Ontwikkelen en valideren van oplossingen en MVP's

Inleiding

Zodra u een eerste bedrijfsconcept krijgt door alle vorige screenings en u er zeker van bent (voor zover het mogelijk is) dat het de moeite waard is om uw tijd en inspanningen verder te investeren, bereikt u een punt in bedrijfsontwerp waar u een **minimum viable product** moet maken.

Het zogenaamde Minimum Viable Product (MVP) is een strategie die wordt gebruikt voor het snel en kwantitatief testen van een waardepropositie, product, dienst of functie. Het werd aanvankelijk gebruikt voor webapplicaties en gepopulariseerd door Eric Ries. Een MVP is niet altijd een kleinere of goedkopere versie van uw eindproduct. Vaak kunt u met een goedkopere proxy de aannames die ten grondslag liggen aan uw waardepropositie sneller en kosteneffectiever testen. **Vraag uzelf af wat u echt wilt leren en wat de goedkoopste hack is om die aanname te testen.** Hieronder vindt u nuttige inzichten over MVP-ontwikkeling en -testen, met dank aan Dan Olsen (*"The Lean Product Playbook: How to Innovate with Minimum Viable Products and Rapid Customer Feedback").*" Gebruik de hoofdstukken om uw MVP-ontwikkeling te ondersteunen.

Geef uw MVP-functieset (Minimum Viable Product) op

Als u inmiddels **een duidelijk begrip heeft van uw waardepropositie**, is de volgende stap in het Lean Product Proces het **bepalen van de functieset voor uw kandidaat voor een minimaal levensvatbaar product (MVP).** U gaat niet beginnen met het ontwerpen van een nieuw product dat uw volledige waardepropositie waarmaakt, omdat dat te lang zou duren en te riskant zou zijn. Voor uw MVP wilt u de minimale functionaliteit identificeren die nodig is om te valideren dat u op de goede weg bent. We kunnen dit een **MVP-kandidaat** noemen in plaats van een MVP omdat het gebaseerd is op uw hypothesen. U heeft nog niet gevalideerd met klanten dat *ze het erover eens zijn* dat het in feite een levensvatbaar product is.

Voor elk voordeel in uw productwaardepropositie wilt u als team brainstormen om met zoveel mogelijk functie-ideeën te komen voor hoe uw product dat voordeel zou kunnen opleveren. U heeft al dit geweldige denken in de probleemruimte gedaan en gaan nu over naar de oplossingsruimte. Terwijl uw team brainstormt, probeert u voort te bouwen op elkaars suggesties en elkaar te pushen om met nog meer creatieve en wilde ideeën te komen. Wanneer u klaar bent met brainstormen, wilt u alle ideeën vastleggen die uw team heeft gegenereerd en ze vervolgens organiseren op basis van

het voordeel dat ze opleveren. Vervolgens wilt u voor elk voordeel de lijst met functie-ideeën bekijken en prioriteren. U kunt elk idee scoren op verwachte klantwaarde om een first-pass prioriteit te bepalen. Het doel is om de top drie tot vijf functies voor elk voordeel te identificeren. Het is op dit moment niet de moeite waard om verder te kijken dan die topfuncties, omdat dingen veel zullen veranderen nadat u uw prototype aan klanten heeft laten zien.

Op dit punt moeten brainstormregels van toepassing zijn. U moet divergent denken beoefenen, *wat betekent dat u zoveel mogelijk ideeën probeert te genereren zonder enig oordeel of evaluatie. Er zal later voldoende tijd zijn voor* convergent denken, *waarbij u de ideeën evalueert en beslist welke u het meest veelbelovend vindt.*

User Stories: functies met voordelen

User stories (gebruikt in Agile development) zijn een geweldige manier om uw functie-ideeën te schrijven om ervoor te zorgen dat het bijbehorende klantvoordeel duidelijk blijft. Een user story (gebruikersverhaal) is een korte beschrijving van het voordeel dat de specifieke functionaliteit moet bieden, inclusief voor wie het voordeel is (de doelklant) en waarom de klant het voordeel wil. Goed geschreven user stories volgen meestal het model:

Als [type gebruiker],

Ik wil [iets doen],

zodat ik [gewenst voordeel] kan doen.

