|  |
| --- |
| **Project Reports – MaxServ** |

|  |  |
| --- | --- |
| Onderwerp | (Support)Rapportage Website voor MaxServ |
| Gereedschapskist | HTML (PhpStorm)  CSS (PhpStorm)  JavaScript/jQuery (PhpStorm)  PHP (PhpStorm)  MySQL (Servers op geïntegreerd Ubuntu in Windows)  TYPO3 (CMS op geïntegreerd Ubuntu in Windows) |
| Projectontwikkeling | Plannen 🡪 Voorbereiden 🡪 Realiseren 🡪 Testen 🡪 Optimaliseren |
| Leerdoelen | * B1-K1-W2: Levert een bijdrage aan het projectplan * B1-K1-W4: Bereidt de realisatie voor * B1-K2-W1: Realiseert (onderdelen van) een product * B1-K2-W2: Test het ontwikkelde product * B1-K3-W1: Optimaliseert het product * Persoonlijk leerdoel 1: Aan het einde van stage in staat zijn volgens de Scrum methode mijn tijdsbesteding in te schatten om deadlines te behalen * Persoonlijk leerdoel 2: Aan het einde van stage in staat zijn om back-end-gerelateerde problemen op te lossen door syntax- en semantische fouten in code te kunnen aflezen, opsporen en te verhelpen |
| Context | Paul (support engineer) moet nu meerdere stappen doornemen om een overzicht te maken met daarin uren-in-overeenkomst, tarief per uur, gebruikte tickets (taken dat gedaan moet worden) voor de klant.  Omdat dit allemaal handmatig gedaan wordt, maakt het alle informatie foutgevoelig waardoor er bepaalde data niet overeen kan komen met de afspraken: uren dat klant betaald (ook bekend als strippenkaart), taken (tickets in JIRA).  Uit JIRA haalt hij alle supportprojecten op waarin hij alle bijhorende gegevens uithaalt. Vervolgens maakt hij handmatig daaruit tabel voor zichzelf.  Door de tabel uit te lezen zet hij de gegevens om in een Excelsheet waarvan meeste velden automatische door formules worden ingevuld zodra hij de gegevens invult. In het Excelsheet zet hij ook bijhorende grafieken bij.  Dat allemaal zet hij om in een Pdf-bestand en wordt naar de klant verstuurd.  Om Paul deze zaken eenvoudiger en comfortabeler te maken wordt er gevraagd om een bestaande website dat al die zaken systematisch en overzichtelijk bijhoudt. Aan de development team (ik) om de website verder uit te breiden en eventueel te verbeteren.  Tenslotte is het ook de bedoeling dat in de toekomst de directie(management) en ook zelfs klanten in de website kunnen komen om de (support)rapportages met bijhorende statistieken in te zien. |

|  |
| --- |
| **Inhoudsopgave** |

Voorblad Pagina 1  
  
Omschrijving bedrijfssituatie Pagina 3  
  
Projectontwikkeling/processen binnen project Pagina 6

Handleiding OTAP Pagina 7  
  
Product Backlog Pagina 19

Sprint Planning Pagina 20  
  
Sprint Backlog Pagina 20   
  
Sprint Review Pagina 21   
  
Sprint Retrospective Pagina 22

|  |
| --- |
| **Omschrijving bedrijfssituatie** |

Binnen MaxServ wordt de Scrum-methode gebruikt om optimaal aan projecten te werken.

**Het Scrum Team:**

Dit zijn de partijen die direct betrokken zijn met het project:

De **Product Owner** (vaak is dat de klant en altijd degene die wensen en eisen stelt aan het product) is verantwoordelijk voor het onderhouden van de **Product Backlog**.

Het ontwikkelteam (**Developement Team**) die (onderdelen van) het product maken of verbeteren.

De **Scrum Master** die voortuitgangen stimuleert binnen het werkproces van het project.

Voordat het project begint moet er een lijst gemaakt worden waarin duidelijk staat wat er gedaan moet worden voor het product. Dit heet volgens Scrum de **Product Backlog**.

De **Product Backlog** bestaat vaak uit epics (grote stukken verhalen) en/of user-stories (kleine informele lijstjes met wensen en eisen).

Een User Story voldoet aan de Definition of Ready als aan de volgende voorwaarden is voldaan:

* De user story is omschreven in de vorm: *als ... wil ik ... zodat...*, met als doel dat alle leden van het team begrijpen wat er moet gebeuren.
* De user story is voorzien van duidelijke acceptatie criteria.
* De user story bevat alle benodigde zaken zoals specificaties, wire frames, designs etc.
* De user story bevat de onderliggende taken
* De user story is gepokerd en kan eenvoudig binnen 1 sprint worden afgerond
* De user story heeft geen externe afhankelijkheden meer
* De user story heeft geen andere impediments meer

De **Product Owner** is verantwoordelijk voor het aanmaken van user stories in de backlog. Echter zal het team de **PO** tijdens de refinement moeten helpen om de stories in de juiste vorm te krijgen. Alleen dan kan het team inschatten hoeveel werk iedere story is en hoeveel punten ze in een sprint kunnen wegwerken.

Hierna volgt een **refinement** sessie waarin de taken verdeeld worden in sub-taken. Ook worden hier vaak aan ieder taak/user-story een **acceptatie-criteria** eraan toegevoegd om de gewenste resultaten (van het product) te bereiken en afspraken met de klant na te komen.

Voordat men überhaupt aan een **Sprint** kan beginnen, moeten de user-stories/ epics alsnog voldoen aan de **Definition of Ready**.

De volgende stap is de **Sprint Planning**. Samen met het team worden de taken verdeelt in termijnen van minstens twee weken en hooguit een maand. Hierin geef je antwoord op de vraag: hoe bereik je de gewenste resultaten van een product?

Dit allemaal wordt helder in beeld gebracht in de **Sprint Backlog**, dat is slechts een overzicht met alle **Sprints**.

