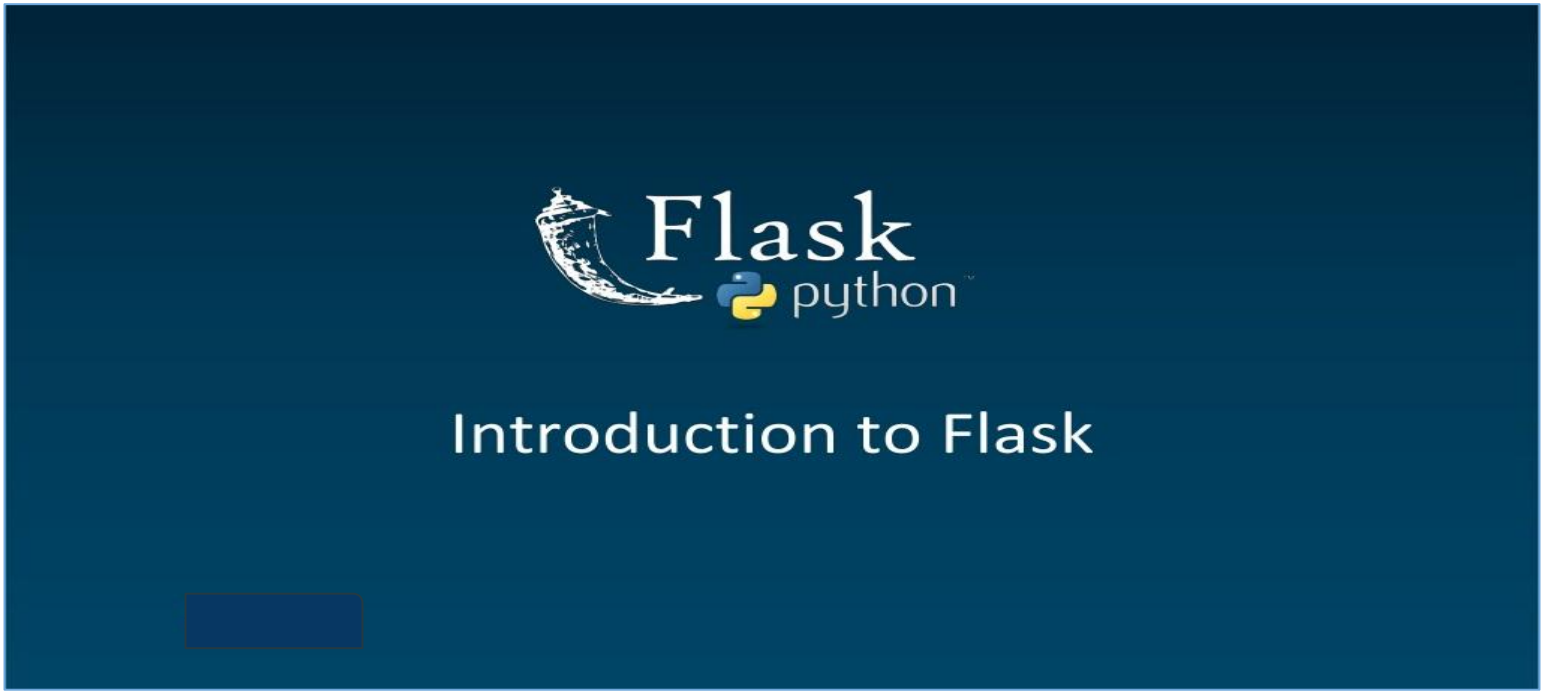
flask



What is Flask?

Flask is a web framework that provides libraries to build lightweight web applications in python. It is developed by **Armin Ronacher** who leads an international group of python enthusiasts (POCCO). It is based on WSGI toolkit and jinja2 template engine. Flask is considered as a micro framework.

What is Jinja2?

Jinja2 is a web template engine which combines a template with a certain data source to render the dynamic web pages.