Hier volgt een voorbeeld van een user story die dit model volgt:

Als professioneel fotograaf,

Ik wil eenvoudig foto's van mijn camera uploaden naar mijn website,

zodat ik mijn klanten snel hun foto's kan laten zien.

Deze sjabloon is een goed begin, maar het schrijven van goede gebruikersverhalen is een verworven vaardigheid. Agile thought leader Bill Wake creëerde een reeks richtlijnen voor het schrijven van goede user stories; om ze gemakkelijker te onthouden, gebruikt hij het acroniem **INVEST**:

- **Independent (Onafhankelijk):** Een goed verhaal moet onafhankelijk zijn van andere verhalen. Verhalen mogen elkaar niet overlappen in concept en moeten in elke volgorde kunnen worden geïmplementeerd.

- **Negotiable (Bespreekbaar):** Een goed verhaal is geen expliciet contract voor functies. De details over hoe het voordeel van een verhaal zal worden geleverd, moeten openstaan voor discussie.

- **Valuable (Waardevol):** Een goed verhaal moet waardevol zijn voor de klant.

- **Estimable (Schatbaar):** Een goed verhaal is er een waarvan de reikwijdte redelijk kan worden ingeschat.

- **Small (Klein):** Goede verhalen hebben de neiging om klein van omvang te zijn. Grotere verhalen zullen meer onzekerheid hebben, dus u moet ze opsplitsen.

- **Testable (Toetsbaar):** Een goed verhaal geeft voldoende informatie om duidelijk te maken hoe te testen dat het verhaal "klaar" is (acceptatiecriteria genoemd*).*

Functies afbreken

Zodra u gebruikersverhalen op hoog niveau heeft geschreven voor uw topfuncties, is de volgende stap om manieren te vinden om elk van hen op te splitsen in kleinere stukjes functionaliteit - een proces dat "**chunking"** wordt genoemd. **Het doel is om manieren te vinden om het bereik te verkleinen en alleen de meest waardevolle stukken van elke functie te bouwen.** Wanneer iemand met een functie-idee komt, zijn er vaak creatieve manieren om minder belangrijke stukken af te snijden. We zullen bewust de term "feature chunk" ("functiestuk") gebruiken in plaats van "feature" ("functie") om u eraan te herinneren dat u niet moet werken met items die groot zijn in omvang, maar eerder dergelijke items moet opsplitsen in kleinere, atomaire componenten.

Laten we het idee illustreren om een user story op hoog niveau af te breken. Stel dat u werkt aan een applicatie voor het delen van foto's en begint met de user story: "Als gebruiker wil ik gemakkelijk foto's kunnen delen met mijn vrienden, zodat ze ervan kunnen genieten". Een manier om dit verhaal op te splitsen is door de verschillende kanalen die een klant kan gebruiken om foto's te delen: Facebook, Twitter, Pinterest, e-mail, sms, enzovoort. Elk van deze zou een afzonderlijke feature chunk of een user story van kleine omvang zijn. Mogelijk hoeft u niet al deze deelkanalen uit te bouwen voor uw MVP. Zelfs als u heeft besloten dat u dat heeft gedaan, helpt het om het verhaal op te splitsen om specifieker te zijn in uw productdefinitie, om nauwkeurigere verkenning van ontwikkeling mogelijk te maken en om u in staat te stellen expliciet prioriteit te geven aan de volgorde waarin u de stukken bouwt. U kunt het bereik ook beperken door de gebruiker in staat te stellen alleen de foto te delen en niets anders voor uw MVP. Mogelijk heeft u ideeën voor extra functionaliteit, zoals het toevoegen van een optioneel bericht aan elke foto of de mogelijkheid om gebruikers in foto's te taggen. Elk van deze zou een aparte feature chunk zijn.

