CHECKLIST

Basic Form Validation

Objectives:

1. Why should we have our server *validate* user input before using it or adding it to the databa
2. How do we validate user input?
3. If the user input fails validation, what should our next steps be?
4. What are *flash messages* in the context of Flask, what are they used for, and how do we use

---

Form validation is a key component of any back-end developer's arsenal. **Validation** is more of a bunch of new code to learn. Here's what we think are the most important concepts in form valic

- Logic: what data do we want to validate?
- Checking if the data is present
- Making sure the data is in the correct format
- Sending the user to the correct destination whether their data is valid or not
- Alerting the user of their errors (if they exist)

The first validation tool we are going to learn is how to check if an input field is empty and how

Getting Started

The most important validation tool is the if/else statement. Every validation is conditional! It sho data we want to do something or **ELSE**, we need to do something different! Form validation cer combined with functions that return **TRUE** or **FALSE** depending on if the data we give them is v

Let's create a sample application with a very simple form. Let's call this example **basic_validatio**

Inside of your project create the server.py file like so.

/basic_validation/server.py

```
from flask import Flask, render_template, redirect, request, session
app = Flask(__name__)
app.secret_key = 'KeepItSecretKeepItSafe'
@app.route('/')
def index():
    return render_template('index.html')
@app.route('/process', methods=['Post'])
def process():
  #do some validations here!
    return redirect('/')
if __name__=="__main__":
    app.run(debug=True)
```

Now we need to make our Index.html file! Based on the information above we need our index.ht to /process with the method of POST.

/basic_validation/templates/index.html

```
<html>
<head>
  <title>Basic Validation Example</title>
</head>
<body>
  <h1>Enter a Valid(Any) Name!</h1>
  <form action='/process' method='POST'>
    Name:<input type="text" name="name">
    <input type="submit" value="Submit">
  </form>
</body>
</html>
```

What an exciting application! Right now if we submit a name all that happens is we get redirecte name and we will still get redirected! **Our goal will be to validate whether the name was valid (message if it was. If there was no name submitted we will display a validation error.**

BACK TO TRACKS

1. Flask Fundamentals 71%
(/m/72/5461)

CHECKLIST

Validation Step 1 -- Conditionals

When validating our form data, we always need to use conditionals. At the base level, every valid conditional statement in pseudo code would look something like this:

If the name field in the POST data is empty:

Display validation error

Else:

Display success message

How should we check if the name field is empty? This is where the handy built-in python functio

Len()

The len function takes in a string as an argument and returns the length of the string! We can us the string is 0 or empty!

```
print(len(""))      # will print 0
print(len("hello")) # will print 5
```

Now we can incorporate this function into our conditional! Let's write out the conditional in actu

```
@app.route('/process', methods=['POST'])
def process():
    if len(request.form['name']) < 1:
        # display validation errors
    else:
        # display success message
    return redirect('/') # either way the application should return to the index and di
```

Validation Step 2 -- Flash Messages on the Server

Flash messages are strings that exist for one redirect cycle. Similar to Session, you can access f python tags {{}} & {%%} on the views and display them to the user. The difference between flash only last for one redirect while session stays until it is manually popped. **This makes flash messa we only need to display the error or message temporarily!**

To use flash messages we first need to import them from Flask. Modify your import statement to

```
from flask import Flask, render_template, redirect, request, session, flash
```

Now using flash is as easy as invoking the flash function and passing in a string message! Let's fi statement and then we'll see how to display the messages on the client-side.

```
@app.route('/process', methods=['POST'])
def process():
    if len(request.form['name']) < 1:
        flash("Name cannot be empty!") # just pass a string to the flash function
    else:
        flash(f"Success! Your name is {request.form['name']}.") # just pass a string to
    return redirect('/') # either way the application should return to the index and di
```

Validation Step 3 -- Flash Messages on the Client

Flask helps us with flash messages by giving us access to a function on the client side that allow list. This function is "get_flashed_messages()" Let's see this in action.

BACK TO TRACKS

1. Flask Fundamentals 71%
(/m/72/5461)

CHECKLIST

```
<html>
<head>
  <title>Basic Validation Example</title>
</head>
<body>
  <h1>Enter a Valid(Any) Name!</h1>
  {% with messages = get_flashed_messages() %}
    {% if messages %}
      {% for message in messages %}
        <p>{{message}}</p>
      {% endfor %}
    {% endif %}
  {% endwith %}
  <form action='/process' method='POST'>
    Name:<input type="text" name="name">
    <input type="submit" value="Submit">
  </form>
</body>
</html>
```

A lot is happening in the code that we added. Let's go over it line by line

- *{% with messages = get_flashed_messages() %}*
  - The with messages here helps us declare a variable in our template that we can use v
- *{% if messages %}*
  - Check if there are even any messages that came back from the get_flashed_message
- *{% for message in messages %}*
  - Loop through all messages
- * {{message}} *
  - Print the messages one by one each in a paragraph tag

Note: In order to use sessions, you have to set a secret key. Refer to this documentation (http://flask.pocoo.org/docs/1.0/quickstart/#sessions) for more information.

Congratulations you have now learned basic validations for checking if an input is empty and errors as flash messages

Visual Demonstration of Form Validation

BACK TO TRACKS

PREVIOUS (/M/72/5461/35183)

NEXT (/M/72/54

1. Flask Fundamentals 71%
(/m/72/5461)

CHECKLIST