

The logo is a large, stylized, light green emblem on a dark teal background. It features a central diamond shape with rounded corners, surrounded by concentric, flowing lines that resemble a stylized leaf or a modern architectural element. The lines are thick and have a slight gradient, giving it a three-dimensional feel.

GlobalOrange

A series of thin, light green lines forming a complex, interconnected pattern that resembles a circuit board or a network diagram. The lines are thin and have a slight gradient, giving it a three-dimensional feel.

Whitepaper
Agentic MVP development

Inhoud

03

Waarom agentic MVP development?

04

Wat moet onder de motorkap kloppen?

05

Security vanaf de eerste versie

07

Van idee naar betrouwbaar product

09

Groeien zonder vast te lopen

10

Checklist & de volgende stap



Snel naar de markt met een basis die meegroeit.

Je ziet een kans in de markt en wilt weten of klanten erop zitten te wachten. AI-agents kunnen helpen om sneller een werkende eerste versie te bouwen. Dat geeft ruimte om eerder te testen, feedback te verwerken en betere productkeuzes te maken.

Maar zodra echte gebruikers instappen, telt meer dan een overtuigende demo. Kunnen klanten alleen hun eigen gegevens zien? Blijft de applicatie snel als het gebruik groeit? En kan een ontwikkelaar een wijziging doorvoeren zonder elders iets stuk te maken? Die vragen horen bij de eerste versie.

Wat verandert er aan MVP development?

Een Minimum Viable Product is de kleinste bruikbare versie waarmee je een belangrijke productaanname toetst. Je beperkt de functionaliteit om snel te leren: lost dit een probleem op, komen gebruikers terug en willen zij betalen?

Bij agentic MVP development voeren AI-agents afgebakende ontwikkeltaken uit. Ze onderzoeken code, stellen wijzigingen voor en schrijven en draaien tests. Ontwikkelaars geven richting en context, en beoordelen het resultaat.

Je product hoeft zelf geen AI-functie te bevatten. 'Agentic' beschrijft hier hoe je het ontwikkelt.

De kans zit in kortere feedbacklussen. Minder tijd tussen een idee, een werkende versie en een gesprek met een gebruiker.

DORA beschrijft AI als een versterker van bestaande sterke én zwakke punten in een organisatie.[1] Daarom vraagt sneller bouwen om heldere kwaliteitscriteria en mensen die de samenhang bewaken.

Deze whitepaper laat zien hoe je snelheid benut en tegelijk grip houdt op de techniek waarop je verder wilt bouwen.



Een kleine scope, een duidelijke kwaliteitslat

Een MVP hoeft nog niet alle toekomstige functies te hebben. Het moet wel passen bij het gebruik dat je vanaf dag één toestaat. Een besloten test met fictieve data stelt andere eisen dan een dienst met betalingen en klantgegevens. Leg daarom vooraf vast wie het product gebruikt, welke gegevens het verwerkt en wat de gevolgen van een fout zijn. Daaruit volgt de minimale technische kwaliteit.

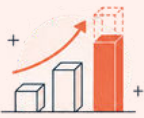
Voorwaarden voor een bruikbaar MVP

**Security**

toegang en gegevens zijn beschermd.

**Onderhoudbaarheid**

wijzigingen blijven begrijpelijk en testbaar.

**Schaalbaarheid**

het product kan de afgesproken groei aan.

**Performance**

belangrijke taken blijven snel genoeg.

Wat kun je uitstellen?

Extra rapportages of een zelden gebruikte koppeling kunnen vaak wachten. Een handmatige stap is soms prima, als die veilig en uitvoerbaar is.

Begin met eenvoudige techniek en duidelijke grenzen tussen onderdelen. Een overzichtelijke applicatie kan geschikter zijn dan veel losse diensten. Voeg complexiteit toe zodra een concrete behoefte dat rechtvaardigt.

Wat regel je vóór de pilot?

Vóór de eerste echte gebruikers wil je toegangscontrole, tests van kritieke handelingen en inzicht in fouten. Zorg ook dat je gegevens kunt herstellen en een mislukte release kunt terugdraaien.