Een **Sprint** is een werkproces dat bestaat uit verschillende stappen die doornomen moeten worden om het doel van het project te bereiken. Dit zijn o.a. een **TO-DO, IN-PROGRESS, TEST, ACCEPTATIE, DONE.**

De werkzaamheden van de huidige dag, betreffend de taken, worden dagelijks (hoogstens 15 minuten) besproken in een **Daily Scrum.**

Aan het einde van een **Sprint** worden de resultaten van het product behandelt. Dit heet een **Sprint Review** en is een informele bijeenkomst van alle betrokken bij het project.

Hier is plaats voor feedback op de afgelopen werkzaamheden en tegelijkertijd wordt er gekeken naar verbeteringen voorin de volgende **Sprints**. Het verloop van een afgelopen **Sprint** wordt verder besproken in een **Sprint Retrospective**.

Als alle voorgaande processen zijn afgerond, dan is dit een geval van een “**Definition of Done**”, zoals men het in MaxServ noemen.

Hieronder zijn voorbeelden van **Definition of Done**:

**Bij een user story:**

* Alle code is ingecheckt.
* De status van het Jira ticket is bijgewerkt.
* Het ticket is voorzien van een uitleg hoe het getest moet worden.
* Alle in het JIRA ticket vermelde teststappen (zowel handmatig als automatisch) zijn succesvol afgerond.
* Configuratie instellingen worden geautomatiseerd doorgevoerd en hoeven niet handmatig ingesteld te worden op iedere omgeving.
* Er is waar nodig een draaiboek opgesteld.
* Er is waar nodig documentatie bijgewerkt.
* Er is testdata / content ingevoerd.
* Conform implementatie specificaties.
* Verantwoordelijkheid content is belegd.
* Redirects zijn ingeregeld indien nodig.
* Voldoet aan WCAG / Drempelvrij.
* De story is in alle vereiste browsers getest zijnde:
  + IE 11 en Edge
  + Safari (laatste 2 versies)
  + Chrome (laatste 2 versies)
  + Mobile iOS browser (iPhone & iPad)
  + Mobile Android browser / Chrome

**Bij een sprint:**

Er is aan alle story criteria voldaan. Daarnaast is:

* Er een succesvolle frontend performance test op de acceptatie omgeving uitgevoerd.
* Backend pagina's verschijnen binnen 10 seconden op het scherm.
* Er is een demo gegeven.

**Bij een release naar acceptatie:**

Er is aan alle story criteria voldaan. Daarnaast is:

* Alles ingericht op de acceptatie omgeving
* Werkt de website onder SSL.
* Hebben alle pagina's SEO-friendly URL's

**Bij een release naar productie:**

Er is aan alle criteria voor een release naar acceptatie voldaan, daarnaast:

* Is alle content aanwezig.
* Is de final checklist nagelopen.
* Content is meetbaar
* Documentatie is aanwezig
* Rechten/Juridische toetsing

Uiteindelijk is het de bedoeling dat er iets bruikbaars komt uit de resultaten van het product. Dit noem je een **Increment** en dat wordt vervolgens toegevoegd aan het eindproduct wat de klant te tevreden moet maken.

Maar dit is NIET PER DEFINITIE het einde van het project. Omdat er meerdere taken zijn die voltooid moeten worden, bestaat een project altijd uit meerdere sprints. Als een project afgerond is of zelfs nog loopt, is het meestal geval dat het team ook aan andere sprints van andere project(en) werken.

|  |
| --- |
| **Projectontwikkeling** |

|  |  |  |
| --- | --- | --- |
| **Stap** | **Omschrijving** | **(Scrum) Artefacten** |
| 1 | Maakt/bedenkt user-stories en/of epics voorin de Product backlog | * Product Backlog |
| 2 | Gaat de user-stories/epics uit de product backlog omzetten in sprints en plant het in termijnen | * Sprint Planning * Sprint Backlog. |
| 3 | Voert het increment door aan het einde van een sprint waaruit een bruikbaar product komt. Ook wordt er een gemaakt verslag over de resultaten die zijn gebleken uit het gewerkte product en project. | * Increment/eindproduct * Sprint Review |
| 4 | Maakt een verslag met daarin de feedback en reflectie op de afgelopen sprints | * Sprint Retrospective |

|  |
| --- |
| **Processen binnen project** |

1. **Intake**

Hier komen alle taken terecht, in afwachting tot goedkeuring.

1. **To refine**

De taken worden voorzien van aanpassingen aan de beschrijving/referenties/acceptatie-criteria, indien nodig.

1. **Ready for development**

De taken zijn nu klaar om uitgevoerd te worden.

1. **In Progress**

Er wordt actief gewerkt aan de taken.

1. **Needs review**

Een senior-developer of andere developers moeten goedkeuring geven m.b.t. de source code en de werking daarachter.

1. **Needs deployment**

Er moet een acceptatie-/testomgeving (uitgesloten voor developers en/of klanten) komen waarnaar (onderdelen van) de website uitgerold kan worden.

1. **Acceptance /Testing**

Zodra er een acceptatie-/testomgeving beschikbaar is voor de website, moet(en) (onderdelen van) de website daarop getest worden.

1. **Ready for release**

Als het na het testen blijkt dat (de onderdelen van) de website geen problemen bevat en op orde is, wordt de website uiteindelijk uitgerold naar de live-/productieomgeving (openbaar, iedereen kan het zien met uitzonderring van eventueel afgesloten webpagina’s) waar de klant kan zien en gebruik van kan maken.

1. **Done**

Als de website eenmaal op de live-/productieomgeving zit en de klant keurt de website goed, dan is het afgerond. Op het moment dat alle taken af zijn of de sprint zelf afgelopen is, kan men beginnen aan een nieuwe sprint.

|  |
| --- |
| **Handleiding OTAP-omgeving** |

**Let op! Dit is een handleiding voor OTAP-omgeving, dat overgenomen is vanuit bestaande documentatie, bedoeld voor beginnende developers**

Als nieuwe developer binnen een MaxServ team komt er veel op je af. Een nieuw team, nieuwe overtuigingen en nieuwe processen. MaxServ wil alle teams volgens dezelfde methodiek laten werken en vandaar dat het hele development proces is omschreven. We proberen zo veel mogelijk gebruik te maken van proven technology en ons te houden aan bekende standaarden.

