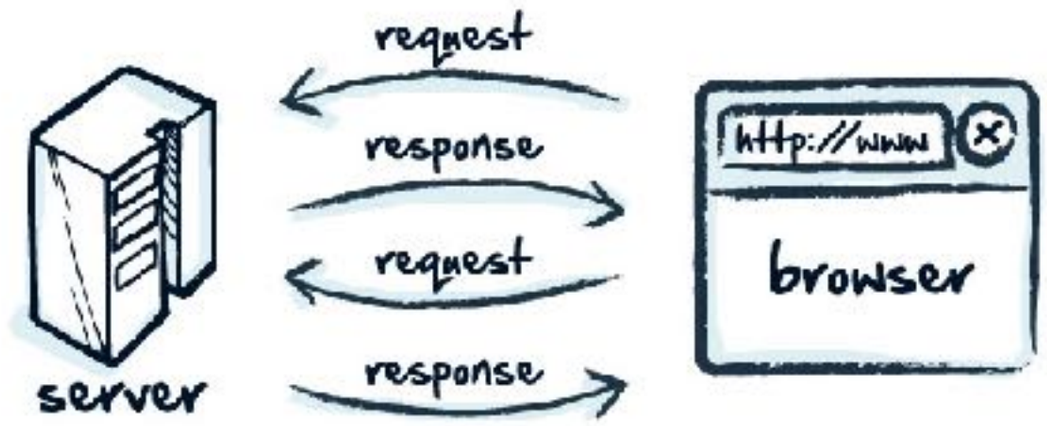
Web Applications with Python

CST 205

How does the web work?

The web is all about requests and responses



Web request — As a result of some interaction, a request is sent from a web browser (client) to a web server.

Web response — A reply from the web server sent to the web browser.

Types of responses

- Web request could be for **static content**, such as an HTML file, an image, or something else stored on the server.

- **Dynamic content** is generated by some code run on the server.

Web application frameworks

- We will be using the **Flask** framework.

- Flask provides a collection of modules to help you build server-side web applications.

- To install (with your virtual environment activated):

```
pip install flask
```

More about Flask

- We just installed Flask and four other modules: **Werkzeug**, **MarkupSafe**, **Jinja2**, and **itsdangerous**

  - (More on these modules on slide 8)

- As of February 28, 2018, the current stable version of Flask is 0.12.2.

A few definitions:

- **module** — a file which can expose classes, functions, and global variables

- **package** — a directory of Python modules.

`WSGI`

- A *traditional* web server does not understand Python.

- The Python community came up with a standard known as the **Web Server Gateway Interface** (`WSGI`).

- For more on **`WSGI`**, refer to PEP 333

Helper modules

- `Werkzeug` — a WSGI library for Python (more info)
- `MarkupSafe` — a library for Python that implements HTML-friendly strings. Uses by the `Jinja2` template engine. (more info)
- `Jinja2` — a templating language for Python. Renders HTML templates and allows for logic in HTML. (more info)
- `itsdangerous` — facilitates secret key exchange for Flask sessions. (more info)

Hello World with Flask

```
from flask import Flask

# create an instance of the Flask class
app = Flask(__name__)

# route() decorator binds a function to a URL
@app.route('/hello')
def hello():
    return 'Hello world from Flask'
```

How to run a Flask application

From the terminal:

```
export FLASK_APP=hello_flask.py
export FLASK_DEBUG=1
flask run
```


```
$env:FLASK_APP = "hello_flask.py"
$env:FLASK_DEBUG = "1"
flask run
```

Flask application object

- The line of code `app = Flask(__name__)` instantiates a Flask application object called `app`.

  - The `__name__` value is maintained by the Python interpreter and, when used anywhere within your program's code, is set to the name of the currently active module. Flask needs this information.

  - The `__name__` value for our current application is `hello_flask`

Function decorator, @

- Augments the function with additional behavior.
- In this case, the **`route`** decorator is provided by Flask.
    - The route decorator arranges for the Flask web server to call the function when a request for the, for example, **`"/hello"`** URL arrives at the server.
- Right now, our “Hello World” example does not have a route for the default `"/"` route

Debugging

- We can add `print()` functions to view functions to help with debugging.
    - The result will show up in the terminal
- Our terminal will also show status codes and other information.
    - `"GET / HTTP/1.1" 200 -` means a successful GET request
    - `"GET /test HTTP/1.1" 404 -` means the requested URL was not found.
    - Many more status codes, info in RFC 7231

Add route for home page

```
@app.route('/')
def home():
    my_string = "<h1>Welcome to my page</h1><p>Have a  nice day!</p>"
    return my_string
```

We don’t need to restart our app for our new page to work since we are using debug mode.

Template Engine

- Templates make HTML much easier to maintain and add more power and expressiveness to your HTML code.

- As mentioned, the template engine shipped with Flask is called **Jinja2**.

Make Flask work with Jinja2

- We will put our HTML files in a special folder called `templates`.

- We also need to add `render_template` to our import statement.

- Lastly, instead of returning a string, we pass `render_template` the template.

Code sample

```
from flask import Flask, render_template

app = Flask(__name__)

@app.route('/mytemplate')
def t_test():
    return render_template('template.html')
```

Flask-Bootstrap

- Bootstrap is an open-source, responsive, mobile-first front-end component library

- Flask-Bootstrap documentation

- `pip install flask-bootstrap`

- `bootstrap/base.html` template becomes available