Leg vast welke beperkingen je accepteert, wie ze bewaakt en wanneer je ingrijpt.



Security begint achter het scherm

Een login op het scherm bewijst nog niet dat je applicatie veilig is. Controleer ook aan de serverkant wie welke gegevens mag bekijken of wijzigen. Test bijvoorbeeld of klant A via een aangepast verzoek bij de gegevens van klant B kan. Gebruik concrete beveiligingseisen, bijvoorbeeld uit OWASP ASVS, als basis voor je controles.[2]

Ook je ontwikkelagents hebben grenzen nodig

Een agent die code kan wijzigen, heeft niet automatisch toegang tot klantdata of productie nodig.

Beperk toegang tot de taak en werkomgeving. Houd productiesleutels buiten prompts en code. Laat risicovolle acties apart beoordelen en leg vast wat een agent heeft uitgevoerd.[3]

Bescherm je gebruikers

Maak vooraf duidelijk welke gegevens je verwerkt, waar ze terechtkomen en wie erbij mag. Gebruik veilige standaardcomponenten voor bijvoorbeeld inloggen. Controleer code én gebruikte softwarepakketten op kwetsbaarheden.

Test ook misbruik: verkeerde rechten, ongeldige invoer en te veel verzoeken. Laat risicovolle onderdelen waar nodig apart onderzoeken.

Beheers de ontwikkelomgeving

Geef agents kleine opdrachten met heldere acceptatiecriteria. Werk in een afgeschermd omgeving met geschikte testdata. Behandel instructies uit externe bestanden als onbetrouwbare invoer.

Laat een ontwikkelaar kritieke wijzigingen beoordelen. Een tweede agent kan helpen bij review, maar neemt de verantwoordelijkheid voor de release niet over.



Klein beginnen.
Goed doorgroeien.



Van idee naar betrouwbaar product

1. Maak duidelijk wat je wilt leren

Kies één doelgroep, één belangrijk probleem en een meetbaar resultaat. Spreek af welke functies daarvoor nodig zijn en welke kunnen wachten.

Breng tegelijk de risico's in kaart: gevoelige data, betalingen, verschillende klantomgevingen of afhankelijkheid van externe diensten. Zo voorkom je dat een snelle start later een dure omweg blijkt.



2. Leg de technische basis vast

Kies een eenvoudige architectuur met gangbare, ondersteunde technologie. Beschrijf de belangrijkste onderdelen, gegevensstromen en toegangsrechten.

Leg kwaliteitscriteria vast vóór het bouwen. Maak ook duidelijk wie code beoordeelt, wie mag vrijgeven en hoe je teruggaat naar een werkende versie. Deze werkwijze sluit aan bij het principe van veilige ontwikkeling gedurende de hele levenscyclus.[4]



3. Bouw in controleerbare stappen

Laat agents kleine taken uitvoeren binnen de gekozen architectuur. Geef bedrijfsregels en randgevallen mee. Een ontwikkelaar controleert of de wijziging bij het hele product past.

Toets aan vooraf afgesproken eisen. Tests van dezelfde agent kunnen dezelfde verkeerde aanname bevatten. Voeg onafhankelijke scenario's toe.



4. Lanceer, meet en leer

Start met een beheersbare groep gebruikers. Meet of zij hun taak afronden en of je productaanname klopt. Volg ook fouten, responstijden en kosten.

Test herstel en spreek af wie incidenten oppakt. Ga breder live wanneer de criteria zijn gehaald. Leg resterende beperkingen vast met een eigenaar en een moment om ze aan te pakken.

Korte iteraties werken als leren én kwaliteit onderdeel zijn van iedere stap.



Groeien zonder vast te lopen

Stel je een klantenportaal voor dat tijdens een demo soepel werkt. Enkele maanden later zijn er meer klanten, grotere bestanden en rapportages die tegelijk worden opgevraagd. Een trage zoekopdracht of ontbrekende limiet kan dan het hele product hinderen.

Spreek daarom een realistisch gebruiksprofiel af: hoeveel gelijktijdige gebruikers verwacht je, hoeveel data verwerken zij en welke taken zijn kritisch? Test met representatieve data, pieken en storings van externe diensten. Meet ook de kosten per klant of belangrijke handeling.