Kleinere batchgroottes zijn beter

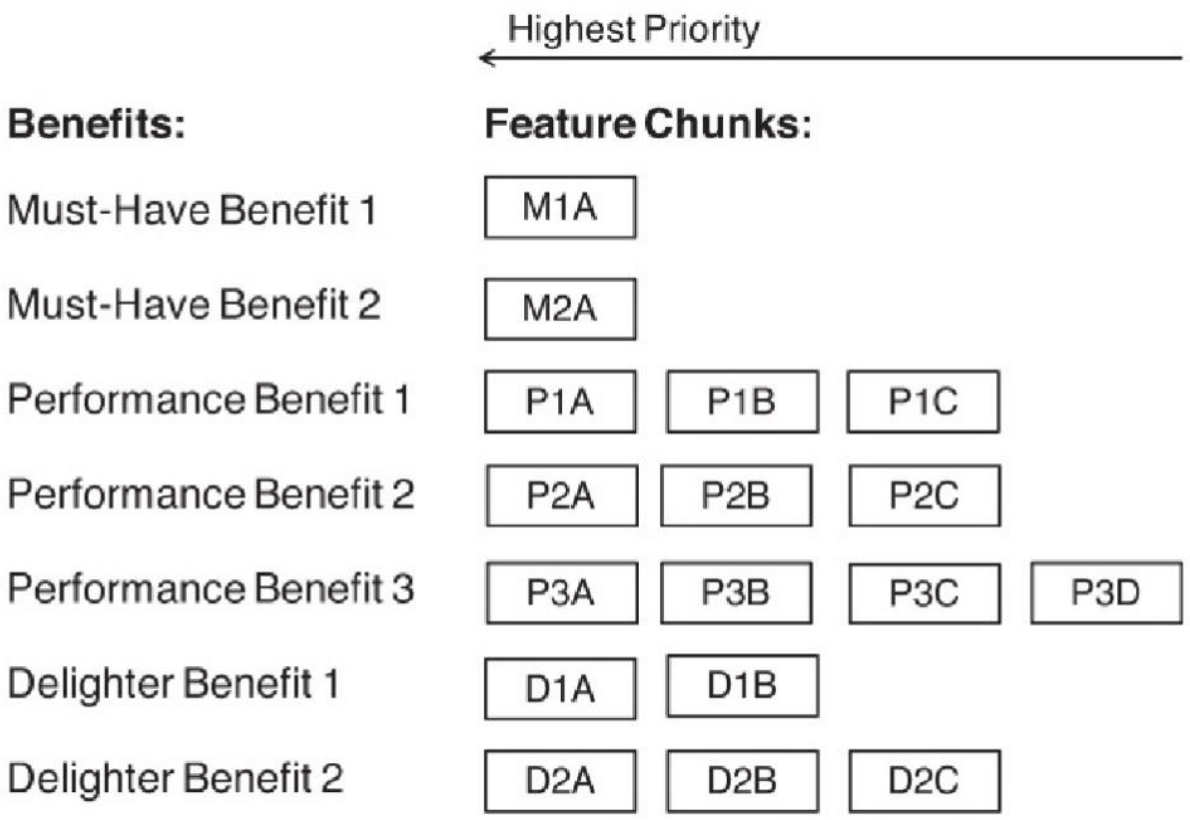
De tactiek om functies op te splitsen is consistent met de Lean manufacturing optimale werkwijze van het werken in kleine batchgroottes. Wanneer een product in een fabriekslijn wordt vervaardigd, is de batchgrootte het aantal producten waaraan tegelijkertijd (in elke stap van het productieproces) wordt gewerkt. De parallel voor softwareontwikkeling is de grootte van de functies of user stories die moeten worden gecodeerd. **Werken in kleinere batchgroottes verhoogt de snelheid omdat ze snellere feedback mogelijk maken, wat risico's en verspilling vermindert.** Als een ontwikkelaar een maand achter haar computer een functie ontwikkelt en deze vervolgens aan de productmanager en ontwerper laat zien, is de kans groter dat er een verbroken verbinding is en dat hun feedback aanzienlijke wijzigingen vereist.

