere domain events toegevoegd worden rondom dit eerste key event en ontstaat er een zekere samenhang. Bij het noteren van key events kan er door moderators opgemerkt worden dat bepaalde deelnemers met elkaar discussiëren over het gebruik of voorkomen van een key event en dat er opmerkingen vallen over zo'n event. De taak van onze eerder aangestelde moderator is om deze discussies op te vangen en een paarse plaknotitie bij het betreffende key event te plaatsen dat het probleem omschrijft. Op deze manier wordt een probleem subtiel ter aandacht gebracht zonder dat de moderator zich moeit of de deelnemers zich aangevallen kunnen voelen. De moderater moet hier beschikken over een goede portie mensenkennis en het oppikken van lichaamstaal. Hierdoor worden key events geïdentificeerd die vaak voor problemen zorgen of moeilijk zijn om te implementeren. Op deze manier kan er goed bepaald worden aan welk key event extra aandacht gegeven moet worden.

Figuur 34 - Event Storming starten

Op bovenstaande afbeelding wordt voorgesteld hoe de sessie er ongeveer uit zou moeten zien. Er zijn enkele key events opgeschreven, maar ook problemen of opmerkingen die de ronde deden over een key event. Als deze basisstructuur opgesteld is worden meer elementen geïntroduceerd om een zekere diepgang te zien in het domein. Aan de hand van blauwe plaknotities wordt het domein event uitgebreid met commando's: deze stellen de acties voor die de gebruikers uitvoeren om tot een key event te komen. Gele plaknotities worden ook toegevoegd om 'actors' voor te stellen. Dit zijn de gebruikers die een actie uitvoeren. Met de toevoeging van deze twee kunnen meer discussies ontstaan over de redenen waarom gebruikers een actie uitvoeren.

Vervolgens schrijven de deelnemers op een roze plaknotitie de externe systemen die een invloed kunnen hebben op of een rol spelen bij de werking van een domain event. Vaak worden bij projecten de externe systemen vergeten, wat later een negatieve impact kan hebben op de werking van de applicatie aangezien deze niet voorzien is op de benodigde externe systemen.

Uiteindelijk wordt de laatste kleur plaknotitie toegevoegd aan de workspace: de paarse of lila notitie. Deze stelt een policy voor: de logica die plaatsvindt nadat een event uitgevoerd wordt en een nieuw commando in werking zet. Zo'n policy begint altijd met het woord "whenever".

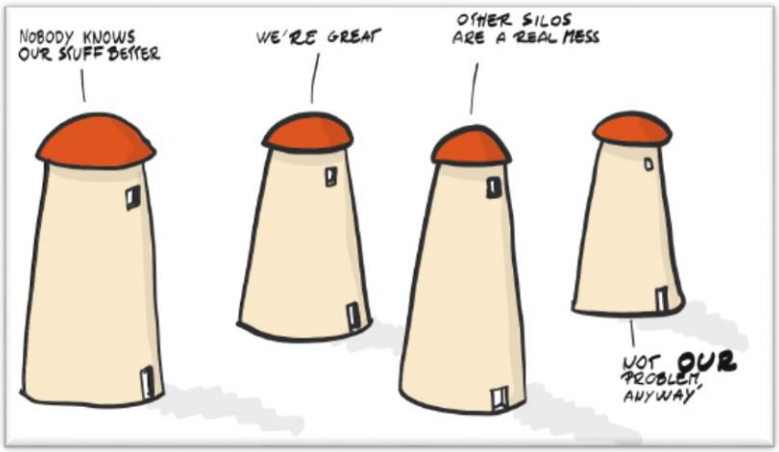
3.2.3 Waarom Event Storming?

De werking van een Event Storming sessie werd reeds besproken. Belangrijk om te weten is welke waarde er nu gecreëerd wordt door het uitvoeren van zo'n workshop.

Allereerst is het belangrijk om te weten dat er verschillende vormen of formaten bestaan rond Event Storming. Het formaat waar we voor dit onderzoek geïnteresseerd in zijn, en dat ook grotendeels aan bod komt in het boek, is de Big Picture Event Storming. Deze dient om een project in grote lijnen uit te wijden, en dit met de participatie van alle betrokken stakeholders.

