# Intro testen

# 1 Wat is Testen??

**Opdracht 1:**

**Wat is de geschiedenis van TMAP?**

TMAP, Test Management Approach, is een testmethode voor softwareproducten die ontwikkeld is in Nederland door Sogeti en voor het eerst toegepast in 1995. Hiervoor was het testen meer gebaseerd op het vinden van fouten en deze oplossen, om zo het ‘perfecte’ softwareproduct te leveren. Bij het ontwikkelen van TMAP heeft men de stap van het oplossen van de fouten geschrapt en zich puur gefocust op het vinden en aanduiden van fouten in het product. In 2006 is TMAP NEXT geïntroduceerd, dit is een meer traditionele manier van testen gebaseerd op de watervalmethode. In 2014 is TMAP HD geïntroduceerd, deze is bedoeld voor Agile-softwareontwikkeling. In 2020 is uiteindelijk ook TMAP Quality for DevOps teams ontstaan, de manier van hoogwaardige IT-levering voor de DevOps-cultuur.

**Wat is de kracht van TMAP?**

De kracht van TMAP is het structureren van het testproces (white- en black-box) van softwareproducten en zorgt ervoor dat de tijdlijnen verkort worden en er dus kosten bespaard worden. Hierdoor worden fouten in een vroeg stadium gevonden. Verder is het inzetbaar in elke situatie ongeacht de opdrachtgever of ontwikkelingssysteem. Verder is TMAP erop gericht om met zo min mogelijk testgevallen de vereiste dekking te bereiken.

**Wat biedt TMAP?**

Tmap biedt het meest optimale testproces. Het geeft ook inzicht en advies over eventuele risico’s over de kwaliteit van het geteste product.

**Waarbij helpt TMAP?**

Omdat er een standaard werkwijze is, zijn testspecificaties overdraagbaar en onderhoudbaar.

**Opdracht 2:**

* **Wat is de definitie van testen?**
* **Wat is het verschil tussen Testen en Toetsen?**

Testen is vaststellen in hoeverre het softwareproduct aan de eisen voldoet. Testen doet men dus aan het einde van het opleveren. Toetsen doet men tijdens het ontwikkelingsproces.

**Opdracht 3:**

**Wat zijn eventuele valkuilen die je in de praktijk kan tegenkomen?**

Ongestructureerde testen zorgt voor chaos. End 2 end testen, je moet automatiseren waar nodig niet waar het kan. Het moet de kwaliteit verbeteren van het proces en niet automatiseren om te automatiseren. Binnen scrum ben je geneigd om in korte sprints te werken en daardoor veel te automatiseren. Hierdoor ben je van tevoren genoodzaakt om goed na te denken over wat ga ik automatiseren en wat niet?

# 2 Waarom is testen belangrijk?

**Opdracht 1:**

**Waarom is testen belangrijk in het proces van software development?**

Testen is belangrijk omdat er dan in vroeg stadium fouten kunnen worden verbeterd en er zo eerder een goed softwareproduct wordt geleverd. Als er niet getest wordt, komen fouten na levering aan het licht en dan is het meestal veel duurder om die fouten op te lossen. Doel van testen is voorkomen is beter dan genezen!

**Opdracht 2:**

**Zoek op het internet naar voorbeelden van waar software mogelijk niet goed genoeg getest was en geef ook aan wat de gevolgen daarvan waren.**

Een voorbeeld wat ik zelf ken in de praktijk is de levering van de game Cyberpunk 2077. Dit is een game waar jaren aan is gewerkt en waar de gaming community reikhalzend naar uitkeek. Probleem is geweest dat de ontwikkeling van de game samenviel met de lancering van next-gen gamingplatformen, denk hierbij aan de PS5 en xbox series x. Het spel draait (op een paar bugs na) fantastisch op de next-gen consoles en op een goede game pc. Echter is de game onspeelbaar op de current-gen (PS4, Xbox one X), waardoor het bedrijf genoodzaakt was om na de meest succesvolle verkoop bij lancering, de verkoop van de game op current-gen moest stoppen en zelfs begonnen is met het terugbetalen omdat het geleverde prodcuct (game) niet aan de eisen voldoet. Hier is het dus misgegaan bij het testen van de game op de current-gen, na zo’n test had de game nooit mogen verschijnen op de huidige generatie spelcomputers.

Een ander voorbeeld is de NEST thermostaat, die na een software update uitviel en ervoor zorgde dat het onmogelijk was om de thermostaat aan te krijgen en dus je huis te verwarmen.

Als laatst een voorbeeld wat veel mensen kennen, is het dieselschandaal van Volkswagen. Er was bij het vaststellen van de uitstoot van roetdeeltjes opzettelijk geknoeid met de software van de auto waardoor deze een testrollerbank herkende en zo dus minder uitstoot veroorzaakte en dus de test saboteerde. Bij een goede gedegen test, was deze opzettelijke ‘fout’ eruit gehaald en had de auto nooit geproduceerd mogen worden. Wat uiteindelijk heeft geleid tot miljardenclaims, kelderende beursaandelen, particulieren die massaal hun auto’s verkochten etc.