HTML FORM IN FLASK WEB APPLICATION

HTML Form

- It is possible to connect HTML form with flask web application
- This is done by calling the special built-in method named **render_template()**

Structure of the Flask Project

```
Project Name
        -- /.css
        -- /templates
                -- /.html
        -- /app.py
```

HTML TEMPLATES

- In order to use design files like HTML, first we need to create a special folder named **templates** in the current flask project
- HTML files can be created and stored inside of the **templates folder** of the current flask project.

Rendering Template in Flask

- In order to call HTML / CSS files in flask web application, a special built-in method named **render_template()** will be used
- This method will take one plus arguments, where
  - First argument is the **HTML file**
  - Second argument is the **optional data to be displayed in the web page**.

Jinja Template

- It is a **popular python library used by the web frameworks** like Flask, Django, FastAPI to serve HTML pages in an efficient and secure way.
- This is the **powerful web template engine for python** which is created by flask people. It is the default template engine in flask.
- It is mainly used to **generate a dynamic HTML / XML page to the user via HTTP response**

Calling Static HTML in Python - Jinja Template

- It is possible to **call / load a static HTML web page in python flask using jinja template** through render_template() function

Sample Example

```
    from flask import Flask, render_template
    app = Flask(__name__)
# home page using route()
    @app.route('/')
    def welcome():
        return render_template('home.html')
# other page using route()
    @app.route('/about')
    def about():
        return "<h4>Welcome to Flask Web<h4>"
```

Python View Function

- The method **welcome()** is a **view function** which is ready to call **home.html via render_template()** method when the application will be called.

HTML FORM PROCESSING IN PYTHON

- It is possible to create web form in python flask
- To create a form in python flask, use the syntax below

Syntax

```
<form action="route-name" method="post">
…
</form>
```

Where,

- route-name is the response of target web page.

FORM ATTIRBUTES

- The form tag has two main important attributes. They are
    1. Action
    2. method

1. Action Attribute

- It is an important attribute of <form> tag and ready to handle the HTML form data
- It is a response URL file which is mainly used to **send the user request to server**.

2. Method Attribute

- It is used to specify the HTTP method for form submission
- Most popular used methods are **get** and **post**

ACCESSING HTML FIELDS (WIDGETS) IN PYTHON FLASK

- Data associated in the HTML form is sent as the HTTP request to server
- Flask provides two or more number of ways to get / access form request data. They are
    1. Using **request.form[“widget-name”]**
    2. Using **request.form.get(“widget-name’)**

