Python - Bottle Framework for Web App Development

Setup

Turn on Errors in PHP.ini

```
python3 -m pip install bottle

Create "static" folder in root directory for static files

Create "views" folder in root directory for template files
```

Very Simple Web App

This lab simply creates a basic HTML web page to verify that Bottle is functioning properly.

lab-simple.py
```
from bottle import run, route

@route('/')
def index():
    page = '''
            <h1>Web Page</h1>
            <p>This is my very simple web page</p>
            '''

    return page

run(host='localhost', port=8080)
```

Note App

This lab allows you to build an HTML note taking app. You create new notes using and HTML form, those notes are saved in a CSV file, and then that file is read and the old notes are written to the page with HTML.

We use || as the separator value in the CSV file so that when we read from the file we do not mistake a users comma as a separator value.

lab-note.py

```
from bottle import run, route, request, redirect, post

@route('/')
def index():
    form = '''
            New Note:
            <br>
            <form action="/process" method="post">
            Title: <input type="text" name="title">
            <br>
            Note: <input type="text" name="note">
            <input type="submit">
            </form>
            '''
    try:
        with open('note-app.txt', 'r') as file:
            data = file.readlines()
            data_html = ''
            for record in data:
                value = record.split('||')
                data_html = f'''
                                <strong>{value[0]}</strong>
                                <p>{value[1]}</p>
                                <hr>
                                {data_html}
                            '''
    except:
        data = ''

    page = f'{form} <br> {data_html}'

    return page

#CONTINUES ON NEXT PAGE
```

```
@post('/process')
def process():
    title = request.forms.get('title')
    note = request.forms.get('note')
    with open('note-app.txt', 'a') as file:
        file.write(f'{title}||{note}\n')
    redirect ('/')

run(host='localhost', port=8080)
```

Host Ping App

This lab uses Dynamic Filters to accept a host name or IP Address, and then a timer value. The host will then be used as a value to Ping and the timer value will be used for how often to repeat.

If a ping is successful a color value will be set to Green, and if not it will be set to Red. We then create a web page where we show the value pinged with the appropriate background color, and then the ping response within <pre> tags.

The page will auto refresh, and re ping based on the interval value.

NOTE: This uses the OS Module to send commands to the OS. Verify the commands are correct for your OS.

lab-ping.py

```
from bottle import run, route
import os

@route('/<host>/<interval:int>')
def index(host, interval):
    command = f'ping -c 1 {host}'

    response = os.popen(command).read()

    # '1 received' for Ubuntu
    if '1 packets received' in response:
        color = 'green'
    else:
        color = 'red'

    page = f'''
                <meta http-equiv="refresh" content="{interval}">
                <h1>Ping App</h1>
                <h3 style="background-color:{color};">{host}</h3>
                <pre>{response}</pre>
            '''

    return page

run(host='localhost', port=8080)
```