Als de ontwikkelaar in plaats daarvan elke dag of twee haar werk aan de productmanager en ontwerper laat zien, voorkomt dat dat er een grote ontkoppeling optreedt. De omvang van feedback en koerscorrecties zal veel kleiner en beter beheersbaar zijn, wat resulteert in minder verspild werk en een hogere productiviteit.

Dit advies geldt ook voor productmanagers en ontwerpers die hun werkproduct (bijvoorbeeld user stories en wireframes) aan hun teamgenoten laten zien. Het voordeel van het werken in kleine batchgroottes geldt ook voor feedback van klanten. Hoe langer u aan een product werkt zonder feedback van klanten te krijgen, hoe meer u een grote ontkoppeling riskeert die vervolgens aanzienlijk moet worden herwerkt.

Beslissen over uw MVP-kandidaat

Zodra u klaar bent met het in stukken snijden, verkennen en prioriteiten stellen, kunt u een eenvoudige indeling maken die de voordelen van uw waardepropositie opsomt, en die voor elk voordeel, de beste functie-ideeën opgesplitst in stukken meldt.



FIGUUR 1. LIJST MET GEPRIORITEERDE FEATURE CHUNKS VOOR ELK VOORDEEL

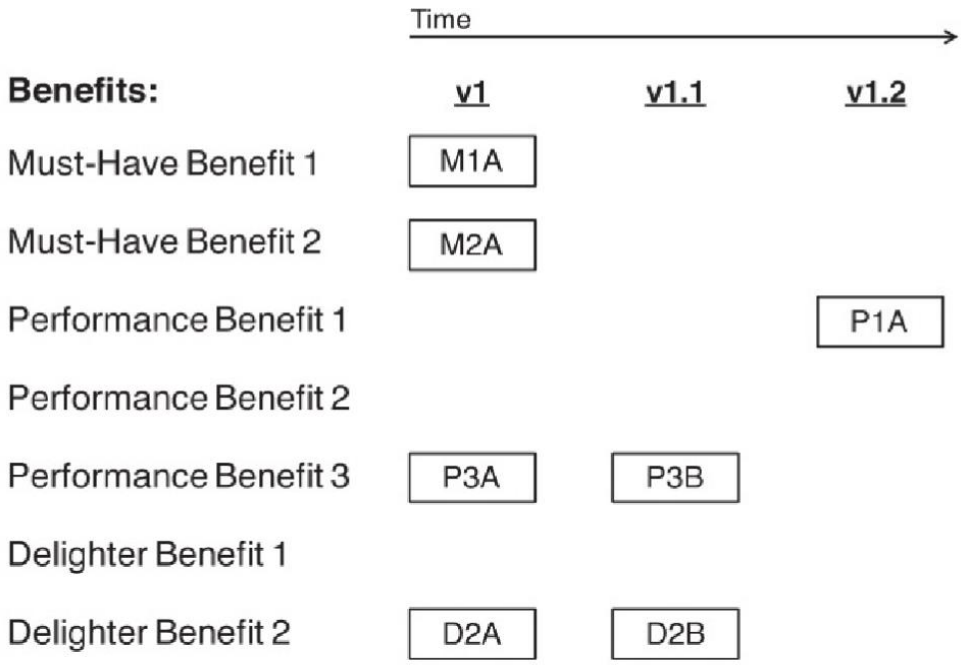
In figuur 1 worden de hoogste feature chunks voor elk voordeel in prioriteitsvolgorde weergegeven, met een hogere prioriteit aan de linkerkant. In plaats van specifieke voordelen of feature chunks te noemen, hebben we ze opzettelijk generieke namen gegeven, zodat u zich gemakkelijker kunt voorstellen dat u ze vervangt door wat relevant zou zijn voor uw product. "M1A" betekent feature chunk A voor Must-Have Benefit 1. "P2B" betekent feature chunk B voor Performance Benefit 2 (prestatievoordeel 2), en "D2C" betekent feature chunk C voor Delighter Benefit 2 (genoegenvoordeel 2). Bij het invullen van een vergelijkbaar raster voor uw product, zou u in plaats daarvan de specifieke labels gebruiken voor uw voordelen en feature chunks.