Een van de problemen die Event Storming dient op te lossen is het silo probleem. Elk bedrijf bevat afdelingen met een gespecialiseerd domeingebied en vaste medewerkers. Vaak zijn deze silo's, zoals die afdelingen genoemd worden, zich enkel bewust van hun eigen processen binnen een bedrijf of project. Ze leggen enkel de focus op hun eigen werk en houden minder en minder rekening met wat er gebeurt in andere afdelingen.

Omdat ze niet erg op de hoogte zijn van alle processen die gebeuren binnen een bedrijf verliezen ze soms het geheel dat moet ontstaan uit het oog. Wanneer er binnen een project dan samengewerkt moet worden tussen silo's ontstaan er vaak problemen en hoge kosten om tot een oplossing te komen. Dit komt omdat we te maken hebben met verschillende technieken en expertises waar niemand echt een globale kennis van heeft.



Figuur 35 - Silo's

Event Storming kan helpen met het verwerven van kennis tussen de silo's door het vormen van een beeld dat alle business processen beschrijft op grote schaal, met overlapping van de silo's. Dit verworven begrip is waarschijnlijk de belangrijkste waarde van een big picture sessie.

Event Storming komt uiteindelijk neer op de volgende stappen:

1. Zie het systeem als een geheel
2. Vind een probleem
3. Zoek de best mogelijke informatie over dit probleem
4. Implementeer de oplossing vanuit het best mogelijke startpunt

3.2.4 Wanneer Event Storming?

Event Storming kan in elke fase van een project uitgevoerd worden. De uitkomst van een Event Storming sessie heeft als doel het bekomen van een oplossing en het beter begrijpen van een project. Hierdoor zal er geen echte backlog van een project ontstaan, maar eerder een algemeen begrip van het project en zijn pijnpunten.

In principe is Event Storming op elk moment van een project gunstig. Aan het begin van een project kan Event Storming de pijnpunten helpen identificeren waardoor hier focus op gelegd kan worden. We leggen dan het startpunt van ons project bij dat event waar het meeste problemen in voorkomen. Wanneer het project al volop in werking is kan een Event Storming sessie bijdragen tot het vinden van een oplossing. Dit omdat gebreken of events waar geen rekening mee gehouden wordt in kaart gebracht kan worden. Op het einde van een project kan Event Storming helpen bij het begrijpen van het geheel, maar ook een confirmatie zijn dat er geen gaten of gemiste requirements in het project geslopen zijn.

3.2.5 Proof of Concept

Samen met JIDOKA werd er ’s avonds een workshop georganiseerd om al deze theorie over Event Storming ook eens uit te testen in de praktijk. Hierdoor konden we pas echt bepalen hoe zo’n sessie nu in zijn werk gaat.

Als onderwerp voor de workshop gingen we de recruitment flow van JIDOKA behandelen, dit betekent dus het volledige proces van het ontstaan van een vacature tot het aannemen en boarden van een nieuwe werknemer. Vervolgens werden de aanwezigen in 2 groepen gesplitst: een groep die het domein opstelde en een andere groep die de taak van facilitators of moderators op zich namen. Trouw aan het voorschrift van Brandolini stelde we ook een kleine legende op die de kleuren van onze plaknotities duidelijk zou maken:

- Oranje: Event
- Roze: Probleem/Opmerking
- Lichtgeel: Data (Extern Systeem)
- Geel: Actors

Figuur 36 - Event Storming legende

De icebreaker is een persoon tussen de deelnemers die instaat voor het opstarten van de volledige sessie. In het begin kunnen de deelnemers wat angstig zijn om direct te participeren, de icebreaker begint met het plakken van een domain event in het midden van onze workspace. Hierop kan de rest van de groep anticiperen en schiet de sessie volledig in gang. Al snel begonnen events en notities vlijtig op de muur te verschijnen rondom dat eerste event en spoedig merkte we dat we naar een grotere muur moesten verhuizen. Een van de nadelen van deze nieuwe locatie was wel het plaatsgebrek. Belangrijk voor een Event Storming sessie is dat er voldoende ruimte is voor iedereen om te participeren. We hebben bij onze Proof of Concept gemerkt dat dit effectief een belangrijke factor is. Omdat er te dicht bij onze muur een tafel stond was het niet mogelijk voor iedereen om vlot mee te werken aan het model. Uiteraard is dit geen cruciaal probleem, eerder een handigheid.

