Python Programming: Flask

Flask: Final Practice Problems

In this homework, you're going to write code for a few problems.

You will practice these programming concepts we've covered in class:

- Rendering templates.
- Creating an API.
- Making `GET`/`POST` requests.

Deliverables

For each of the challenges listed below, you will create a new `.py` file and write code to solve the problem. For example, you would create `problem1.py` with your solution code to the first problem. Run the file from the command line to check your work.

Reminder: On your laptop, you can run the file from your command line with the following:

```
python problem1.py
```

> **Hint**: After finish writing your code, launch your server, go into your browser, and be sure that your Flask app is outputting the intended data.

Requirements:

- By the end of this, you should have four different `.py` files (three for the first problem and one for the second problem).

Problem 1: "Rendering Like Rembrandt"

Skill You're Practicing: Using templates to render Python.

Create a Flask app that renders an HTML template. In the template, display a greeting and a `name` variable (don't forget to pass the template the argument!).

Then, create a CSS change the color of the font of your template's variable.

Example Test Code

```
render_template('index.html', name=user)
```

Example Test Output

```
"Hi there Akilah. It's great to see you today!"
```

Hint 1:

Remember: Templates for variables use the double brackets `{{}}`.

Hint 2:

Don't forget the module `render_template`.

Hint 3:

Your directory should look like:

```
project
|
|
└──app
|   |   problem1.py
|   |
|   |
|   └──templates
|   |    └── index.html
|   |
|   |
|   └──static
|       └──style.css
```

Problem 2: "A Detective, a PI"

Skill You're Practicing: Creating an API.

Write a Flask app that makes a `GET` request and returns a JSON of one of the items in a list.

Example Test Code

```
return jsonify({'pie ingredient': 'ingredients[0]'})
```

Example Test Output

```
{'pie ingredient': 'apples'}
```

Hint 1:

Refer to your class notes from the Variables lesson for how to read in a variable directly in a Flask app.

Hint 2:

There are two modules that we'll need to execute this, in addition to our standard `from flask import Flask`: `jsonify`, and `requests`.

Hint 3:

Try passing the variable name into your function, as well as making that your endpoint in the route.

Problem 3: "The POST Man Deliverth"

Skill You're Practicing: Creating an API.

Write a Flask app that makes a `POST` request and returns a JSON of one of the items in a list.

Example Test Code

```
ingredients.append(ingredient)
return jsonify({'pie ingredient': ingredients})
```

Example Test Output

```
{'pie ingredient': 'apples'}
```

Hint 1:

Refer to your class notes from the Variables lesson for how to read in a variable directly in a Flask app.

Hint 2:

There are two modules that we'll need to execute this, in addition to our standard `from flask import Flask`: `jsonify`, and `requests`.

Hint 3:

Try passing the variable name into your function, as well as making that your endpoint in the route.

Hint 4:

```
request.get_json()
```