What is WSGI?

It is an acronym for web server gateway interface which is a standard for python web application development. It is considered as the specification for the universal interface between the web server and web application.

First Flask application

In this section of the tutorial, we will build our first python website built using the Flask framework.

```
from flask import Flask
app = Flask(__name__) #creating the Flask class object
@app.route('/') #decorator defines the
def home():
    return "hello, this is our first flask website";
if __name__ =='__main__':
    app.run(debug = True)
```

Flask App routing

App routing is used to map the specific URL with the associated function that is intended to perform some task. It is used to access some particular page in the web application.

In our first application, the URL ('/') is associated with the home function that returns a particular string displayed on the web page.

```
from flask import Flask
app = Flask(__name__)
@app.route('/home/<name>')
def home(name):
return "hello,"+name;
if __name__ =="__main__":
app.run(debug = True)
```

URL Building

The url_for() function is used to build a URL to the specific function dynamically. The first argument is the name of the specified function, and then we can pass any number of keyword argument corresponding to the variable part of the URL.

This function is useful in the sense that we can avoid hard-coding the URLs into the templates by dynamically building them using this function.

```
from flask import *


app = Flask(__name__)
  @app.route('/admin')
def admin():
    return 'admin'
@app.route('/librarion')
def librarion():
    return 'librarion'
@app.route('/student')
def student():
    return 'student'
```

```
@app.route('/user/<name>')
def user(name):
    if name == 'admin':
        return redirect(url_for('admin'))
    if name == 'librarion':
        return redirect(url_for('librarion'))
    if name == 'student':
        return redirect(url_for('student'))
if __name__ =='__main__':
    app.run(debug = True)
```

Benefits of the Dynamic URL Building

1. It avoids hard coding of the URLs.
2. We can change the URLs dynamically instead of remembering the manually changed hard-coded URLs.
3. URL building handles the escaping of special characters and Unicode data transparently.
4. The generated paths are always absolute, avoiding unexpected behavior of relative paths in browsers.
5. If your application is placed outside the URL root, for example, in /myapplication instead of /, url_for() properly handles that for you.

Flask HTTP methods

HTTP is the hypertext transfer protocol which is considered as the foundation of the data transfer in the world wide web. All web frameworks including flask need to provide several HTTP methods for data communication.

1) GET

It is the most common method which can be used to send data in the unencrypted form to the server.

2) HEAD

It is similar to the GET but used without the response body.

3) POST

It is used to send the form data to the server. The server does not cache the data transmitted using the post method.

4) PUT

It is used to replace all the current representation of the target resource with the uploaded content.

5) DELETE

It is used to delete all the current representation of the target resource specified in the URL.

login.html

```
1.  <html>
2.    <body>
3.      <form action = "http://localhost:5000/login" method = "post">
4.        <table>
5.        <tr><td>Name</td>
6.        <td><input type ="text" name ="uname"></td></tr>
7.        <tr><td>Password</td>
8.        <td><input type ="password" name ="pass"></td></tr>
9.        <tr><td><input type = "submit"></td></tr>
10.     </table>
11.       </form>
12.   </body>
13. </html>
```

```
1.  from flask import *
2.  app = Flask(__name__)
3.
4.  @app.route('/login',methods = ['POST'])
5.  def login():
6.      uname=request.form['uname']
7.      passwrd=request.form['pass']
8.      if uname=="admin" and passwrd=="12345":
9.          return "Welcome %s" %uname
10.
11. if __name__ == '__main__':
12.     app.run(debug = True)
```

note:- 1st we just want to start(shift+F10) a flask file and then run a html page

Rendering external HTML files

Flask facilitates us to render the external HTML file instead of hardcoding the HTML in the view function. Here, we can take advantage of the jinja2 template engine on which the flask is based.