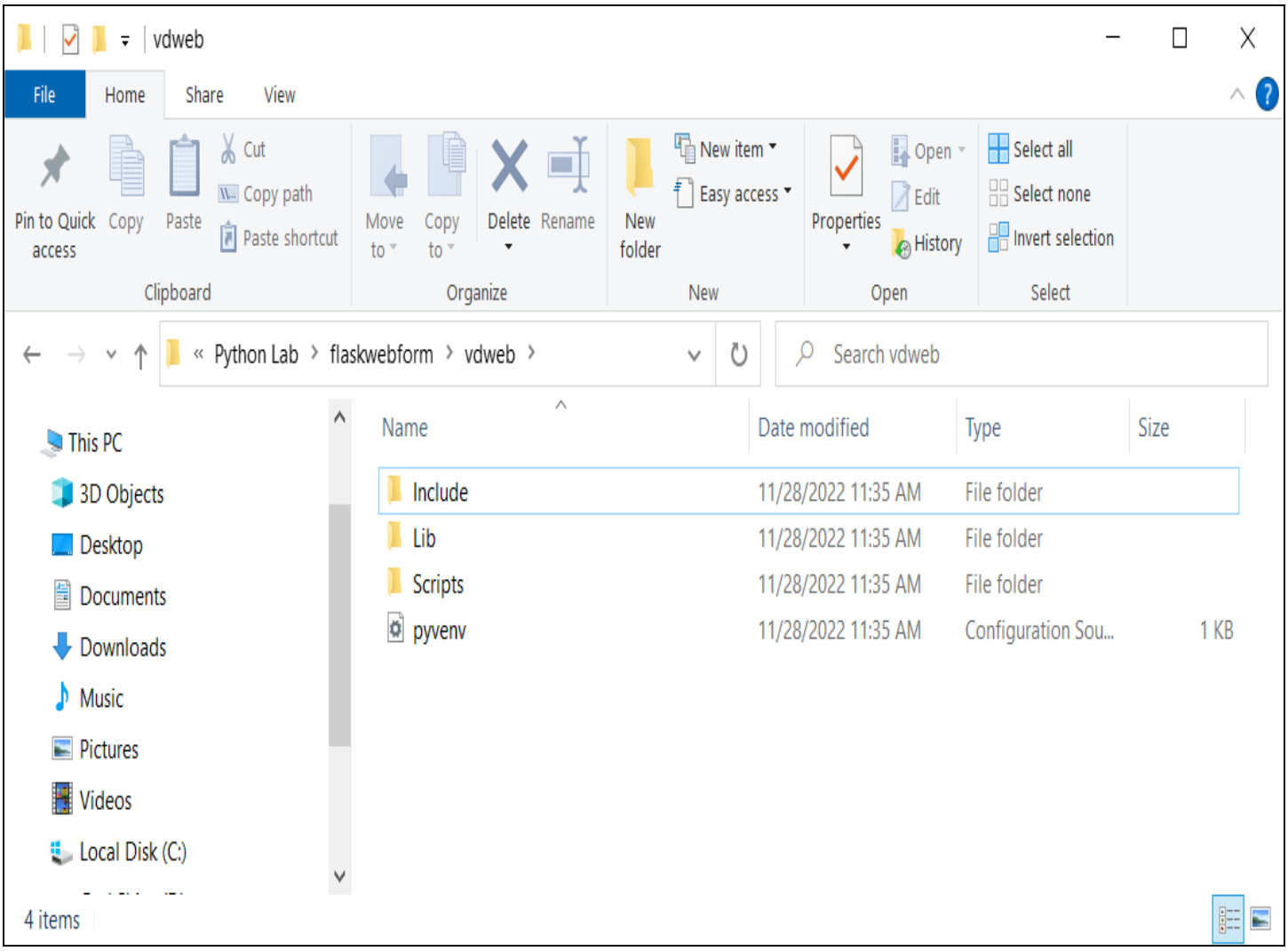
Where,

- Widget-name represents the HTML form tags like button, label, checkbox, radio button, list, etc,…
- The module **request** must be imported in the code before getting the client data.

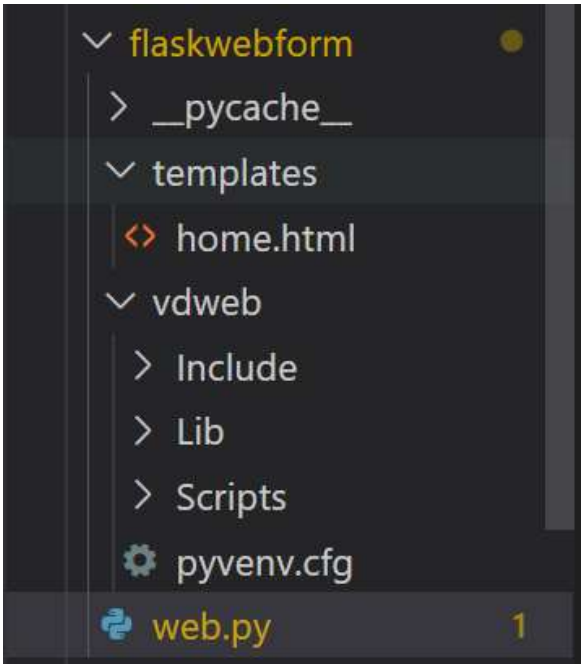
I. EXAMPLE OF HTML FORM BASED FLASK WEB APPLICATION

Application Type	:	Web Application
Web Framework	:	**Flask**
Language	:	Python (Python 3)
Python SDK	:	**3.10.5**
Tools Used	:	VSC Editor
Tested OS	:	Windows 10
Virtual Directory	:	**vdweb**
Number of Routes	:	2
Project Folder Name	:	**flaskwebform**

CONTENTS OF DIRECTORY AND VIRTUAL DIRECTORY



CURRENT PROJECT STRUCTURE (flaskwebform)



1. SOURCE CODE

FRONT END – UI DESIGN

(home.html)

```
<html>
   <body>
   <center>
      <form action="fact" method="post">
      <h1 style="background-color:#7ac70c;font-size: 20px;color:#050f2c
">Flask Web Application - Factorial Finder</h1>
         Enter a number:<br/>
         <input type="text" name="tb" /><br/>
         <input type="submit" value="Click Here"/>
      </form>
   </center>
   </body>
</html>
```

This will call the python route named **/fact**.