Figuur 37 - Event Storming werking 1

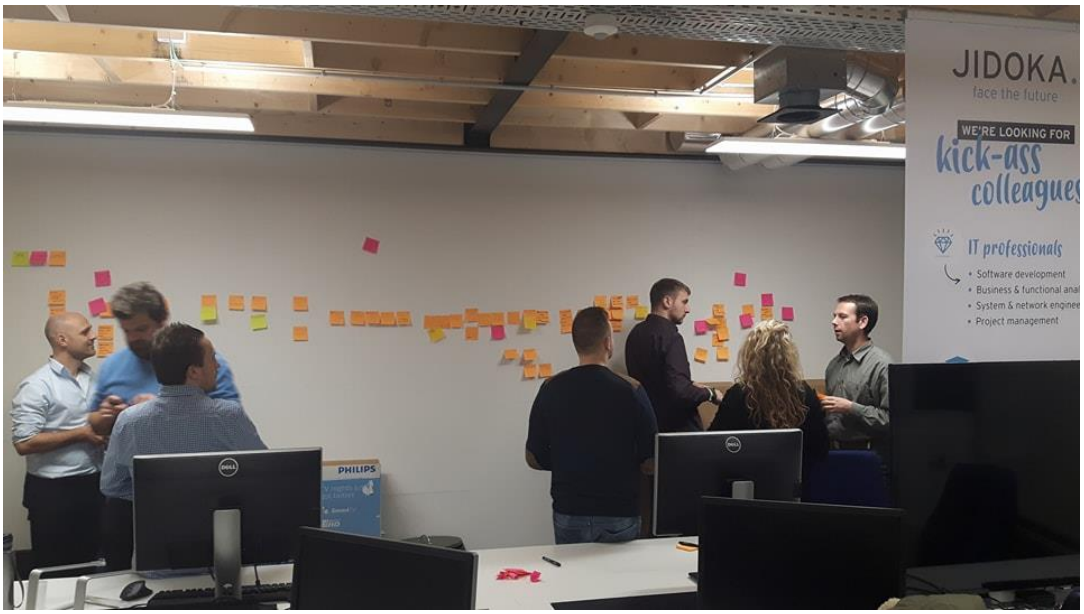
Het aantal notities liep al snel hoog op, tot we op een punt kwamen dat er gediscussieerd werd over bepaalde events en waar ze eigenlijk zouden moeten plaatsvinden in de flow. Het is namelijk van belang dat er enige chronologische volgorde van events ontstaat op onze workspace. Zo kwamen de eerste roze notities, de opmerkingen, opduiken. Deze dienen om op een subtiele manier aan te duiden dat er ergens een probleem zich voordoet in het domein. Dit door een gat in de flow of een onduidelijkheid bij een event. Zo'n onduidelijkheid kan slaan op een fout in volgorde, geen voorgaand event dat tot het huidige event leid of het ontbreken van een duidelijke actor. Het plakken van elke roze notitie trok behoorlijk wat aandacht. De deelnemers waren onmiddellijk benieuwd welke opmerking er over het domein gemaakt werd. Op deze manier werd er gezamenlijk snel een oplossing gevonden voor het gegeven probleem.

Om ons domein beter te begrijpen en enkele problemen met actors op te lossen introduceerden we de gele notities die de actors voorstellen. Deze notities plaatsten we bij events die specifiek gericht waren op of uitgevoerd werden door een bepaalde actor.

Bij het overlopen van ons domein kwamen we tot de vaststelling dat er in ons verhaal geen externe systemen aan de pas komen. We konden deze notities dus achterwege laten.