Flask provides us the render_template() function which can be used to render the external HTML file to be returned as the response from the view function.

templates/message.html

```
1.  <html>
2.  <head>
3.  <title>Message</title>
4.  </head>
5.  <body>
6.  <h1>hi, welcome to the website
    </h1>
7.  </body>
8.  </html>
```

script.py

```
1.  from flask import *
2.  app = Flask(__name__)
3.
4.  @app.route('/')
5.  def message():
6.      return
    render_template('message.html')
7.  if __name__ == '__main__':
8.    app.run(debug = True)
```

1st we want to create one **templates** inside the application directory and save the HTML templates referenced in the flask script in that directory.

script.py

```
from flask import *
app = Flask(__name__)


@app.route('/user/<uname>')
def message(uname):
    return
render_template('message.html',na
me=uname)
if __name__ == '__main__':
  app.run(debug = True)
```

templates/message.html

```
1. <html>
2. <head>
3. <title>Message</title>
4. </head>
5. <body>
6. <h1>hi, {{ name }}</h1>
7. </body>
8. </html>
```

script.py

```
from flask import *
app = Flask(__name__)

@app.route('/table/<int:num>')
def table(num):
    return render_template('print-table.html',n=num)
if __name__ == '__main__':
  app.run(debug = True)
```

print-table.py

```
<html>
<head>
<title>print table</title>
</head>
<body>
<h2> printing table of {{n}}</h2>
{% for i  in range(1,11): %}
   <h3>{{n}} X {{i}} = {{n * i}} </h3>
{% endfor %}
</body>
</html>
```

customer.html

```
<html>
   <body>
      <h3>Register the customer, fill the following form.</h3>
      <form action = "http://localhost:5000/success" method = "POST">
         <p>Name <input type = "text" name = "name" /></p>
         <p>Email <input type = "email" name = "email" /></p>
         <p>Contact <input type = "text" name = "contact" /></p>
         <p>Pin code <input type ="text" name = "pin" /></p>
         <p><input type = "submit" value = "submit" /></p>
      </form>
   </body>
</html>
```

result_data.html

```
<!doctype html>
<html>
   <body>
      <p><strong>Thanks for the registration.
Confirm your details</strong></p>
      <table border = 1>
         {% for key, value in result.items() %}
            <tr>
               <th> {{ key }} </th>
               <td> {{ value }} </td>
            </tr>
         {% endfor %}
      </table>
   </body>
</html>
```

```
from flask import *
app = Flask(__name__)
@app.route('/')
def customer():
   return render_template('customer.html')

@app.route('/success',methods = ['POST', 'GET'])
def print_data():
   if request.method == 'POST':
      result = request.form
      return render_template("result_data.html",result =
result)


if __name__ == '__main__':
   app.run(debug = True)
```

login.py

```
from flask import *
app = Flask(__name__)
@app.route('/error')
def error():
    return "<p><strong>Enter correct password</strong></p>"
@app.route('/')
def login():
    return render_template("login.html")
@app.route('/success',methods = ['POST'])
def success():
    if request.method == "POST":
        email = request.form['email']
        password = request.form['pass']
    if password=="jtp":
        resp = make_response(render_template('success.html'))
        resp.set_cookie('email',email)
        return resp
```

login.html

```
<html>
<head>
   <title>login</title>
</head>
<body>
   <form method = "post" action = "http://localhost:5000/success">
      <table>
         <tr><td>Email</td><td><input type = 'email' name = 'email'></td></tr>
         <tr><td>Password</td><td><input type = 'password' name = 'pass'></td></tr>
         <tr><td><input type = "submit" value = "Submit"></td></tr>
      </table>
   </form>
</body>
</html>
```

success.html

```
<html>
<head>
<title>success</title>
</head>
<body>
    <h2>Login successful</h2>
    <a href="/viewprofile">View Profile</a>
</body>
</html>
```

profile.html

```
<html>
<head>
    <title>profile</title>
</head>
<body>
    <h3>Hi, {{name}}</h3>
</body>
</html>
```