Zodra u uw lijst met feature chunks op voordeel heeft georganiseerd en ze heeft geprioriteerd, is het tijd om een aantal moeilijke beslissingen te nemen. **U moet beslissen over de minimale set functionaliteiten die zal uw doelgroep aanspreken**. U gaat naar de meest linkse kolom met feature chunks kijken en bepalen welke volgens u in uw MVP-kandidaat moeten zitten. Terwijl u dit doet, moet u verwijzen naar uw productwaardepropositie. Om te beginnen moet uw MVP-kandidaat alle must-haves hebben die u heeft geïdentificeerd.

Daarna moet u zich concentreren op het belangrijkste prestatievoordeel dat u van plan bent te gebruiken om de concurrentie te verslaan. U moet voor dit voordeel de set

feature chunks selecteren waarvan u denkt dat ze voldoende bieden voor klanten om het verschil in uw product te zien.

Delighters (genoegen) maken ook deel uit van uw differentiatie. U moet uw hoogste delighters opnemen in uw MVP-kandidaat. Dat is misschien niet nodig als u een heel groot voordeel heeft op een prestatievoordeel. Het doel is om ervoor te zorgen dat uw MVP-kandidaat *iets* bevat dat klanten superieur vinden aan de producten van anderen en, idealiter, uniek. De feature chunks waarvan u denkt dat ze in uw MVP-kandidaat moeten zitten, blijven in de meest linkse kolom, die u als **'v1'** kunt bestempelen, zoals u ziet in figuur 2, terwijl de andere naar rechts worden geduwd. U kunt een voorlopig product-stappenplan maken door dit proces voort te zetten en kolommen te maken voor elke toekomstige versie met elke kolom met de functiebrokken die u wilt toevoegen.



FIGUUR 2. BESLISSEN WELKE FEATURE CHUNKS IN UW MVP-KANDIDAAT ZITTEN

Omdat u van plan bent om de beste te zijn in prestatievoordeel 3, neemt u de hoogste prestatiefunctiestuk, P3A, op in uw MVP-kandidaat. U bent ook van plan om te differentiëren met differentiator 2, dus u neemt feature chunk D2A op in uw MVP-kandidaat. U MVP-kandidaat heeft ook de twee must-haves.

Als u uitkijkt naar uw volgende versie, v1.1, bent u van plan om verder te investeren in prestatievoordeel 3 en delighter 2 met respectievelijk feature chunks P3B en D2B. In de versie daarna, v1.2, bent u van plan om prestatievoordeel 1 aan te pakken met de hoogste prestatiefunctiestuk P1A.

Het wordt niet aanbevolen om in het begin meer dan één of twee secundaire versies vooruit te plannen, omdat veel dingen geneigd zijn te veranderen wanneer u uw MVP-kandidaat voor de eerste keer aan klanten laat zien. U zult leren dat sommige van uw hypothesen niet helemaal juist waren en nieuwe hypothesen bedenken. U kunt uiteindelijk van gedachten veranderen over welk voordeel het belangrijkst is of met ideeën komen voor nieuwe functies om dezelfde voordelen aan te pakken. Dus als u voorzichtige plannen heeft gemaakt die verder gaan dan uw MVP, moet u bereid zijn om ze terzijde te schuiven en nieuwe plannen te bedenken op basis van wat u van klanten leert. Het kan echter zo zijn dat u twee of drie feature chunks nodig heeft voor een bepaald voordeel, afhankelijk van uw situatie en hoe klein uw feature chunks zijn. Het idee is nog steeds hetzelfde: om te kiezen welke feature chunks in die meest linkse kolom moeten staan, die overeenkomt met uw MVP-kandidaat.