BACK END – APPLICATION LOGIC
(web.py)

Route 1	:	Home Page (/)
Route 2	:	Content Page (/fact)

```
# load the flask library
from flask import Flask, render_template, request
# create an object for Flask class
obj=Flask(__name__)
# create a home page using route
@obj.route("/")
# call HTML web page in home route
def index():
    return render_template("home.html")
# create another web page for factorial task
@obj.route("/fact", methods=['GET', 'POST'])
# view function for static HTML / dynamic web page
def dispfact():
# get the user input via form object
    num=request.form["tb"]
# convert string to intger
    n=int(num)
    f=1
# code for factorial
    for i in range(0,n):
        f=f*(i+1)
# display the result as HTTP response in the web page
    return "<h1 style='color:#a626aa;'><center>Factorial is:
"+str(f)+"</center></h1>"
```

Route 1

Route 2

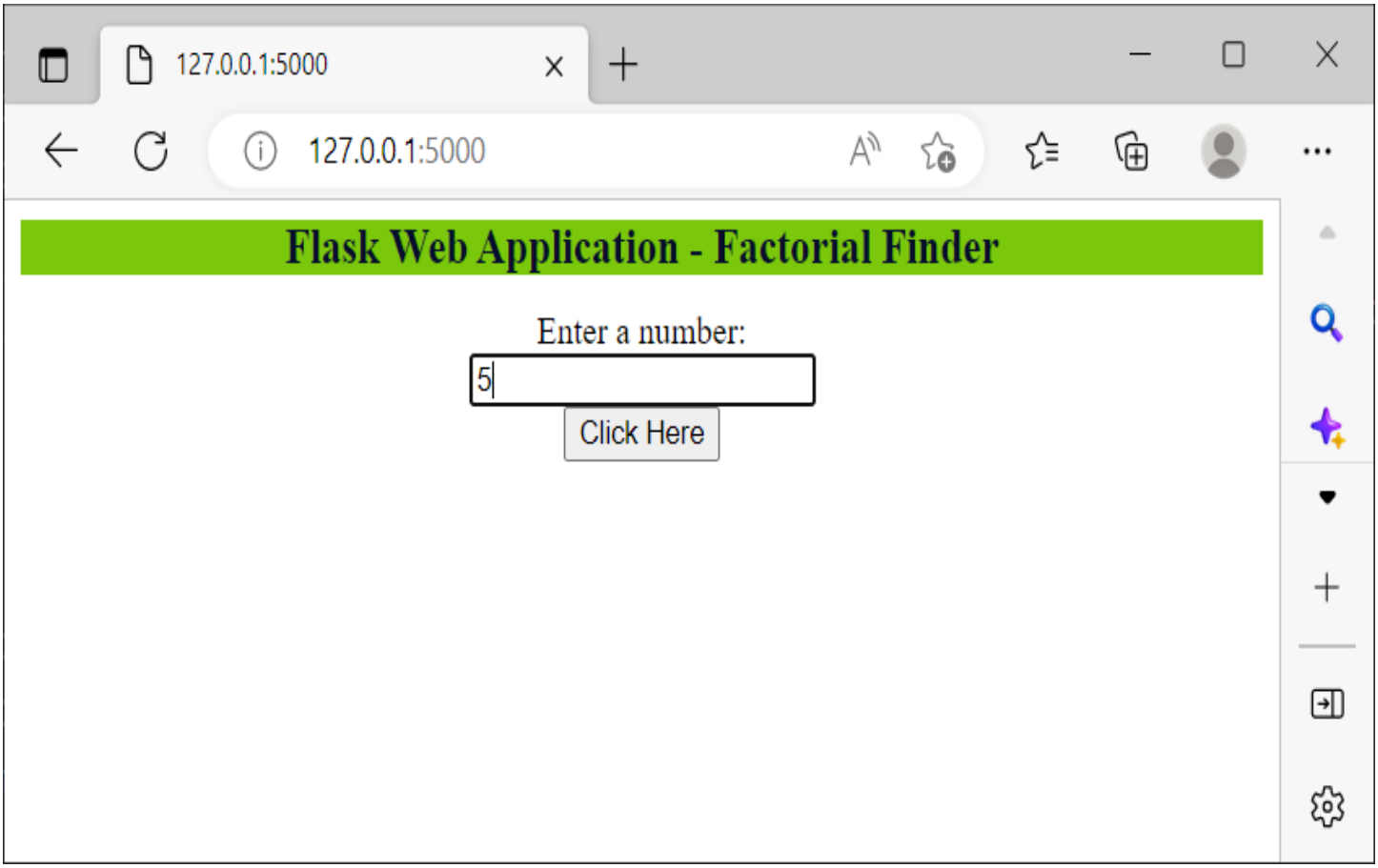
2. RUN THE FLASK WEB APPLICATION

```
PROBLEMS 1   OUTPUT   DEBUG CONSOLE   TERMINAL   JUPYTER

(vdweb) PS F:\VSC Projects Location\Python Lab\flaskwebform> flask run
 * Serving Flask app 'web.py'
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment.
nstead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
127.0.0.1 - - [28/Nov/2022 16:56:22] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [28/Nov/2022 16:57:42] "POST /fact HTTP/1.1" 200 -
```

3. OUTPUT

3.1 HOME PAGE



3.2 FACTORIAL RESULT