We beginnen met een schematische weergave van het development proces. We starten bij het voorbereiden van een userstory en we eindigen bij het opleveren op de productie-omgeving van het project. Iedere stap zullen we daarna ook tekstueel bespreken.

**Let op! Deze stappen zijn bedoeld voor ontwikkeling binnen een project. De stappen voor ontwikkeling / aanpassingen binnen support-uren zijn anders.**

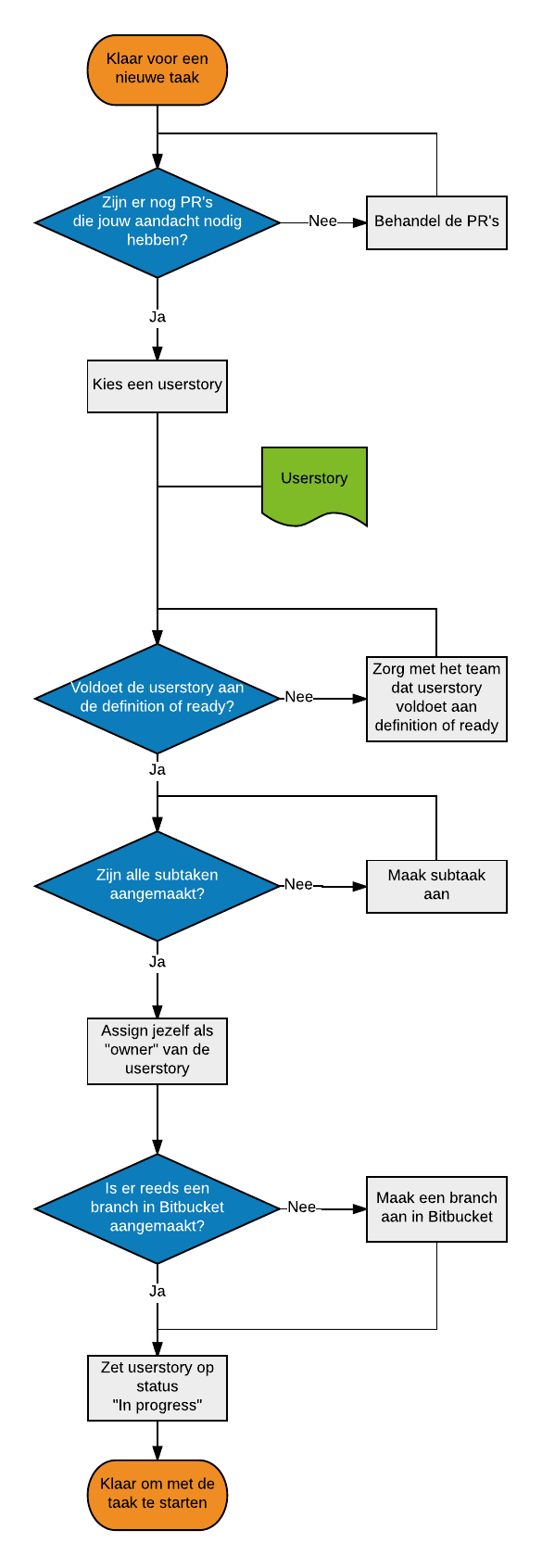
**1 Voorbereiding nieuwe taak**

Voordat je gaat beginnen aan een nieuwe taak moet er gekeken worden of de taak al volledig is om hem daadwerkelijk te gaan uitvoeren.

1. Allereerst controleer je of er nog openstaande PR's / releases zijn waarin jouw bijdrage vereist is. Is dit het geval, behandel je eerst deze PR's.
2. Zoek een userstory binnen de actieve sprint van het project in JIRA die het hoogst in de lijst staat en waar jij nog een bijdrage kan leveren.
3. Allereerst moet je controleren of de userstory voldoet aan de definition of ready? Is dit nog niet het geval, overleg dan met het team hoe de userstory daaraan kan gaan voldoen.  
     
   *Normaal gesproken zouden de userstory al voor de start van de sprint moeten voldoen aan de definition of ready en zou dit eigenlijk nooit voor moeten komen.*
4. Zijn alle subtaken al aangemaakt binnen de userstory? Overleg met het team welke subtaken nodig zijn om de userstory volledig af te ronden. Denk hierbij aan subtaken als: "Content-element in de backend beschikbaar maken", "Gebruikers rechten geven op element", "Opmaken van content-element conform design". Ga hiermee door tot alle subtaken zijn aangemaakt.
5. Als alle acceptatie-criteria zijn opgegeven en alle subtaken zijn aangemaakt, bekijk je nog even of de userstory nu voldoet aan de definition of ready. Heb je alle informatie die je nodig hebt om de userstory te kunnen uitvoeren? Nog niet? Zet dan in JIRA een vlag op het issue en ga samen met de product-owner kijken of je deze zaken kunt oplossen. Zijn alle benodigdheden binnen, loop dan het hele proces nog een keer na want wellicht zijn de acceptatie-criteria of de subtaken nu niet helemaal dekkend meer.
6. Assign jezelf als "owner" van de userstory. Vanaf nu ben je verantwoordelijk dat er voortgang zit in de userstory.
7. Controleer vervolgens of er al een feature branch in de GIT-repository is voor deze userstory. Zo ja, ben je klaar om verder te gaan naar de volgende stap: de uitvoering. Is er nog geen branch voor de userstory? Maak deze dan aan in Stash (officieel tegenwoordig Bitbucket). Let daarbij op het aftakken:
   1. Nieuwe features, verbeteringen en normale bug-fixes **binnen een lopend project** takken we af van **Development**.
   2. Nieuwe features, verbeteringen en normale bug-fixes waar er **geen lopend project** is, takken we af van **Master**.
   3. Hotfixes, security-updates etc takken we af van Master.

