**Flask Exercise**

**Goal: To use the Python library Flask to integrate Python with HTML and CSS in the browser**

***Alternative instructions available here:*** [*https://flask.palletsprojects.com/en/3.0.x/tutorial/*](https://flask.palletsprojects.com/en/3.0.x/tutorial/)

**First, install Flask:**

pip install flask

**A screenshot of a computer program

Description automatically generated**

**Next, you could make a folder (for your Flask app files) via the GUI, or use VSC terminal:**

mkdir my\_flask\_app

**Then point towards this new directory (folder) by changing directory (cd):**

cd my\_flask\_app

**In my particular case, I created my folder in another repository, so had one folder higher.**

**A blue background with white text

Description automatically generated**

**Then within this folder you’ll want to create a python file (.py) that will import the flask library.**

**Recommended that you start by creating an app.py file in your new directory/folder.**

**Then within your app.py file you can copy/paste the following to get started:**

**Folder structure:**

\my\_flask\_app

**\app.py**

**app.py:**

from flask import Flask

# First Flask App

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

@app.route('/')

def home():

return "Hello, Flask!"

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

app.run(debug=True)

**My GUI view:**

**A screenshot of a computer program

Description automatically generated**

**Then, to test this works, go back to the VSC terminal (make sure you’re still in the same directory), and run the following:**

python app.py

***A screenshot of a computer program

Description automatically generated***

**If your web browser hasn’t opened automatically, you should be able to click on the localhost link provided, or copy this link into your web browser. By default it is usually:** [**http://127.0.0.1:5000**](http://127.0.0.1:5000)

**This should then display the following: (I’ve zoomed in for illustration purposes):**

***A screenshot of a chat

Description automatically generated***

**Now, to add HTML, we need to create a templates folder. This is the same with the Django library.**

**Again, this could be done via the GUI or via VSC terminal mkdir command.**

mkdir templates

**Then in the templates folder, you’ll want to create a html file. For example, you could use the below code to get you started:**

**Folder structure:**

\my\_flask\_app  
 \templates

**\home.html**

\app.py

**home.html:**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Home</title>

</head>

<body>

<h1>Hello, Flask with Templates!</h1>

</body>

</html>

**A screen shot of a computer program

Description automatically generated**

**Folder structure:**

\my\_flask\_app  
 \templates

\home.html

**\app.py**

**We’ll also want to change the app.py file to include the render\_template, and the return value of the home function. Changes are highlighted in green:**

from flask import Flask, render\_template

# First Flask App

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

@app.route('/')

def home():

return render\_template('home.html')

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

app.run(debug=True)

**If your terminal running the HTTP server is still running, you should be able to return your web browser and see that the contents should have changed to display the HTML form (again I’ve zoomed in for illustration purposes):**

***A screenshot of a computer

Description automatically generated***

**Next step: add another route! For example, an ‘about’ page using another ‘route’:**

**This involves a new html file representing the about page, which will need to be added in the templates folder:**

\my\_flask\_app  
 \templates

\home.html  
 **\about.html**

\app.py

**Here’s some code for the about.html file:**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>About</title>

</head>

<body>

<h1>Welcome to the about page</h1>

</body>

</html>

A screen shot of a computer screen

Description automatically generated

**Here’s the updated app.py file with a new route for the /about page. Note that this about   
Changes are highlighted in green:**

from flask import Flask, render\_template

# First Flask App

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

@app.route('/')

def home():

return render\_template('home.html')

@app.route('/about')

def about():

return render\_template('about.html')

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

app.run(debug=True)

**Remember that the about page is defined to be a child of the home page (/about)**

[**http://127.0.0.1:5000/about**](http://127.0.0.1:5000/about)

***A screenshot of a computer

Description automatically generated***

**To go back to the home page, this is the root folder directory:**

[**http://127.0.0.1:5000**](http://127.0.0.1:5000)

**A screenshot of a computer

Description automatically generated**

**Furthermore, you can bring in your HTML knowledge now and write hyperlink reference tags for easier navigation between HTML pages:**