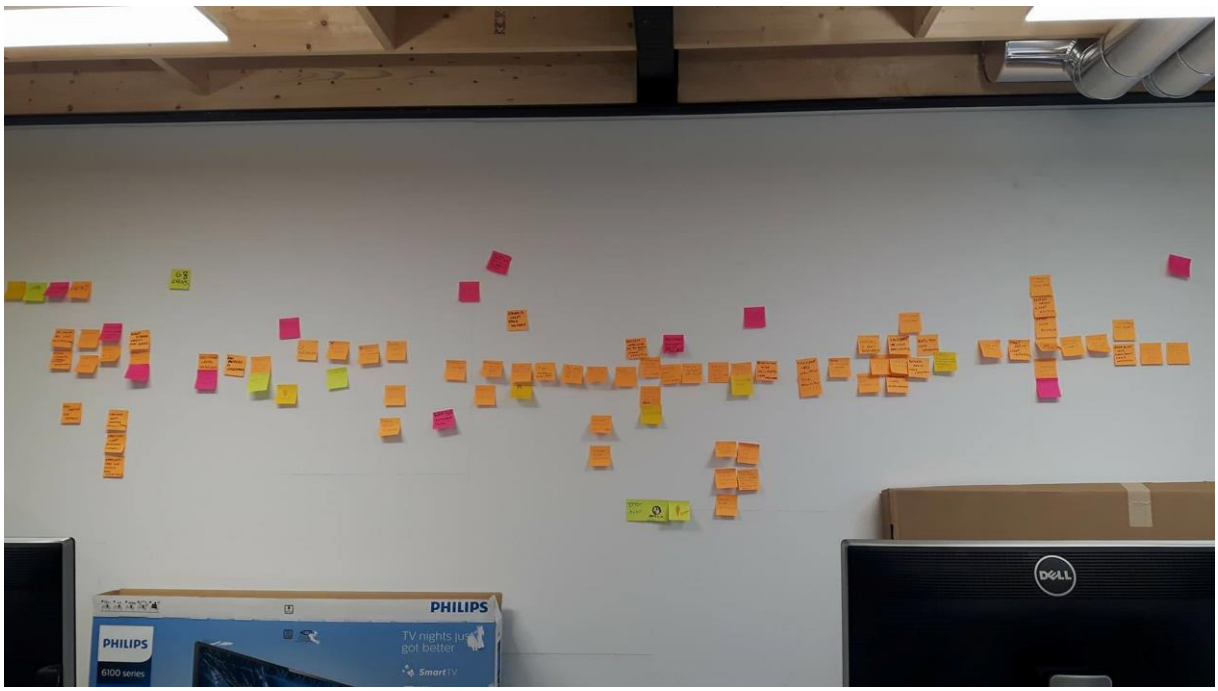
Na een goede twee uur hadden we het grote deel van de recruitment flow op de muur gekregen. Omdat het toevoegen van notities een beetje was stilgevallen gingen we over naar een controletechniek om gaten op te sporen: een van de deelnemers moet de volledige flow vertellen als een verhaal. Indien het verhaal volledig verteld kan worden, zonder dat er een hapering of gat voorkomt, zit het domein al goed in elkaar. In ons geval zat het domein goed, ons verhaal kon in grote lijnen verteld worden zonder dat er iets ontbrak.

Deze techniek kan overigens ook in de omgekeerde richting uitgevoerd worden, het verhaal wordt dan van einde tot begin verteld. Het nut van dit omgekeerd te doen is dat er soms achterhaald kan worden dat een bepaald event bijvoorbeeld geen oorsprong heeft, dit moet dan opgevuld worden.



Figuur 38 - Event Storming werking 2

Na verder discussiëren over het domein begonnen we tot een resultaat te komen waarin iedereen zich kon vinden. De volledige flow was visueel neergezet door Event Storming. We kwamen uiteindelijk tot onderstaand resultaat:



Figuur 39 - Event Storming resultaat

5 Conclusie

5.1 Conclusie

Een onverwachte conclusie van dit onderzoek naar Event Storming en Story Mapping is dat beide technieken op een project toegepast kunnen worden. Uit onderzoek en uitvoering is gebleken dat beide complementair gebruikt kunnen worden.