Laten we een stap terug doen en nadenken. Op dit punt in het Lean Product Process heeft u behoorlijk wat werk verzet. U heeft:

- hypothesen over uw doelklanten gevormd
- hypothesen over hun achtergestelde behoeften gevormd
- de waardepropositie verwoorden die u van plan bent na te streven, zodat uw product beter en anders is
- de beste functie-ideeën waarvan u denkt dat ze aan die behoeften zullen voldoen geidentificeerd en ze op in kleinere stukken gesplitst
- Prioriteit gegeven aan die feature chunks op basis van ROI
- een set van die feature chunks voor uw MVP-kandidaat geselecteerd, waarvan u veronderstelt dat klanten deze waardevol zullen vinden

U heeft heel rigoureus nagedacht om zover te komen, maar uw MVP is nog steeds slechts een kandidaat, een bundel van onderling verbonden hypothesen. U moet feedback van klanten krijgen over uw MVP-kandidaat om die hypothesen te testen. Maar voordat u kunt testen, moet u een oplossingsruimterepresentatie van uw MVP-kandidaat maken die u aan klanten kunt laten zien, wat de volgende stap is in het Lean Product Process.

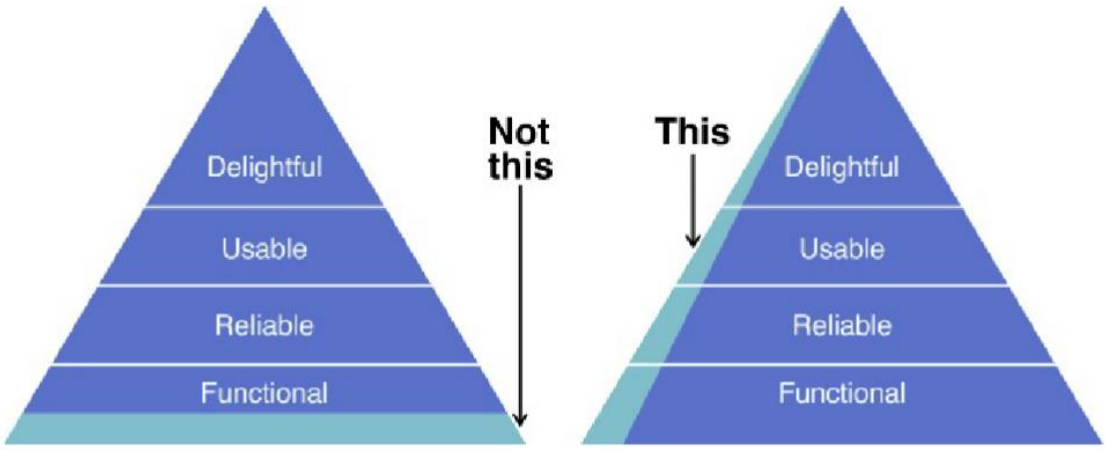
Uw MVP maken - Wat is (en is niet) een MVP?

Er is een pittig debat geweest over wat kwalificeert als een MVP. Sommige mensen beweren heftig dat een landingspagina een geldige MVP is. Anderen zeggen van niet en staan erop dat een MVP een echt, werkend product moet zijn of op zijn minst een interactief prototype. De manier om deze tweedeling op te lossen is om te beseffen dat dit allemaal methoden zijn om de hypothesen achter uw MVP te testen. Door de term "MVP-tests" te gebruiken in plaats van MVP, verdwijnt het debat. Dit maakt nauwkeurigere terminologie mogelijk door het gebruik van MVP te reserveren voor daadwerkelijke producten.

Hoewel het waar is dat een MVP opzettelijk beperkt is in omvang ten opzichte van uw hele waardepropositie, moet wat u aan klanten vrijgeeft boven een bepaalde lat liggen om waarde voor hen te creëren.