**Remember that in this current directory:   
- home is "/"  
- about is "/about"**

**As this is HTML code, you can update the home and about html pages without having to change any python code:**

**A simple example of what you could add to home.html (remember you’re linking to the about page here)**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Home</title>

</head>

<body>

<h1>Hello, Flask with Templates!</h1>

<a href = "/about">about</a>

</body>

</html>

**A simple example of what you could add to about.html (remember you’re linking to the home page here):**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>About</title>

</head>

<body>

<h1>Welcome to the about page</h1>

<a href = "/">home</a>

</body>

</html>

**With more pages of course, you’ll scale up to having a navigation bar, but for this first app, this exercise is to demonstrate how to use HTML in conjunction with Python’s Flask library.**

**Let’s check that we can navigate between our two pages via the links given:**

**A screenshot of a computer

Description automatically generated**

**Clicking on the about link takes me back to the home page, and furthermore, I can also get to the about page via the link provided on the home page!**

**A screenshot of a computer

Description automatically generated**

**Now let’s add some CSS, so we can improve the presentation:**

**Like the Django structure, we’ll need to add a new folder named ‘static’:**

**Like before, this can be achieved via the folder GUI (create a folder like you usually would in a folder directory), or via the VSC terminal command:**

mkdir static

**This folder should inside your root directory for your flask app, but outside of the templates folder, like the below:**

\my\_flask\_app

\static

**\style.css**

\templates

\home.html  
 \about.html

\app.py

**As seen above, in this static folder is where you should place your css files.**

**For this first app in Flask, let’s add some styling for the <h1> tags:**

**Here’s some example code for the ‘style.css’ file:**

h1 {

color: #243bb9;

}

**Reminder of the file directory:**

**A screenshot of a computer

Description automatically generated**

**Then in your HTML files, you’ll need to add a link to an external style sheet, and also make use the ‘jinja’ {{ and }} tags:**

**Here’s the updated home.html script:**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Home</title>

<!-- Link to the CSS file -->

<link rel="stylesheet" href="{{ url\_for('static', filename='style.css') }}">

</head>

<body>

<h1>Hello, Flask with Templates!</h1>

<a href = "/about">about</a>

</body>

</html>

**Here’s the updated about.html script:**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>About</title>

<!-- Link to the CSS file -->

<link rel="stylesheet" href="{{ url\_for('static', filename='style.css') }}">

</head>

<body>

<h1>Welcome to the about page</h1>

<a href = "/">home</a>

</body>

</html>

**Revisiting the web browser you should then see the font colour change for the h1 tags in both pages:**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**So now that we’ve managed to integrate both HTML and CSS into our first Flask app, let’s write and utilise a Python script. Let’s create a python script in a separate file, named data\_processing.py:**

\my\_flask\_app

\static

\style.css

\templates

\home.html  
 \about.html

\app.py

**\data\_processing.py**

def process\_data():

# Simulate some data processing

l = [1,2,3,4,5,6,7]

total = sum(l)

return total

**Now we need to update the app.py script to import the python function ‘process\_data’ from the data\_processing.py file. The returned result also needs to be passed to the home.html page.**

from flask import Flask, render\_template

from data\_processing import process\_data # Import your script

# First Flask App

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

@app.route('/')

def home():

result = process\_data() # Call the function from the script

return render\_template('home.html', result=result)

@app.route('/about')

def about():

return render\_template('about.html')

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

app.run(debug=True)

**Now we let’s update the home.html page to display the result passed, again using the {{ and }} tags:**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Home</title>

<!-- Link to the CSS file -->

<link rel="stylesheet" href="{{ url\_for('static', filename='style.css') }}">

</head>

<body>

<h1>Hello, Flask with Templates!</h1>

<h2>{{ result }}</h2> <!-- Display the processed data result -->

<a href = "/about">about</a>

</body>

</html>

**Let’s check our web browser to see whether this sum performed in a python script is displayed via our html page.**

**All going well you should see a numerical result:**

**A screenshot of a computer

Description automatically generated**

**Flask Exercises**

1. …

2. …

3. …