PHINMA EDUCATION
MAKING LIVES BETTER THROUGH EDUCATION

ITE 306: Integrative Programming and Technologies
Module #19

Name: _____ Class number: _____
Section: _____ Schedule: _____ Date: _____

Lesson title: Building a webpage using Python
Lesson Objectives:
At the end of this module, I should be able to:
1.) Create a simple flask application
2.) Know what is Flask

Materials:
Activity Sheets, smart phone or computer, internet connection
References:
- https://www.geeksforgeeks.org/flask-creating-first-simple-application/?ref=rp
- https://www.tutorialspoint.com/flask/flask_templates.htm

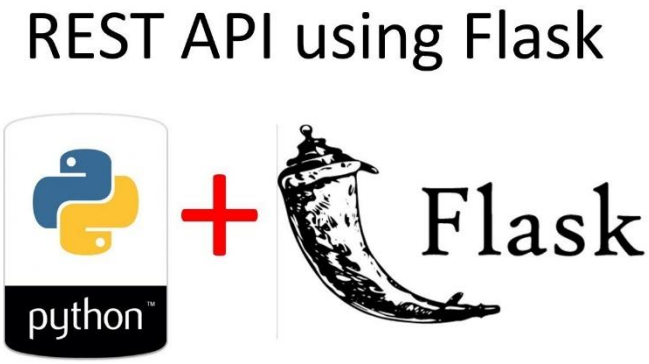
Productivity Tip:
"Study breaks increase productivity. You can't work at high intensity forever: you'll need to take a break to refresh and recharge."

A. LESSON PREVIEW/REVIEW

1) Introduction

There are many modules or frameworks which allows to build your webpage using python like bottle, Django, flask etc. But the real popular ones are Flask and Django. Django is easy to use as compared to Flask but Flask provides you the versatility to program with.

To understand what Flask is you have to understand few general terms.



1. **WSGI** - Web Server Gateway Interface (WSGI) has been adopted as a standard for Python web application development. WSGI is a specification for a universal interface between the web server and the web applications.

2. **Werkzeug** It is a WSGI toolkit, which implements requests, response objects, and other utility functions. This enables building a web framework on top of it. The Flask framework uses Werkzeug as one of its bases.

3. **jinja2** jinja2 is a popular templating engine for Python. A web templating system combines a template with a certain data source to render dynamic web pages.

Name: _____ Class number: _____
Section: _____ Schedule: _____ Date: _____

2) Activity 1: What I Know Chart, part 1

Try answering the questions below by writing your ideas under the first column *What I Know*. It's okay if you write key words or phrases that you think are related to the questions. Leave the third column blank for Activity 4.

What I Know	Questions:	What I Learned (Activity 4)
	1. What is Web Framework?	
	2.What are the Web Frameworks that you know?	

B.MAIN LESSON

1) Activity 2: Content Notes

What is Web Framework?

Web Application Framework or simply Web Framework represents a collection of libraries and modules that enables a web application developer to write applications without having to bother about low-level details such as protocols, thread management etc.

What is Flask?

Flask is a web application framework written in Python. It is developed by **Armin Ronacher**, who leads an international group of Python enthusiasts named Pocco. Flask is based on the Werkzeug WSGI toolkit and Jinja2 template engine. Both are Pocco projects.

WSGI

Web Server Gateway Interface (WSGI) has been adopted as a standard for Python web application development. WSGI is a specification for a universal interface between the web server and the web applications.

Werkzeug

It is a WSGI toolkit, which implements requests, response objects, and other utility functions. This enables building a web framework on top of it. The Flask framework uses Werkzeug as one of its bases.

Name: _____ Class number: _____
Section: _____ Schedule: _____ Date: _____

Jinja2

Jinja2 is a popular templating engine for Python. A web templating system combines a template with a certain data source to render dynamic web pages.