Je hoeft nog niet voor miljoenen gebruikers te bouwen. Je wilt weten waar de huidige grenzen liggen en welke volgende stap nodig is als je die nadert.

Schaalbaarheid

Controleer wat er gebeurt als gebruik en datavolume toenemen. Kun je capaciteit toevoegen? Blijft één drukke klant de rest niet hinderen?

Begrens zware taken en gelijktijdige verzoeken. Leg vast bij welke meetwaarden je opschaalt en wat dat kost.

Performance

Maak 'snel genoeg' meetbaar. Bijvoorbeeld: 95% van de zoekopdrachten is binnen twee seconden klaar bij de afgesproken belasting. Dit is een voorbeeld, geen algemene norm.

Meet de hele gebruikers-handeling. Zoek de oorzaak van vertraging voordat je extra capaciteit toevoegt.



Zeven vragen vóór je live gaat

1. Werkt de belangrijkste gebruikersreis?

Laat een echte gebruiker de kerntaak uitvoeren. Vraag naar bewijs van gebruik en feedback, én naar tests van foutscenario's.

2. Zijn klantgegevens gescheiden?

Vraag om toegangs- en misbruiktests. Laat aantonen dat een gebruiker geen gegevens van een andere klant kan lezen of wijzigen.

3. Wat gebeurt er bij groei?

Bekijk testresultaten bij afgesproken belasting, datavolumes en pieken. Vraag ook waar de huidige grens ligt.

4. Kan een ander het onderhouden?

Laat een tweede ontwikkelaar een kleine wijziging doen en de tests draaien. Controleer of opzet, keuzes, afhankelijkheden en installatie zijn vastgelegd.

5. Is herstel aantoonbaar mogelijk?

Vraag om een uitgevoerde hersteltest, een terugvalplan voor releases en afspraken over wie incidenten oppakt.

6. Houd je grip op code en kosten?

Regel toegang tot broncode, accounts en documentatie. Maak afspraken over eigendom, licenties, hosting, AI-gebruik en overdracht.

7. Wie is technisch verantwoordelijk?

Benoem wie de kwaliteit beoordeelt, de release vrijgeeft en openstaande risico's opvolgt.

Vraag om bewijs dat bij het risico past: een testresultaat, een herstelproef of een gedocumenteerde keuze. Leg openstaande punten vast met een eigenaar en een moment van herbeoordeling. Dat geeft meer houvast dan de belofte dat het product 'toekomstbestendig' is. Geen aanpak sluit elk risico uit; je kunt wel aantonen wat je hebt gecontroleerd en wat aandacht nodig heeft.



Jouw MVP: snel leren, met grip op de techniek

Begin met de vraag wat je met jouw MVP wilt bewijzen. Breng daarna de risico's in kaart die je eerste gebruikers echt raken. Zo ontstaat een gerichte scope met een technische basis die past bij je product.

Met een Product Discovery kun je productkeuzes, architectuur en kwaliteitscriteria samen scherp krijgen. Dat geeft richting aan het werk van zowel ontwikkelaars als AI-agents.

Neem naar het eerste gesprek je doelgroep, belangrijkste gebruikersreis en verwachte gebruik mee. Ook een bestaande demo is een goed vertrekpunt.

Wil je een MVP laten ontwikkelen of weten of je huidige versie klaar is voor echte gebruikers? Bespreek met GlobalOrange welke productaannames en technische keuzes eerst aandacht verdienen.

Bronnen & verdieping

[1] DORA — State of AI-assisted Software Development (2025)
dora.dev/research/2025/dora-report/

[2] OWASP — ASVS
owasp.org/projects/asvs

[3] OWASP — AI Agent Security Cheat Sheet
cheatsheetseries.owasp.org

[4] NIST — SSDF, SP 800-218
csrc.nist.gov/pubs/sp/800/218/final

AI-agents inzetten voor ontwikkeling van je product?

Maak ruimte voor snelheid en bouw een basis waarop je verder kunt.

Neem contact op met Guido Sival

+31 20 420 4307

globalorange.nl