De naam van de branch moet minimaal het ticketnummer bevatten van de userstory.   
  
*Tip: als je de branch aanmaakt vanuit je JIRA-ticket, krijgt hij automatisch een goede naam.*

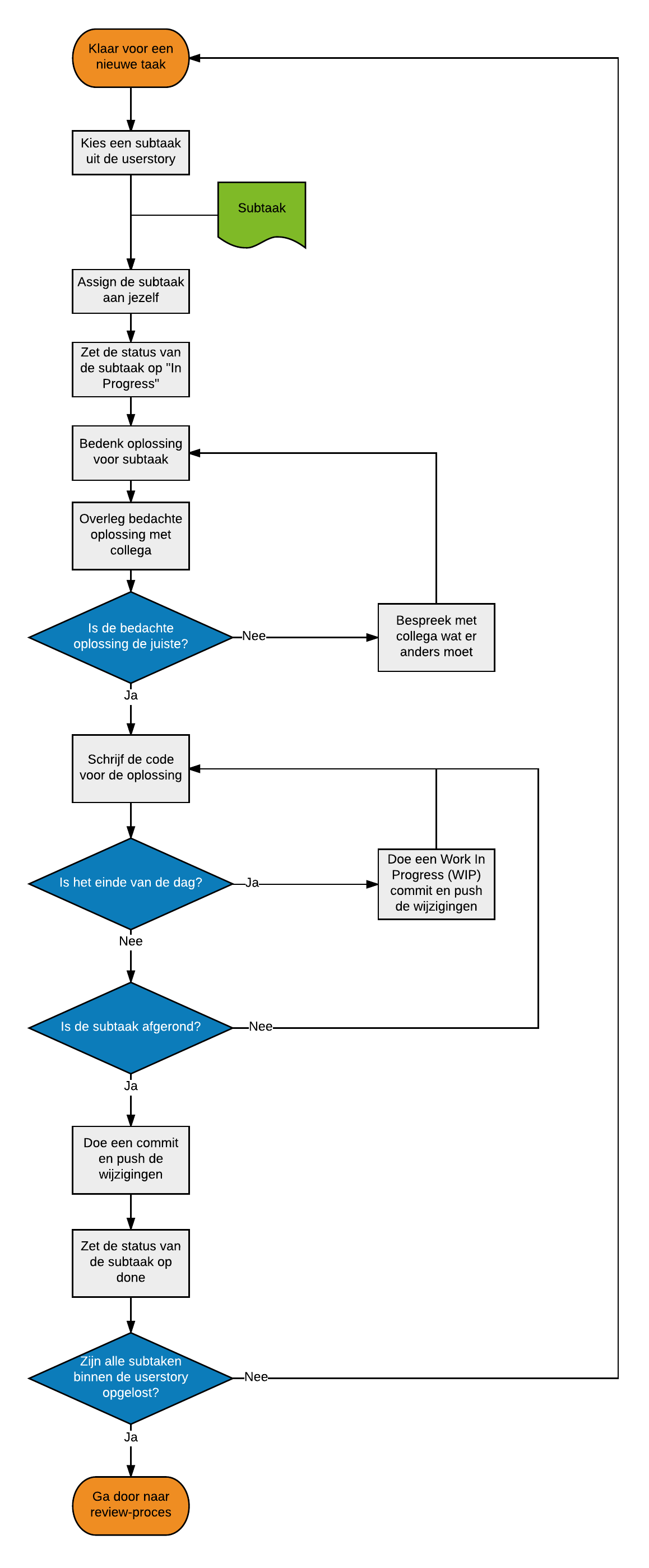
1. Als de branch is aangemaakt, zet je de userstory op de status "In progress" en kan je doorgaan met de volgende stap: De uitvoering



**2 Development**

In de vorige stap is een userstory gekozen en is deze userstory volledig aangevuld met de informatie die nodig is om de userstory en daarmee de subtaken uit te voeren. In de daadwerkelijke uitvoering zullen we ook weer een aantal stappen doorlopen.

1. Kies uit de userstory een subtaak die je wilt gaan oppakken en assign hem in JIRA aan jezelf. Zet de taak vervolgens op de status "In progress".
2. Bedenk in je hoofd een oplossing hoe je het "probleem" uit de taak gaat oplossen. Maak hier desnoods wat schetsjes voor en overleg het met een ander teamlid. Als het teamlid het idee heeft dat het de juiste weg is, ga je verder met de ontwikkeling. Anders probeer je samen tot een betere oplossing te komen.
3. Je kunt nu in de branch die je in de voorbereiding voor de userstory hebt gemaakt nu je daadwerkelijke code gaan schrijven.
4. Zodra je klaar bent met het schrijven van je code en je jouw subtaak hebt afgerond zorg je dat al de code gecommit is in de specifieke branch. Je commit-message moet daarbij aan de volgende "eisen" voldoen:
   1. Bij een "normale" commit moet het bericht in het formaat: "[TICK-123] Added field subheader to CE Text with icon" en moet in het Engels zijn.
   2. Als je een commit doet maar nog niet klaar bent met dat stuk code vermeldt je dat door de term WIP (Work In Progress) te gebruiken. Het bericht wordt dan in het formaat: "[TICK-123][WIP] Added possibility to add a map with markers on a page"
5. Zet de subtaak vervolgens op de status "Done". **Let op!** Het gaat om de subtaak en dus NIET de userstory.
6. Overleg in het team of alle subtaken zijn afgerond. Zo ja, ga dan door naar de volgende stap: Reviewing. Zijn nog niet alle subtaken afgerond, geef dan aan binnen het team dat ze verder kunnen met de andere subtaken.



