**Hoe Maak Je een Website met Python en Flask: Een Stap-voor-Stap Handleiding voor Beginners**

In deze tutorial leer je hoe je een eenvoudige website kunt maken met Python en het Flask-framework. Flask is een micro-framework dat eenvoudig te begrijpen is, zelfs voor mensen zonder coderingservaring. We doorlopen stap voor stap het proces, zodat je een werkende website kunt creëren zonder dat je veel technische kennis nodig hebt.

**Wat is Flask?**

Flask is een lichtgewicht webframework voor Python. Het stelt je in staat om snel en eenvoudig een webapplicatie te bouwen. Omdat het zo simpel is, is het een perfecte keuze voor beginners die willen leren hoe ze websites kunnen maken zonder diepgaande kennis van webontwikkeling.

**Benodigdheden**

* **Python**: Zorg ervoor dat je Python op je computer hebt geïnstalleerd. Je kunt het downloaden van de officiële website: [python.org](https://www.python.org/).
* **Text editor**: Een editor zoals VSCode, Sublime Text, of zelfs de ingebouwde editor van Python, IDLE, is genoeg.
* **Flask installeren**: We installeren Flask via de terminal of command prompt. Je kunt dit doen door de volgende opdracht uit te voeren:
* pip install Flask

**Stap 1: Maak een Basis Flask Project**

1. **Maak een map voor je project**  
   Maak een nieuwe map waar je je websiteproject wilt opslaan. Noem deze bijvoorbeeld mijn\_website.
2. mkdir mijn\_website
3. cd mijn\_website
4. **Maak een Python-bestand**  
   Maak een nieuw bestand met de naam app.py in de map mijn\_website. Dit bestand bevat de code voor je website.
5. **Schrijf de eerste Flask-code**

Open app.py en schrijf de volgende code:

from flask import Flask

# Maak een Flask-applicatie

app = Flask(\_\_name\_\_)

# Stel een route in

@app.route('/')

def home():

return "Welkom op mijn website!"

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)

Wat gebeurt hier?

* + We importeren Flask.
  + We maken een nieuwe Flask-applicatie aan met Flask(\_\_name\_\_).
  + We stellen een route in, wat betekent dat we aangeven welke functie uitgevoerd moet worden wanneer een bepaalde URL wordt bezocht. In dit geval, wanneer de gebruiker naar de root (/) gaat, wordt de functie home() aangeroepen en tonen we de tekst "Welkom op mijn website!".
  + De app.run(debug=True) zorgt ervoor dat de server in de ontwikkelmodus draait, zodat je wijzigingen snel kunt testen.

**Stap 2: De Website Laten Werken**

1. **De server starten**  
   Open de terminal (of command prompt) en navigeer naar je projectmap. Voer dan het volgende commando uit om de server te starten:
2. python app.py
3. **De website bekijken**  
   Als alles goed is gegaan, zou je de volgende boodschap in je terminal moeten zien:
4. \* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)

Dit betekent dat je server actief is op http://127.0.0.1:5000/. Open een webbrowser en ga naar dit adres. Je zou de boodschap "Welkom op mijn website!" moeten zien.

**Stap 3: HTML-pagina's Toevoegen**

De website die we tot nu toe hebben, is heel eenvoudig. Laten we het uitbreiden door HTML-pagina's toe te voegen.

1. **Maak een templates-map**  
   Flask gebruikt de templates-map om HTML-bestanden op te slaan. Maak een nieuwe map in je project met de naam templates:
2. mkdir templates
3. **Maak een HTML-bestand**  
   Maak een nieuw bestand in de templates-map met de naam home.html:
4. <!DOCTYPE html>
5. <html lang="nl">
6. <head>
7. <meta charset="UTF-8">
8. <meta name="viewport" content="width=device-width, initial-scale=1.0">
9. <title>Mijn Website</title>
10. </head>
11. <body>
12. <h1>Welkom op mijn website!</h1>
13. <p>Dit is een eenvoudige website gebouwd met Python en Flask.</p>
14. </body>
15. </html>
16. **Werk de Flask-code bij**  
    Nu gaan we de Flask-app bijwerken om de HTML-pagina te renderen in plaats van alleen tekst terug te geven. Open app.py en vervang de home()-functie door de volgende code:
17. from flask import Flask, render\_template
18. app = Flask(\_\_name\_\_)
19. @app.route('/')
20. def home():
21. return render\_template('home.html')
22. if \_\_name\_\_ == '\_\_main\_\_':
23. app.run(debug=True)

De functie render\_template('home.html') vertelt Flask om het HTML-bestand home.html weer te geven in plaats van de eenvoudige tekst.

1. **Herstart de server**  
   Stop de server door CTRL+C in te drukken en start hem opnieuw:
2. python app.py
3. **De pagina bekijken**  
   Ga opnieuw naar http://127.0.0.1:5000/ in je webbrowser. Je zou nu de HTML-pagina met de welkomstboodschap moeten zien.

**Stap 4: Een Eenvoudige Navigatie Toevoegen**

Laten we de website iets interactiever maken door een tweede pagina toe te voegen.

1. **Maak een nieuwe route**  
   Voeg in app.py een nieuwe route toe voor de pagina "Over mij":
2. @app.route('/over')
3. def over():
4. return render\_template('over.html')
5. **Maak een nieuw HTML-bestand voor de "Over mij" pagina**  
   Maak in de templates-map een nieuw bestand over.html:
6. <!DOCTYPE html>
7. <html lang="nl">
8. <head>
9. <meta charset="UTF-8">
10. <meta name="viewport" content="width=device-width, initial-scale=1.0">
11. <title>Over Mij</title>
12. </head>
13. <body>
14. <h1>Over Mij</h1>
15. <p>Hier kun je meer informatie vinden over de maker van deze website.</p>
16. <a href="/">Terug naar de homepage</a>
17. </body>
18. </html>
19. **Link de pagina's aan elkaar**  
    We willen nu een navigatielink naar de "Over mij"-pagina toevoegen op de homepage. Pas home.html aan door een link toe te voegen:
20. <h1>Welkom op mijn website!</h1>
21. <p>Dit is een eenvoudige website gebouwd met Python en Flask.</p>
22. <a href="/over">Ga naar de Over mij-pagina</a>
23. **Herstart de server**  
    Na het toevoegen van de nieuwe pagina's, start je de server opnieuw:
24. python app.py

Ga naar de homepage en klik op de link naar de "Over mij"-pagina. Als alles goed is gegaan, kun je nu tussen de pagina's navigeren.

**Stap 5: Je Website Verdere Functionaliteit Geven**

Nu je de basisprincipes van Flask hebt geleerd, kun je verder experimenteren en de website uitbreiden:

* **Formulieren**: Voeg formulieren toe voor gebruikersinvoer.
* **CSS**: Stijl je pagina’s met CSS om ze aantrekkelijker te maken.
* **Interactiviteit**: Maak gebruik van JavaScript om je website dynamisch te maken.

**Conclusie**

In deze tutorial hebben we geleerd hoe je een eenvoudige website kunt maken met Python en Flask. Je hebt geleerd hoe je routes instelt, HTML-pagina's maakt en links toevoegt om pagina's aan elkaar te verbinden. Flask is een krachtig, maar eenvoudig framework waarmee je snel aan de slag kunt, zelfs zonder veel ervaring in coderen.

Nu kun je verder experimenteren en je website naar wens uitbreiden. Veel succes met je project!