Flash Messages and Form Validation in Flask

Contents:

Part 1: Connect to Git on the console

Check to see if git is installed. In the terminal window in VSC:

```
git --version
```

If not installed:
On a Mac computer, it should prompt you to install it.

On a Windows computer, go to https://git-scm.com/download/win When you click on this the download should start automatically. Follow the install instructions.

Configure git:

```
git config --global user.name "your_username"

git config --global user.email "youremail_address@example.com"

git config --global --list
```

Part 2: Flash messages to give feedback to the user

Create a folder in which to save this tutorial.
Open the folder in Visual Studio Code.
In the terminal window, download the code by using this code:

```
git clone https://github.com/msjones3/flask_labs.git -b flash
```

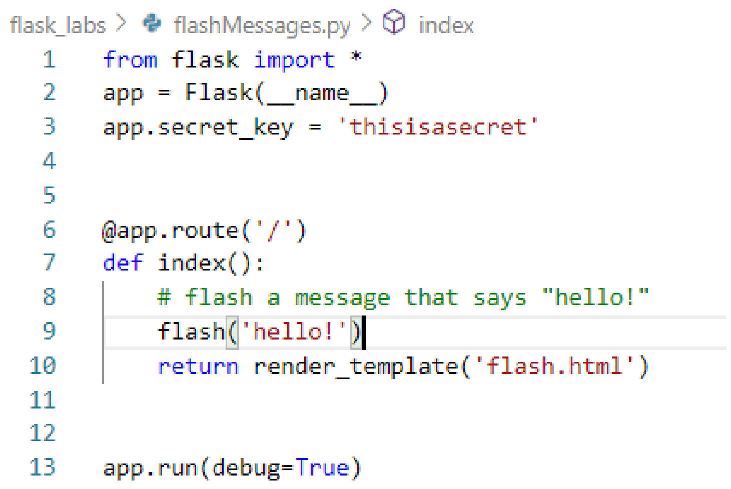
Navigate into this folder by typing:

```
cd flask_labs
```

Install flask

```
pip install flask
```

Update the flash.py file to flash the message ‘hello’

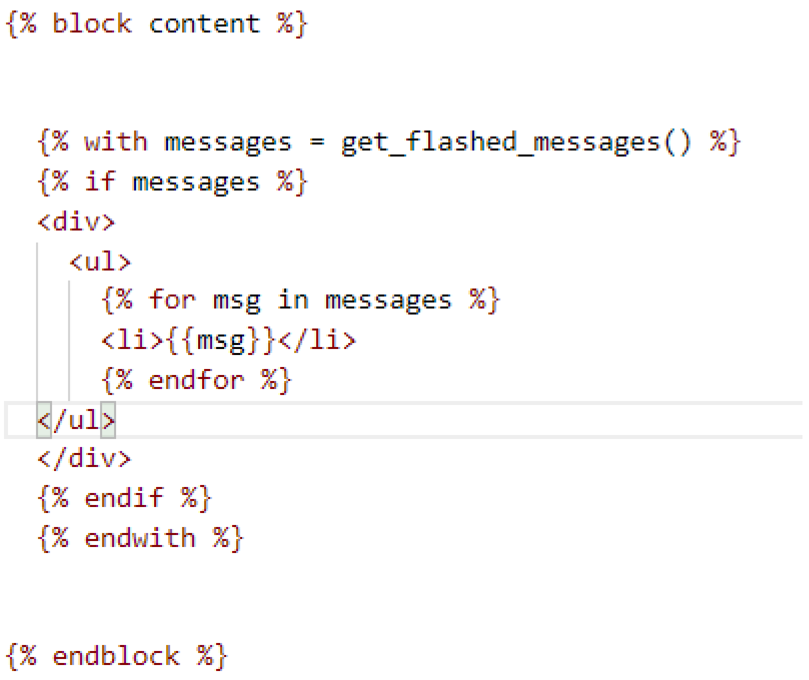


```python
from flask import *
app = Flask(__name__)
app.secret_key = 'thisisasecret'


@app.route('/')
def index():
    # flash a message that says "hello!"
    flash('hello!')
    return render_template('flash.html')


app.run(debug=True)
```

We will now need a container to hold our messages in the flash.html template:

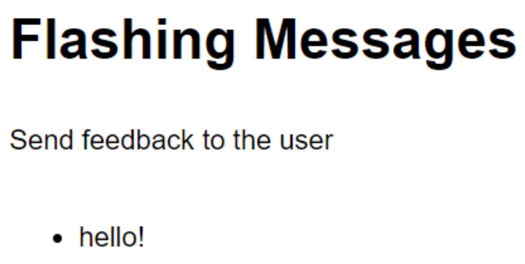


```
{% block content %}


  {% with messages = get_flashed_messages() %}
  {% if messages %}
  <div>
    <ul>
      {% for msg in messages %}
      <li>{{msg}}</li>
      {% endfor %}
  </ul>
  </div>
  {% endif %}
  {% endwith %}


{% endblock %}
```

When you run the file now, you should see the following:

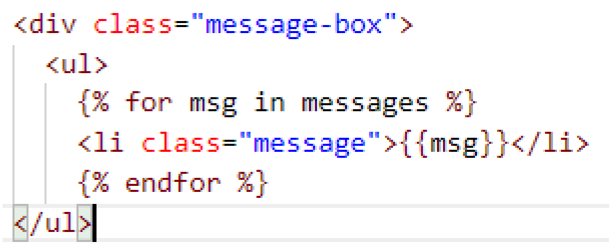


Flashing Messages

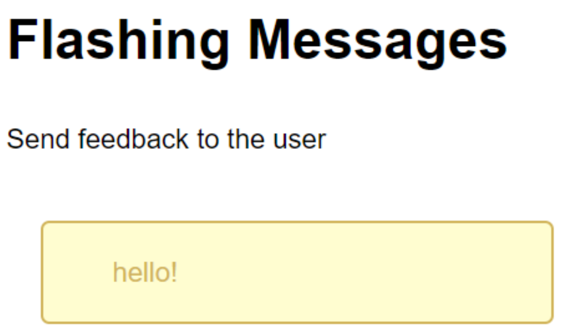
Send feedback to the user

- hello!

CSS Styling
Let’s add some classes to the html tags so that we can style this message. I have already create the css code for the classes ‘message-box’ and ‘message’



```
<div class="message-box">
  <ul>
    {% for msg in messages %}
    <li class="message">{{msg}}</li>
    {% endfor %}
</ul>
```



Flashing Messages

Send feedback to the user

hello!

Part 3: Simple Form Validation in Flask

Create a new folder called validation
In VSC, open this folder.
In the terminal, download the starter files by using this code:

```
git clone https://github.com/msjones3/flask_labs.git -b validate
```

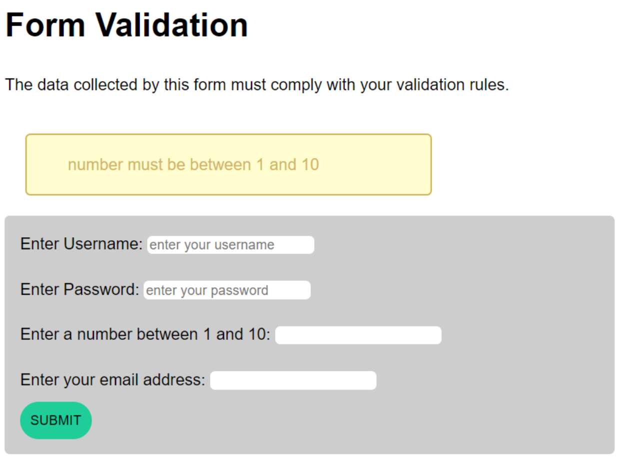
Navigate into this folder by typing:

```
cd flask_labs
```

Install flask

```
pip install flask
```

1. Start by adding the code to the validation.html page (as above) so that you can view errors.
2. Install Flask and run the application.
3. Submit the form and see what happens.
4. Submit the form again, but this time use the username 'admin' and the password 'secret'
5. Study the code, and then write some more validation. After each one, test and run.
    a. Username cannot be blank
    b. Password cannot be blank
    c. Number must be a number
    d. Number must be between 1 and 10
    e. Email must have an @ symbol in it



Form Validation

The data collected by this form must comply with your validation rules.

number must be between 1 and 10

Enter Username: enter your username

Enter Password: enter your password

Enter a number between 1 and 10:

Enter your email address:

Part 4: Sanitizing inputs

Sanitizing inputs means we will just change the value of the input after the form has been submitted. This is to make sure that the data is both safe (free from SQL injection attacks) as well as desirable (consistent and high-quality data).

1) Let’s start with the email form. People often use capital letters throughout their email address.
    a) Save the email as a variable called “email”.
    b) Change the email address to be only lowercase letters.
    c) Test to see that this works by printing to the terminal.
2) Add a textarea field to the form.
    a) Make sure that users cannot enter any more than 120 chars.
    b) Make the data a little less nefarious by changing all single quotes to double quotes.
    c) Test to see if it worked.