Mission Brief & Plan van Aanpak: Document Generatie App (Fase 1)

Datum: 9 juni 2025 **Versie:** 1.1

1. Mission Brief

1.1 Projectnaam

Document Generatie Web App v1.0

1.2 Missie

Het ontwikkelen van een efficiënte, deelbare en gebruiksvriendelijke web-app binnen de Google Workspace-omgeving (middels Google Apps Script). De app stroomlijnt en automatiseert het proces van data-invoer en het genereren van professionele PDF-documenten. Het einddoel is om de huidige, versnipperde workflow te vervangen door één centrale, opzichzelfstaande en betrouwbare oplossing.

1.3 Kernprobleem

Het huidige proces voor het aanmaken van offertes, werkbonnen en facturen leunt op meerdere, losgekoppelde Google Sheets en complexe scripts. Dit leidt tot:

- **Foutgevoeligheid:** Handmatige data-invoer en -overdracht verhoogt de kans op fouten.
- **Tijdverlies:** Repetitieve handelingen en het zoeken naar data kosten onnodig veel tijd.
- **Onderhoudsuitdagingen:** De complexiteit van de bestaande scripts maakt aanpassingen en uitbreidingen lastig.
- **Beperkte Deelbaarheid:** De huidige opzet is niet eenvoudig te delen of te schalen met andere gebruikers.

1.4 Oplossing (Scope Fase 1)

Voor fase 1 bouwen we een **opzichzelfstaande (standalone) Web App**. Deze app functioneert volledig via zijn eigen URL en beheert op de achtergrond een Google Sheet die als database fungeert. Alle gebruikersinteractie vindt plaats via de web-app interface.

De app stelt de gebruiker in staat om via een overzichtelijk formulier de gegevens voor een nieuwe opdracht (klant, projectdetails, materialen, etc.) in te voeren. Na het opslaan van deze data kan de gebruiker met een druk op de knop de volgende documenten als **PDF genereren** op basis van de aangeleverde Google Doc sjablonen:

1. Offerte
2. Werkbon
3. Factuur

4. Gecombineerde Factuur

1.5 Succescriteria voor Fase 1

1. Er is een functionele, standalone web-app die toegankelijk is via een unieke URL.
2. De app beheert zijn eigen data door te lezen en schrijven naar een Google Sheet die als database op de achtergrond draait.
3. Gebruikers kunnen via het webformulier alle benodigde data voor een opdracht invoeren en opslaan.
4. Vanuit de app kan een correct opgemaakte PDF-offerte, -werkbon, -factuur en -gecombineerde factuur worden gegenereerd.
5. Gegenereerde PDF's worden automatisch opgeslagen in een specifieke map in Google Drive, met een logische bestandsnaam.
6. De Apps Script-code is modulair opgezet, voorzien van commentaar en klaar voor toekomstige uitbreidingen.

2. Plan van Aanpak (Fase 1)

Stap 1: Fundament & Architectuur

- **Actie:** Nieuw standalone Google Apps Script-project aanmaken en publiceren als web-app.
- **Actie:** Google Drive mappenstructuur opzetten binnen de Drive van de gebruiker:
  - `/DocumentenApp/`
    - `/Sjablonen/` (Hier plaatsen we de Google Docs sjablonen)
    - `/Gegenereerde Documenten/`
      - `/Offertes/`
      - `/Werkbonnen/`
      - `/Facturen/`
- **Actie:** Logica ontwikkelen voor de "Database" Sheet. De web-app zal bij de eerste start controleren of de Sheet bestaat en deze anders aanmaken met de volgende tabbladen:
  - **Opdrachten:** Hier komt elke nieuwe opdracht als een rij. Kolommen bevatten o.a. `OpdrachtID`, `KlantID`, `Omschrijving`, `Datum`, `Status`, etc.
  - **Opdrachtregels:** Hier komen de product/dienst-regels die bij een opdracht horen. Kolommen: `RegelID`, `OpdrachtID`, `Omschrijving`, `Aantal`, `Eenheid`, `Prijs`.
  - **Klanten:** Een lijst van alle klanten. Kolommen: `KlantID`, `Klantnaam`, `Adres`, `Postcode`, `Plaats`, `Contactpersoon`, etc.
  - **Configuratie:** Hier slaan we instellingen op, zoals de ID's van de database-sheet, de sjabloondocumenten en de mappen.

Stap 2: Backend Ontwikkeling (`Code.gs`)

- **Actie:** Een `doGet()` functie schrijven die de HTML voor de web-app laadt en serveert.
- **Actie:** Server-side functies ontwikkelen die aangeroepen kunnen worden via `google.script.run`:

    - `getInitialData()`: Haalt de klantenlijst en andere benodigde data op voor het vullen van de web-app interface.
    - `saveOpdracht(opdrachtObject)`: Ontvangt het data-object van het webformulier en schrijft dit weg naar de 'Opdrachten' en 'Opdrachtregels' tabbladen in de database-sheet.
    - `generateDocument(opdrachtID, type)`: De kernfunctie. Haalt de juiste data op basis van `opdrachtID`, kiest het juiste sjabloon (`'Offerte'`, `'Werkbon'`, etc.), vult deze en genereert de PDF.

Stap 3: Frontend Ontwikkeling (bv. `WebApp.html`)

- **Actie:** HTML-formulier ontwerpen voor het invoeren van opdrachtdata (klant selecteren, omschrijving, dynamisch regels voor producten/diensten toevoegen).
- **Actie:** Basis-CSS toepassen voor een schone, overzichtelijke en mobielvriendelijke layout.
- **Actie:** Client-side JavaScript schrijven voor:
    - Het dynamisch vullen van de interface met data (bv. klantenlijst) bij het laden.
    - Het valideren van de invoer in het formulier.
    - Het samenstellen van een `opdrachtObject` op basis van de formulierinvoer.
    - Het aanroepen van de `saveOpdracht` en `generateDocument` functies met `google.script.run`.
    - Het tonen van duidelijke feedback aan de gebruiker (bv. "Bezig met opslaan...", "PDF is aangemaakt en staat in Google Drive.").

Stap 4: Logica voor Documentgeneratie

- **Actie:** De `generateDocument` functie verder uitwerken. Deze functie zal de volgende stappen doorlopen:
    1. Maak een kopie van het relevante Google Doc sjabloon.
    2. Haal de data voor de betreffende opdracht op uit de "Database" Sheet.
    3. Open de body van het gekopieerde document.
    4. Loop door alle placeholders (bv. `[Factuuradres_Klantnaam]`) en vervang deze met de correcte data via `body.replaceText('[placeholder]', data)`.
    5. Voor tabellen/loop-items: ontwikkel een specifieke functie om rijen in een Google Doc tabel dynamisch te vullen.
    6. Sla het document op en sluit het.
    7. Converteer het gevulde Google Doc naar een PDF-blob.
    8. Sla de PDF op in de juiste map in Google Drive met een logische naam (bv. `OFFERTE_2025-001_Klantnaam.pdf`).
    9. Verwijder het tijdelijke, gekopieerde Google Doc.

Stap 5: Testen en Implementatie

- **Actie:** De volledige workflow end-to-end testen door meerdere scenario's te doorlopen voor alle documenttypes.

- **Actie:** Code review uitvoeren om te zorgen dat de code schoon, efficiënt en gedocumenteerd is.
- **Actie:** Een korte handleiding of instructievideo maken voor de eindgebruikers.
- **Actie:** Officiële oplevering van Fase 1.