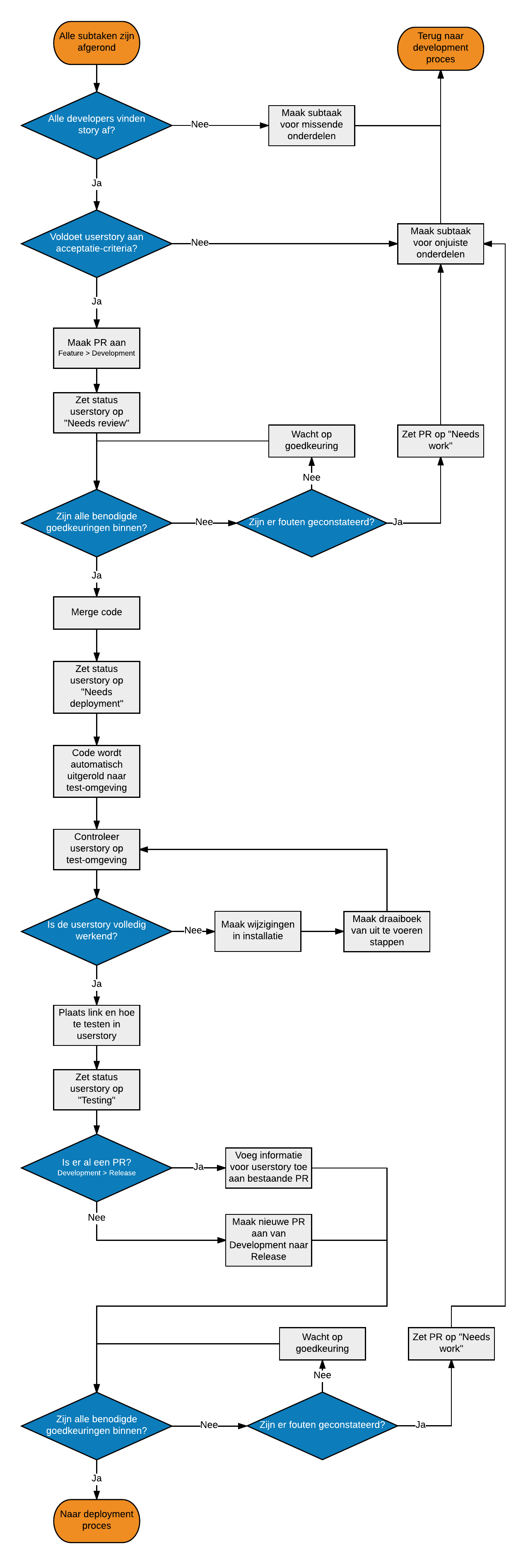
**3 Reviewing**

Als alle subtaken zijn afgerond gaan we het review-proces in. Ook hierin hebben we weer een stappenplan opgesteld.

1. Bespreek met alle teamleden die aan de userstory gewerkt hebben of iedereen er van overtuigd is dat de userstory zo goed is . De "owner" van de userstory (die in stap 1 - voorbereiding van het proces is bepaald) neemt hierin de leiding. Is alles volgens de teamleden in orde? Dan ga je verder met het review-proces. Zo niet, maak een extra subtaak aan en ga terug naar stap 2 - uitvoering van het proces.
2. Loop als owner van de userstory na of de userstory aan alle acceptatie-criteria voldoet. Is dit het geval dan ga je verder met het review-proces en anders maak je een nieuwe subtaak aan om de userstory aan de acceptatie-criteria te laten voldoen. Beschrijf in deze subtaak wat er niet goed is en waar je dit hebt getest. Vervolgens ga je terug naar stap 2 - uitvoering.
3. Als alle teamleden een go hebben gegeven en de userstory aan alle acceptatiecriteria voldoet, moet er een pull request (PR) gemaakt worden in Stash. Deze moet van de branch voor de userstory richting de development branch (of naar release indien het een hotfix betreft). De titel van de PR bestaat uit het ticketnummer en titel van de userstory. De omschrijving van de PR moet de volgende onderdelen bevatten:
   * Wat is er uitgevoerd?
   * Waar kan het getest worden?
   * Zijn er nog bijzonderheden?

Vervolgens worden aan deze PR alle developers binnen het projectteam als reviewer gemarkeerd.

1. Als de PR is aangemaakt, zet je de userstory op de status "Needs review". Jij als owner van de userstory bent vanaf nu verantwoordelijk dat de PR zo snel mogelijk wordt getest door je teamgenoten.
2. Afhankelijk van de inhoud van de PR dient er minimaal 1 senior goedkeuring te geven per dicipline (frontend, backend, operations) die binnen de userstory valt. Is het een gecombineerde front- en backend story, dan moet er dus minimaal 1 senior frontend en 1 senior backend developer zijn goedkeuring gegeven hebben. Zijn de benodigde goedkeuringen binnen? Dan ben je als owner verantwoordelijk dat de code gemerged wordt in Stash.
3. Zet de userstory nu op de status "Needs deployment"
4. Zodra de PR is gemerged naar development, zal hij automatisch worden gedeployed naar de test-omgeving. Houdt hiervoor je HipChat / Slack kanaal in de gaten waarin de output van Bamboo wordt geplaatst. Zodra hij gedeployed is, ga je verder naar het volgende actiepunt.
5. Controleer of de wijziging te zien zijn op de test-omgeving. Je zorgt er vervolgens voor dat op deze omgeving de userstory te testen is en noteert de stappen om het te kunnen testen in een draaiboek.
6. Zodra de userstory is ingericht op de test-omgeving zet je de status van de userstory op "Testing".
7. Controleer of er reeds een PR van development naar release is. Zo ja, voeg dan de informatie (waar te testen, hoe toe te passen, link naar draaiboek) omtrent de op te leveren userstory toe aan de omschrijving van de PR.   
   Is er nog geen PR, dan maak je die aan en vermeldt je de informatie (waar te testen, hoe toe te passen, link naar draaiboek). Geef hierbij de scrum-master, de betrokken ontwikkelaars in de PR en degende die de releases verzorgt op als reviewer. De titel van de PR moet zijn: [TICKET-NR] Acceptatie-release [datum]. Hierbij gebruik je als ticketnummer het ticket van het release-management (van deze maand) voor het project en als datum de datum waarop het gereleased moet worden.
8. Iedere reviewer controleert de werkzaamheden of deze juist zijn en geeft zijn goedkeuring aan de PR. Zijn de werkzaamheden niet goed, zet dat de PR op de status "Needs work".

