Flask_WTF and Quick Reference Cheat Sheet

by khalidhariry via cheatography.com/169470/cs/35550/

#Barebones App

```
# importing Flask
from flask import Flask, render _te mplate
# importing the flask_wtf library
from flask_wtf import FlaskForm
# importing the Fields
from wtforms import String Field, Passwo rdF ield,
Submit Field
# importing the valida tions
from wtform s.v ali dators import DataRe quired
```

# Getting Started

```
app = Flask(__name__)
app.co nfi g["S ECR ET_ KEY "] = " Thi sis ase -
cre tke y"
class LoginF orm (Fl ask Form):
        email = String Fie ld( lab el= “Em ail”, \
                        val ida tor s=[ Dat aRe -
qui red()])
        submit = Submit Fie ld( lab el= “lo gin”)
@app.r out e(“/“)
def login():
        log in_form = LoginF orm()
        log in_ for m.v ali dat e_o n_s ubmit()
        return render _te mpl ate (“l ogi n.h tml”,
form=l ogi n_form)
```

label

```
email = StringField(label=“Email”)
# the label property in use in login.html when the
form object is passed over.
<form method -"PO ST" action=“{{ url_fo r(' login')
}}">
{{ form. csrf_token }}
{{ form.e mai l.label }} {{ form.e mai l(s ize=30)
}}
{{ form.s ubmit }}
</f orm>
```

# Validation

# Error handling

```
# When a form is submitted, there may be a number
of errors, so a List of errors can be generated
and passed over to our form HTML as a property on
the field which generated the error, e.g.
form.< fie ld>.errors
{{ form.e mai l.label }} <br> {{ form.email
(size=30) }}
{% for err in form.e mai l.e rrors %}
<span style="color:red" >{{ err }}< /sp an>
{% endfor %}
# Form validation
# to validate the user's entry when they hit
submit. so we have to edit our route and make sure
it is able to respond to POST requests and then to
valida te_ on_ sub mit() e.g.
login_ for m.v ali dat e_o n_s ubmit()
```

#Template Inheritance Using Jinja2

```
#1. This line of code tells the templating engine
(Jinja) to use "base.html" as the template for this
page
{% extends " bas e.h tml " %}
#2.This block inserts a custom title into the
header of the template.
{% block title %}Succ ess{% endblock %}
#3.This block provides the content of the website.
The part that is going to vary between webpages.
{% block content %}
   <div class= " con tai ner ">
      <h1 >Top Secret </h 1>
      <iframe src="ht tps :// gip hy.c om /em -
bed /Ju 715 y9o syy mQ" width= " 480 " height -
="36 0" frameB ord er= " 0" class= " gip hy- emb -
ed" allowF ull Scr een ></ ifr ame>
      <p> <a href="h ttp s:/ /gi phy.co m/g -
ifs /ri ck- ast ley -Ju 715 y9o syy mQ">via
GIPHY< /a> </p>
   </ div>
{% endblock %}# label
email = String Fie ld( lab el= “Em ail”)
# the label property in use in login.html when the
form object is passed over.
<form method -"PO ST" action=“{{ url_fo r(' -
login') }}">
{{ form. csrf_token }}
{{ form.e mai l.label }} {{ form.e mai l(s ize=30)
}}
{{ form.s ubmit }}
</f orm>
```

```
email = StringField(Label='Email', validators=
[DataRequired()])
# The validators parameter accepts a List of
validator Objects. DataRe quired makes the field
required fields, so the user must type something,
otherwise an error will be generated.
# iIn order to make sure that we are giving all
users field valida tion, we have to switch off the
browser valida tion, and we do that with an
attribute on the form element called novali date.
<form method= " POS T" action="{{ url_fo r(' -
login') }}” novali dat e>
```