Flask is often referred to as a micro framework. It aims to keep the core of an application simple yet extensible. Flask does not have built-in abstraction layer for database handling, nor does it have form a validation support. Instead, Flask supports the extensions to add such functionality to the application. Some of the popular Flask extensions are discussed later in the tutorial.

Flask – Application

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

@app.route('/')
def hello_world():
   return 'Hello World'

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

Importing flask module in the project is mandatory. An object of Flask class is our **WSGI** application.

Flask constructor takes the name of **current module (__name__)** as argument.

The **route()** function of the Flask class is a decorator, which tells the application which URL should call the associated function.

`app.route(rule, options)`

- The **rule** parameter represents URL binding with the function.
- The **options** is a list of parameters to be forwarded to the underlying Rule object.

In the above example, **'/'** URL is bound with **hello_world()** function. Hence, when the home page of web server is opened in browser, the output of this function will be rendered.
Finally the **run()** method of Flask class runs the application on the local development server.
`app.run(host, port, debug, options)`

All parameters are optional

	Parameters & Description
1	**host** Hostname to listen on. Defaults to 127.0.0.1 (localhost). Set to '0.0.0.0' to have server available externally
2	**port** Defaults to 5000

Name: _____ Class number: _____
Section: _____ Schedule: _____ Date: _____

3	**debug** Defaults to false. If set to true, provides a debug information
4	**options** To be forwarded to underlying Werkzeug server.

The above given **Python** script is executed from Python shell.

```
Python Hello.py
```

A message in Python shell informs you that

```
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

Open the above URL **(localhost:5000)** in the browser. **'Hello World'** message will be displayed on it.

Debug mode

A **Flask** application is started by calling the **run()** method. However, while the application is under development, it should be restarted manually for each change in the code. To avoid this inconvenience, enable **debug support**. The server will then reload itself if the code changes. It will also provide a useful debugger to track the errors if any, in the application.

The **Debug** mode is enabled by setting the **debug** property of the **application** object to **True** before running or passing the debug parameter to the **run()** method.

```
app.debug = True
app.run()
app.run(debug = True)
```

Flask – Routing

Modern web frameworks use the routing technique to help a user remember application URLs. It is useful to access the desired page directly without having to navigate from the home page.
The **route()** decorator in Flask is used to bind URL to a function.

For example:

```
@app.route('/hello')
def hello_world():
   return 'hello world'
```

Here, URL **'/hello'** rule is bound to the **hello_world()** function. As a result, if a user visits **http://localhost:5000/hello** URL, the output of the **hello_world()** function will be rendered in the browser.

The **add_url_rule()** function of an application object is also available to bind a URL with a function as in the above example, **route()** is used.

Name: _____ Class number: _____
Section: _____ Schedule: _____ Date: _____

A decorator's purpose is also served by the following representation:

```
def hello_world():
   return 'hello world'
app.add_url_rule('/', 'hello', hello_world)
```

Flask – HTTP methods

Http protocol is the foundation of data communication in world wide web. Different methods of data retrieval from specified URL are defined in this protocol.

The following table summarizes different http methods:

	Methods & Description
1	**GET** Sends data in unencrypted form to the server. Most common method.
2	**HEAD** Same as GET, but without response body
3	**POST** Used to send HTML form data to server. Data received by POST method is not cached by server.
4	**PUT** Replaces all current representations of the target resource with the uploaded content.
5	**DELETE** Removes all current representations of the target resource given by a URL

By default, the Flask route responds to the **GET** requests. However, this preference can be altered by providing methods argument to **route()** decorator.

2) Activity 3: Skill-building Activities

Let's practice! After completing each exercise, you may refer to the *Key to Corrections* for feedback. Try to complete each exercise before looking at the feedback.

Coding Time: Following the following steps in creating flask application.

1.) Create a web application that will say Hello and print the user name.

STEP 1:	**Installation:** We will require two package to setup your environment. *virtualenv* for a user to create multiple Python environments side-by-side. Thereby, it can avoid compatibility issues between the different versions of the libraries and the next will be *Flask* itself. 1. **virtualenv** pip install virtualenv 2. **Flask** pip install Flask

Name: _____ Class number: _____
Section: _____ Schedule: _____ Date: _____

	After completing the installation of the package, let's get our hands on the code.
	`from flask import Flask`
	`app = Flask(__name__)`
	
	`@app.route('/')`
	`def hello_world():`
	`  return 'Hello World'`
	`if __name__ == '__main__':`
	`  app.run()`

STEP 2:	Save it with a file name ***myflask1.py*** and then run the script we will be getting an output like this.
	Python 3.5.0 Shell
	File Edit Shell Debug Options Window Help
	`Python 3.5.0 (v3.5.0:374f501f4567, Sep 13 2015, 02:27:37) [MSC v.1900 64 bit (AMD64)] on win32`
	`Type "copyright", "credits" or "license()" for more information.`
	`>>>`
	` RESTART: C:\Users\knapseck\Desktop\GFG Internship\23. Introduction to Flask\1.py`
	` * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)`

STEP 3:	Then go to the url given there you will seeing your first webpage displaying hello world there on your local server. Go to the url *http://127.0.0.1:5000/*

STEP 4:	Digging further into the context, the **route()** decorator in Flask is used to bind URL to a function. Now to extend this functionality our small web app is also equipped with another method **add_url_rule()** which is a function of an application object is also available to bind a URL with a function as in the above example, route() is used.
	
	Modify ***myflask1.py*** and add the following lines of code then save it. Don't forget to run again the script.
	
	`from flask import Flask`
	`app = Flask(__name__)`
	
	`@app.route('/hello/<name>')`
	`def hello_name(name):`
	`  return 'Hello %s!' % name`
	
	`if __name__ == '__main__':`
	`  app.run()`

STEP 5:	Go to the url http://127.0.0.1:5000/hello/<name> Replace *<name>* with your name.

Name: _____ Class number: _____
Section: _____ Schedule: _____ Date: _____

	Example: http://127.0.0.1:5000/hello/Juan
STEP 6:	Save your file for checking.

2.) Create an HTML form and use the POST method to send form data to a URL.

Follow this folder directory:

- **pythontest** *(folder)*
  - login.html *(html file)*
  - **login** *(folder)*
    - myflask2.py *(python file)*

STEP 1:

Now let's create a html login page. Save this new file as ***login.html*** inside the *pythontest* folder.

```
<html>
    <body>
        <form action = "http://localhost:5000/login" method = "post">
             <p>Enter Name:</p>
             <p><input type = "text" name = "nm" /></p>
             <p><input type = "submit" value = "submit" /></p>
        </form>
    </body>
</html>
```

STEP 2:

Create a new python script and save it as ***myflask2.py*** inside the login folder, this python script will create the server.

```
from flask import Flask, redirect, url_for, request
app = Flask(__name__)

@app.route('/success/<name>')
def success(name):
       return 'welcome %s' % name