 **Deployment naar acceptatie-omgeving**

 **4 Deployment**

Om de daadwerkelijke gemerde code ook uitgerold te krijgen naar de verschillende omgevingen in Ontwikkel-, Test-, Acceptatie- en Productie-omgeving (OTAP) werken we met releases en deployments in Bamboo.

**4.1 Deployment naar test-omgeving**

De depolyment naar de test-omgeving gebeuren automatisch zodra een feature-branch wordt gemerged in de Develoment-branch. Hier komt geen handmatige actie bij kijken naast het mergen naar development. Let er wel op dat eventuele inrichtings-zaken wel handmatig doorgevoerd moeten worden op de test-omgeving.

**4.2 Deployment naar acceptatie-omgeving**

De acceptatie-omgeving is in sommige gevallen geplaatst op dezelfde fysieke server als de productie-omgeving. Het is dus belangrijk dat er goed gekeken wordt naar de impact van een change. Voor een release naar de acceptatie-omgeving moeten de volgende stappen doorlopen worden.

1. In stap 10 van het review-proces is er een PR gekomen van de development branch naar de release branch. Als iedere reviewer de PR heeft goedgekeurd, kan je verder naar de volgende stap. Missen er om wat voor reden dan ook goedkeuringen op de PR maar is hij bijvoorbeeld wel goedgekeurd door een senior Developer, overleg dan even met die senior.
2. Is er een draaiboek aanwezig? Zo ja, neem deze door zodat je weet wat er speelt. Raadpleeg bij vragen de schrijver van de betreffende draaiboek-actie.
3. Merge de PR en wacht tot bamboo de release-branch succesvol heeft kunnen builden.
4. Bij een succesvolle build van Bamboo ga je vanuit de betreffende build een release maken.
5. Kies vervolgens de omgeving waar je naar wilt uitrollen, in dit geval de acceptatie-omgeving.
6. Deploy nu de zojuist gemaakte release
7. Zodra Bamboo klaar is met de deployment (hoe snel dit is hangt af van beschikbaarheid en het type project) zal hij een melding hiervan geven in het HipChat-kanaal. Ook zal het scherm waarin je de build hebt gestart een melding geven dat de deployment geslaagd is.
8. Controleer of de release daadwerkelijk geslaagd is en de acceptatie-omgeving nog bereikbaar is. Test in ieder geval de volgende onderdelen:
   1. Homepage werkt
   2. Inloggen /typo3 lukt
   3. Formulier op website kan verzonden worden
9. Als de omgeving nog werkt, voer je de acties uit het draaiboek uit.
10. Controleer of de nieuwe features daadwerkelijk werken
11. Werken de nieuwe features niet? Overleg dan direct met de "owner" van de userstory wat het probleem kan zijn en los dit op middels een hotfix of als er stappen miste in het draaiboek door het draaiboek aan te vullen.   
    Werken de userstories zoals verwacht, zet dan de gereleasde userstories op de status "Acceptance" en vermeldt per issue op welke pagina het door de Product Owner te testen is.
12. Assign het ticket aan de Product Owner
13. Controleer of er al een PR van release naar master is. Zo ja, voeg dan de informatie (waar te testen, hoe toe te passen, link naar draaiboek) omtrent de op te leveren userstory toe aan de omschrijving van de PR.   
    Is er nog geen PR, dan maak je die aan en vermeldt je de informatie (waar te testen, hoe toe te passen, link naar draaiboek). Geef hierbij de scrum-master, de betrokken ontwikkelaars van de issues in de PR en degene die de releases verzorgt op als reviewer. De titel van de PR moet zijn: [TICKET-NR] Productie-release [datum]. Hierbij gebruik je als ticketnummer het ticket van het release-management (van deze maand) voor het project en als datum de datum waarom het gereleased moet worden.

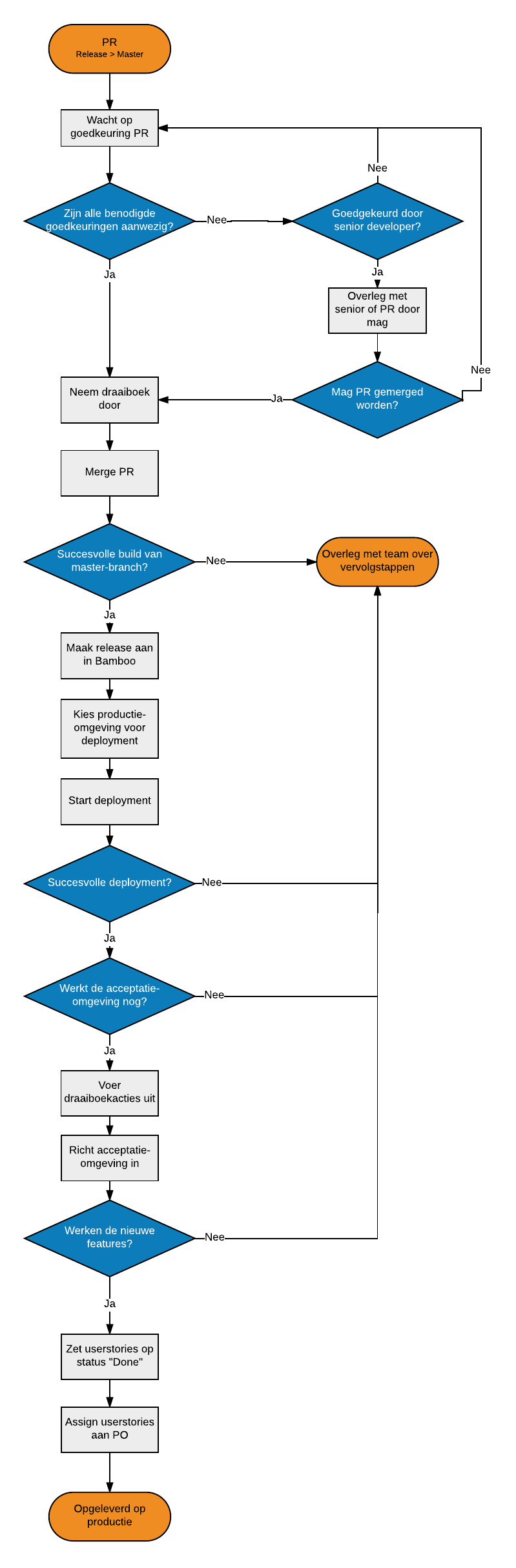
**4.3 Deployment naar productie-omgeving**