Event Storming geeft een visuele representatie van de processen en domeinen binnen een project. Het creëert een algemeen begrip onder alle stakeholders en legt de grote problemen van een project bloot. Maar dit is allemaal high level, er ontstaat geen concrete backlog.

User Story Mapping vangt dit op. Met deze techniek wordt er gedetailleerder ingegaan op processen. Niet enkel worden deze processen visueel uitgelijnd maar worden ze ook nog eens geprioriteerd volgens de waarde die ze bieden tot de applicatie.

Een groot verschil zit dus in de scope. Bij Story Mapping leg je de weg bloot om een gericht product te realiseren, en dit kan je zelfs indelen in verschillende releases. Event Storming draait meer om het ontdekken van de grote omkadering rond of de context van het product of de organisatie.

De meest ideale setting zou zijn wanneer er bij de start van een project begonnen wordt met Event Storming. De uitkomst van die sessie kan dan verder uitgebreid worden door er Story Mapping op los te laten.

Op deze stageopdracht was Story Mapping voldoende geweest om de story's en gebreken te identificeren.

5.2 Persoonlijke Reflectie

Ik moet eerlijk zeggen dat zowel het vergaren van informatie over dit onderwerp, als het omzetten van theorie in workshops, een enorm aangename ervaring was.

Dit is volgens mij ook een onderzoek dat me enorm veel helpt in de toekomst. Als Software Manager, en zeker als Project Manager, is User Story Mapping iets dat gewoon niet vermeden kan worden. Vroeger of later heb je er altijd mee te maken. Ik zal in de toekomst dus veel hebben aan de ervaring die ik nu al heb opgedaan met deze techniek, zowel voor het opstellen van een backlog als voor een vlotte communicatie met klanten.

Ook het onderzoeken van Event Storming, een techniek die nog aan het opkomen is, was zeer verrijken Tijdens de workshops heb ik al opgevangen dat deze techniek al bij enkele bedrijven toegepast wordt op grote projecten. Ik geloof dat ik dus veel baat heb gehad bij dit onderzoek.

Bibliografie

[1] Atlassian, „The #1 software development tool used by agile teams”, Atlassian, 2017. [Online]. Available: https://www.atlassian.com/software/jira.

[2] Slack, „It’s the foundation for teamwork”, Slack, 2017. [Online]. Available: https://slack.com/features.

[3] Bitbucket, „For the code that drives us”, Atlassian, 2017. [Online] Available: https://bitbucket.org/.

[4] OAuth, „OAuth 2.0”, OAuth, 2017. [Online] Available: https://oauth.net/2/.

[5] Jason Watmore, „Angular 2/5 – Custom Modal Window / Dialog Box”, Jason Watmore’s Blog, 24 januari 2017. [Online]. Available: http://jasonwatmore.com/post/2017/01/24/angular-2-custom-modal-window-dialog-box.

[6] Sunit Parekh, „Story Mapping, Visual Way of Building Product Backlog”, ThoughtWorks, 12 januari 2015. [Online]. Available: https://www.thoughtworks.com/insights/blog/story-mapping-visual-way-building-product-backlog

[7] Steven A. Lowe, „An introduction to event storming: The east way to achieve domain-driven design”, TechBeacon, [Online]. Available: https://techbeacon.com/introduction-event-storming-easy-way-achieve-domain-driven-design

[8] Mike Ensor, „Event Storming: A powerful Tool for Solving Business Problems”, Washington Technologie, 18 april 2017. [Online]. Available: https://www.washingtontechnology.org/event-storming-a-powerful-tool-for-solving-business-problems/

[9] Jeff Patton, „User Story Mapping”, O’Reilly Media, 1 oktober 2015.

[10] Alberto Brandolini, „Introducing Event Storming”, Leanpub, laatste update 16 januari 2017. [Online]. Available: https://leanpub.com/introducing_eventstorming.