Veel mensen interpreteren de term MVP verkeerd door te veel nadruk te leggen op het woord *minimum.* Ze gebruiken dit als een excuus om een gedeeltelijke MVP te bouwen die te weinig functionaliteit heeft om door een klant als levensvatbaar te worden beschouwd. Anderen gebruiken 'minimum' om een slordige gebruikerservaring of een buggy-product te rationaliseren.

Het diagram in figuur 3 illustreert het verschil tussen deze onjuiste manier van interpreteren van MVP en de juiste interpretatie.



FIGUUR 3. EEN MVP BOUWEN

Vergelijkbaar met de hiërarchie van de behoeften van de webgebruiker, scheidt deze figuur de verschillende aspecten van een product. In dit geval **wordt een piramide van vier hiërarchische lagen gebruikt om de kenmerken van een product te beschrijven: functioneel, betrouwbaar, bruikbaar en verrukkelijk**. De piramide aan de linkerkant illustreert de misvatting dat een MVP slechts een product is met beperkte functionaliteit, en dat betrouwbaarheid, bruikbaarheid en genot kunnen worden genegeerd. In plaats daarvan laat de piramide aan de rechterkant zien dat hoewel een MVP beperkte functionaliteit heeft, deze "compleet" moet zijn door die drie hogere attributen aan te pakken.

Tot slot - MVP-tests

Eindelijk, nadat u al het werk heeft gedaan, is het nu tijd om uw MVP te testen. Voor een sterke afsluiting van onze sessie gewijd aan het ontwikkelen en testen van MVP's, raadpleegt u het appendix test framework (met dank aan Strategyzer) en probeert u de meest geschikte testtechnieken te vinden die u kunt toepassen tijdens de zoek- en ontwerpfase van uw bedrijfsidee.

Bespreek ze in uw team en kies waar/hoe u dienovereenkomstig begint.

Verder lezen & informatie:

- (pdf) Minimum Viable Product of Meerdere Facet Product? De rol van MVP in software-startups (researchgate.net)
- Ontwikkel de oplossing (iitoolkit.com)
- Validatie van de oplossing en constructie van de MVP - Scaleapse
- Wat is de MVP-fase in een startup? | Londen Bedrijfsadvies (hwca.com)

Bronnen & documentatie:

AG, S. (no date) Testing your business model, Resource Library. Available at: https://www.strategyzer.com/assets/resources (Accessed: November 7, 2022).

Improving improvement (no date) Develop the Solution. Available at: https://www.iitoolkit.com/process/solution.html#:~:text=Developing%20the%20solution%20has%20direct,evaluate%20potential%20ideas%20for%20improvement (Accessed: November 7, 2022).

Pansini, S. (2022) Validation of the solution and construction of the MVP, Scaleapse. Available at: https://www.scaleapse.com/validation-of-the-solution-and-construction-of-the-mvp/ (Accessed: November 7, 2022).

Umbraco.Web.PublishedModels.TeamMember (2018) What is the MVP stage in a startup?: London Business Advice, Haines Watts Group. Haines Watts Group. Available at: https://www.hwca.com/accountants-london/opinion/mvp-stage-tech%20startup/#:~:text=MVP%20is%20an%20abbreviation%20for,basic%20principle%20is%20relatively%20simple (Accessed: November 7, 2022).

Burns, P. (2018). New Venture Creation. Macmillan Education UK. https://doi.org/10.1057/978-1-352-00051-1

Lenarduzzi, V. (2017) MVP explained: A systematic mapping study on the definitions of minimal viable product, 2016 42th Euromicro Conference on Software Engineering and Advanced Applications (SEAA). Available at: https://www.academia.edu/33157207/MVP_Explained_A_Systematic_Mapping_Study_on_the_Definitions_of_Minimal_Viable_Product (Accessed: November 7, 2022)

Nguyen Duc, Anh & Abrahamsson, Pekka. (2016). Minimum Viable Product or Multiple Facet Product? The Role of MVP in Software Startups. 118-130. 10.1007/978-3-319-33515-5_10.