**Aandachtspunt 1:** De release naar een productie-omgeving moet altijd met de grootste zorg worden uitgevoerd gezien de release direct invloed kan hebben op de live website van de klant!

**Aandachtspunt 2:** Een release naar productie wordt alleen gedaan door de door MaxServ aangewezen releasers en worden alleen uitgevoerd binnen het beschikbare time-window (maandag t/m donderdag tussen 10.00 en 15.00 uur). Dit wordt gedaan om de kans op (langere) downtime te verminderen.

De stappen verschillen niet heel erg met de stappen voor een deployment naar de acceptatie-omgeving, maar toch nog even alle stappen.

1. In stap 13 van het release-proces naar de acceptatie-omgeving is er een PR gekomen van de release-branch naar de master-branch. Als iedere reviewer de PR heeft goedgekeurd, kan je verder naar de volgende stap. Missen er om wat voor reden dan ook goedkeuringen op de PR maar is hij bijvoorbeeld wel goedgekeurd door een senior Developer, overleg dan even met die senior.
2. Is er een draaiboek aanwezig? Zo ja, neem deze door zodat je weet wat er speelt. Raadpleeg bij vragen de schrijver van de betreffende draaiboek-actie.
3. Merge de PR en wacht tot bamboo de master-branch succesvol heeft kunnen builden.
4. Bij een succesvolle build van Bamboo ga je vanuit de betreffende build een release maken.
5. Kies vervolgens de omgeving waar je naar wilt uitrollen, in dit geval de productie-omgeving.
6. Deploy nu de zojuist gemaakte release
7. Zodra Bamboo klaar is met de deployment (hoe snel dit is hangt af van beschikbaarheid en het type project) zal hij een melding hiervan geven in het HipChat-kanaal. Ook zal het scherm waarin je de build hebt gestart een melding geven dat de deployment geslaagd is.
8. Controleer of de release daadwerkelijk geslaagd is en de productie-omgeving nog bereikbaar is. Test in ieder geval de volgende onderdelen:
   1. Homepage werkt
   2. Inloggen /typo3 lukt
   3. Formulier op website kan verzonden worden
9. Als de omgeving nog werkt, voer je de acties uit het draaiboek uit.
10. Controleer of de nieuwe features daadwerkelijk werken
11. Werken de nieuwe features niet? Overleg dan direct met de "owner" van de userstory die niet werkt en de scrum-master (projectmanager) voor een eventuele terugrol van de voorgaande versie of een andere oplossing.  
    Werken de userstories zoals verwacht, zet dan de gereleasde userstories op de status "Done" en vermeldt per issue waar het geimplementeerd is.
12. Assign het ticket aan de Product Owner om te laten weten dat het issue gereleased is.

**Deployment naar productie-omgeving**

|  |
| --- |
| **Product Backlog** |

*Zie ieder word-document met bijhorende ticketnaam en -inhoud, voor volledige weergave en verdere details*

[MXSO-3] [Aanpassen van Servicecontracten/Supportklanten lijst](https://jira.maxserv.com/browse/MXSO-3)

[MXSO-6] [User Story - Klant detail pagina tab 01 - uren report](https://jira.maxserv.com/browse/MXSO-6)

[MXSO-7] [User Story - Mogelijkheid om een report als PDF te exporteren voor specifiek support project](https://jira.maxserv.com/browse/MXSO-7)

[MXSO-8] [User Story - Mogelijkheid om een report als PDF te exporteren voor alle support projecten](https://jira.maxserv.com/browse/MXSO-8)

[MXSO-14] [User Story - Klant detail pagina algemene informatie](https://jira.maxserv.com/browse/MXSO-14)

[MXSO-15] [Aanpassen van Support klant/project details overzicht](https://jira.maxserv.com/browse/MXSO-15)

[MXSO-17] [Export van de reports](https://jira.maxserv.com/browse/MXSO-17)

[MXSO-19] [Onderzoek mogelijkheden data exporteren naar PDF](https://jira.maxserv.com/browse/MXSO-19)

[MXSO-20] [User Story - Klant detail pagina tab 02 - Open issues](https://jira.maxserv.com/browse/MXSO-20)

[MXSO-21] [User Story - Klant detail pagina tab 03 - Opmerkingen](https://jira.maxserv.com/browse/MXSO-21)

[MXSO-22] [User Story - Klant detail pagina tab 04 - Contactpersonen](https://jira.maxserv.com/browse/MXSO-22)