@app.route('/login',methods = ['POST', 'GET'])
def login():
       if request.method == 'POST':
                user = request.form['nm']
```

Name: _____ Class number: _____
Section: _____ Schedule: _____ Date: _____

	`return redirect(url_for('success',name = user))`
	`else:`
	`user = request.args.get('nm')`
	`return redirect(url_for('success',name = user))`
	
	`if __name__ == '__main__':`
	`app.run(debug = True)`

STEP 3:	Go to your cmd and run the ***myflask2.py***. After the development server starts running, open login.html in the browser. Enter name in the text field and click *submit* button. Please notice the URL. Enter Name: subhajit submit **Output:** "welcome subhajit".
STEP 4:	Save your file for checking.

3) Activity 4: What I Know Chart, part 2

It's time to answer the questions in the *What I Know* chart in Activity 1. Log in your answers in the third column of the table in Activity 1.

4) Activity 5: Check for Understanding

A. Multiple Choices. Encircle the letter of the correct answer.

1.) It is a web application framework written in Python.
   a. Flask
   b. WSGI
   c. Werkzeug
   d. Jinja2

2.) It is a popular templating engine for Python.
   a. Flask
   b. WSGI
   c. Werkzeug
   d. Jinja2

3.) It represents a collection of libraries and modules that enables a web application developer to write applications without having to bother about low-level details such as protocols, thread management etc.
   a. Web Framework
   b. Werkzeug
   c. route()
   d. get()

4.) It is a standard for Python web application development. It is a specification for a universal interface between the web server and the web applications.
   a. Flask
   b. WSGI
   c. Werkzeug
   d. Jinja2

Name: _____ Class number: _____
Section: _____ Schedule: _____ Date: _____

5.) This fuction tells the application which URL should call the associated function.
   a. Web Framework
   b. Werkzeug
   c. route()
   d. get()

B. TRUE / FALSE. Write true if the statement is correct, otherwise write the correct answer.

_____1. By default, the Flask route responds to the POST requests.
_____2. Web framework is the foundation of data communication in world wide web.
_____3. The PUT method removes all current representations of the target resource given by a URL.
_____4. A Flask application is started by calling the run() method.
_____5. An object of Flask class is our WSGI application.

C. LESSON WRAP-UP

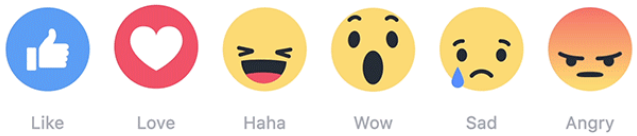
Activity 6: Thinking about Learning

Congratulations for finishing this module! **Shade** the number of the module that you finished to track how much work you have accomplished and how much work there is left to do.

You are done with the session! Let's track your progress.

Period 1											Period 2											Period 3								
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31

Rate the session for today by encircling the emoji that best captures your experience and write your reason for choosing that emoji.



Reason/s: _____

FAQs

1. **What can you do with flask?**
   Answer: *Flask is a web framework. This means flask provides you with tools, libraries and technologies that allow you to build a web application. This web application can be some web pages, a blog, a wiki or go as big as a web-based calendar application or a commercial website.*

2. **Which is better Django or flask?**
   Answer: *Both Django and Flask are hugely popular among Python programmers. Django is a full-stack web framework for Python, whereas Flask is a lightweight and extensible Python web framework. Likewise, Jinja2 template engine makes it easier for programmers to build dynamic web applications.*

Name: _____ Class number: _____
Section: _____ Schedule: _____ Date: _____

KEY TO CORRECTIONS

Answers to Checking for Understanding:

A. Multiple Choices. Encircle the letter of the correct answer.

1.) It is a web application framework written in Python.
a. Flask
b. WSGI
c. Werkzeug
d. Jinja2

2.) It is a popular templating engine for Python.
a. Flask
b. WSGI
c. Werkzeug
d. Jinja2

3.) It represents a collection of libraries and modules that enables a web application developer to write applications without having to bother about low-level details such as protocols, thread management etc.
a. Web Framework
b. Werkzeug
c. route()
d. get()

4.) It is a standard for Python web application development. It is a specification for a universal interface between the web server and the web applications.
a. Flask
b. WSGI
c. Werkzeug
d. Jinja2

5.) This function tells the application which URL should call the associated function.
a. Web Framework
b. Werkzeug
c. route()
d. get()

B. TRUE / FALSE. Write true if the statement is correct, otherwise write the correct answer.

_____**1.** By default, the Flask route responds to the POST requests. ***GET***
_____**2.** Web framework is the foundation of data communication in world wide web. ***Http protocol***
_____**3.** The PUT method removes all current representations of the target resource given by a URL. *Delete*
_____**4.** A Flask application is started by calling the run() method. *True*
_____**5.** An object of Flask class is our WSGI application. *True*