**Velocircular project**

**Werking**  
- Eerste dinsdag van de maand fietsen ophalen (meekijken naar hoe ze het aanpakken)  
- Deze fietsen moeten geregistreerd worden vb. merk, soort fiets, etc.  
 -> Het registreren van iedere fiets neemt momenteel te veel tijd in beslag  
 -> Daarom tool ontwikkelen om dit proces te versnellen  
- Onderscheid tussen gewone en duurder fietsen nodig voor garantie  
-> Op fietsen van minder goed kwaliteit niet te veel garantie geven   
-> Daarom via kringwinkel (hier maakt garantie minder uit)  
  
Werking fietstool:  
- Fietsen worden op voorhand in verschillende categorieën opgedeeld  
- Iedere categorie heeft een bepaald einddoel  
- Er wordt een foto genomen van de fiets met daarachter een blad met een bepaalde   
 kleur (elke kleur staat voor een categorie)  
 -> Met AI registratie van fietsen versnellen door fietsen op verschillende onderdelen te   
 controleren + kleur van papier helpt op voorhand al met het opdelen in de juiste   
 categorie  
  
Kleuren:  
- ORANJE = Afbraak – voor reserve onderdelen  
- NOG GEEN KLEUR = Productie – studenten fietsen  
- GROEN = SLOOP- oud ijzer  
- ROZE = Verkoop – premium  
- ROOD = VITES - kringwinkel

**Vragen:**- Hoe gaat het in zijn werk met ChatGPT?  
 ->  
  
- Google Firebase gelinkt aan streamlit?  
 -> Google Firebase subscriptie nu niet actief  
 ->   
  
- Is het mogelijk om de uploadlimiet te vergroten?  
- Code voor kleurherkenning blad ->

**Code voor tool (categorieën)**

{

"type": "function",

"function": {

"name": "fiets\_categorie",

"description": "Welke categorie heeft de fiets op basis van de kleur achter de fiets in de afbeelding? Analyseer de achtergrondkleur en bepaal vervolgens de categorie.",

"parameters": {

"type": "object",

"properties": {

"Afbraak": {

"type": "boolean",

"description": "Antwoord hier 'true' als de kleur oranje is gedetecteerd op het papier achter de fiets."

},

"Productie": {

"type": "boolean",

"description": "Antwoord hier 'true' als er geen specifieke kleur is gedetecteerd, wat betekent dat het in de categorie 'Productie' valt."

},

"Sloop": {

"type": "boolean",

"description": "Antwoord hier 'true' als de kleur groen is gedetecteerd op het papier achter de fiets."

},

"Verkoop": {

"type": "boolean",

"description": "Antwoord hier 'true' als de kleur roze is gedetecteerd op het papier achter de fiets."

},

"Kringwinkel": {

"type": "boolean",

"description": "Antwoord hier 'true' als de kleur rood is gedetecteerd op het papier achter de fiets."

}

},

"required": [

"Afbraak",

"Productie",

"Sloop",

"Verkoop",

"Kringwinkel"

]

}

}

}

**Code voor kleurherkenningsfunctie (indien nodig)**from PIL import Image

def detect\_dominant\_color(image\_bytes):

"""Detecteer de dominante kleur in het gebied achter de fiets."""

image = Image.open(BytesIO(image\_bytes))

image = image.resize((100, 100)) # Verklein voor snellere verwerking

# Tel alle kleuren in de afbeelding

colors = image.getcolors(10000) # Maximaal 10.000 unieke kleuren

dominant\_color = max(colors, key=lambda x: x[0])[1] # De meest voorkomende kleur

# Match de dominante kleur met categorieën

if is\_color\_close(dominant\_color, (255, 165, 0)): # Oranje

return "Afbraak"

elif is\_color\_close(dominant\_color, (0, 255, 0)): # Groen

return "Sloop"

elif is\_color\_close(dominant\_color, (255, 0, 255)): # Roze

return "Verkoop"

elif is\_color\_close(dominant\_color, (255, 0, 0)): # Rood

return "Kringwinkel"

else:

return "Productie"

def is\_color\_close(color1, color2, threshold=50):

"""Controleer of twee kleuren vergelijkbaar zijn binnen een drempelwaarde."""

return all(abs(c1 - c2) <= threshold for c1, c2 in zip(color1, color2))

**Probleem met deze code is dat hij kijkt naar de dominante kleur in de afbeelding, maar een klein A4 blad gaat niet de dominante kleur zijn**-> positie van het blad belangrijk  
-> Vb. blad vanboven in de hoek van de afbeelding zodat AI dan enkel in specifiek gebied moet gaan zoeken en niet de volledige foto

**Integratie in call\_gpt\_model**

base64\_image\_bytes = base64.b64decode(base64\_image)

category = detect\_dominant\_color(base64\_image\_bytes)

result\_as\_dict["fiets\_categorie"] = category