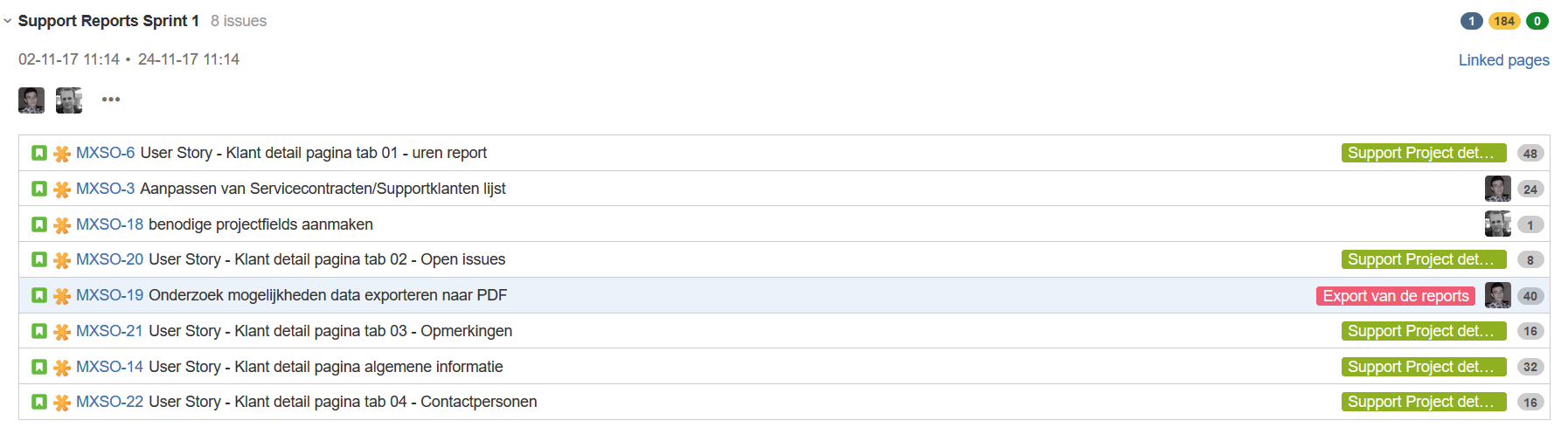
[MXSO-43] [User Story - Testen van 'development' pagina](https://jira.maxserv.com/browse/MXSO-43)

[MXSO-46] [User story - Optimaliseren van 'development' pagina](https://jira.maxserv.com/browse/MXSO-46)

|  |
| --- |
| **Sprint planning en sprint backlog** |

Hoeveelheid story points per user-story is bepaald aan de hand van de geschatte werktijd dat nodig is om dat doel te bereiken.

*Voorbeeld: 1 uur = 2 story points, 8 uur/1 dag = 16 story points.*

**

Voor de eerste sprint heb ik gekozen om zoveel mogelijk tickets/issues (user-stories) op te pakken. Daarvoor wilde ik alle werkuren gebruiken dat toegestaan is binnen een Scrum sprint.

De eerste sprint startte op 2 november 2017 en eindigde op 24 november 2017 omdat ik van 27 november 2017 t/m 1 december 2017 vakantie had.



Voor de tweede sprint heb ik ervoor gekozen om de tickets van de vorige sprint mee te nemen, samen met de overige tickets.

De tweede sprint startte op 7 december 2017 en eindigde op 22 december 2017.

|  |
| --- |
| **Sprint review** |

Sprint 1 (02/11/2017 – 24/11/2017): niet van toepassing vanwege het feit dat de user-stories niet afgerond konden worden. Dit door ontbrekende data van rapportages en onverwachte belemmeringen. Zie voor verdere informatie de bijhorende retrospectief.

Sprint 2 (7/12/2017 – 22-12-2017): niet van toepassing omdat de sprint niet volledig was afgerond. Dit kwam door backend-gerelateerde problemen op de lokale reports website, die tussentijds plaats vonden. Ook was er nog steeds onmogelijk om alle recente gegevens uit de nieuwe versie van Jira plugin ‘profields’ op te halen, omdat de bijhorende API ontbrak. Zie voor verdere informatie de bijhorende retrospectief.

|  |
| --- |
| **Sprint retrospectief** |

Sprint 1 (02/11/2017 – 24/11/2017): Deadline niet gehaald doordat de user-stories niet voltooid konden worden. In de loop van de sprint kwamen onverwachte belemmeringen zoals gebrek aan kennis van TYPO3 extensies, wat nodig is om de inhoud en de werking van de website te kunnen begrijpen. Daarnaast ontbraken bepaalde voor de rapportages, in de website. Tegelijkertijd kwam het ook door problemen in het onderhoud van de website waarin opgeteld 1 week aan verloren ging: de hostomgeving en de inhoud van de website. Zo ontstonden meerdere keren een error in de configuratie van zowel de front-end als de back-end van de website. Door deze gevolgen had ik minder tijd dan gepland om de user-stories verder af te werken.

Sprint 2 (07/12/2017 – 22/12/2017): Ook hier was de deadline niet te halen. Alhoewel de meeste taken voorbij de ‘in-progress’ fase waren, kwamen tussendoor problemen in de backend van de website, met name de database.

Wat ***NIET*** was gelukt: Wanneer er een (PHP)‘actie’ (dat altijd verwijst naar een webpagina) als de parameter wordt meegegeven in de url, wordt die opgeslagen in de database. Op het moment dat de website een ‘actie’ ontbreekt in de source code, wordt de website net zolang onderbroken door de error totdat die weer dezelfde ‘actie’ vindt.

Daarnaast kon de website alle support-gerelateerde open issues niet ophalen. De Jira (MaxServ) server gaf geen toegang tot die data omdat die de rechten van de gebruiker niet kon uitlezen. Maar dit werd tegen het einde van de sprint wel opgelost in de source code.

De pdf-conversie van de supportpagina’s ging ook niet zonder fouten: vector afbeeldingen (SVG bestanden) waarvan de grootte niet bepaald kon worden en daardoor niet werd ingevoerd, html tags die niet herkend worden.

Wat ***WEL*** was gelukt: de correcte weergave van de staafdiagram aan te tonen, door alle benodigde data op te halen met de juiste SQL query uit te voeren. Zoals eerder benoemd het weergeven van de open issues. De mogelijkheid om alle opmerkingen en contactpersonen, onder de tabbladen op de detailpagina, te beheren vanuit de TYPO3 backend.