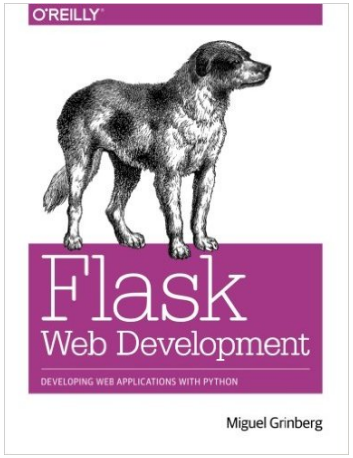
Flask I

Harry J. Wang, Ph.D.

University of Delaware

Resources

- Optional Book: Flask Web Development: Developing Web Applications with Python by Miguel Grinberg (1st Edition)
- Free online tutorials: https://blog.miguelgrinberg.com/post/the-flask-mega-tutorial-part-i-hello-world



- Sample Course Application: SongBase: https://github.com/udmis/songbase

Tutorial at https://github.com/udmis/web-app-dev/wiki/Set-up-Python-2.7.x

Virtual Environment

The full path to this folder cannot have spaces and special characters!!!

- We need different virtual environments for different projects due to different Python version, different packages, same packages with different versions.
- To setup the virtual environment with packages:
  - In the project folder, create a **requirements.txt** file, which includes all packages for the project if any
  - In the terminal, go to the project folder and create the virtual environment:

```
$ virtualenv venv
```

  - The activate the virtual environment:

```
$ source venv/bin/activate
```

  - You should see the name of the virtual environment in front of the prompt:

```
(venv) dami:exercise harrywang$
```

  - Install the required packages:

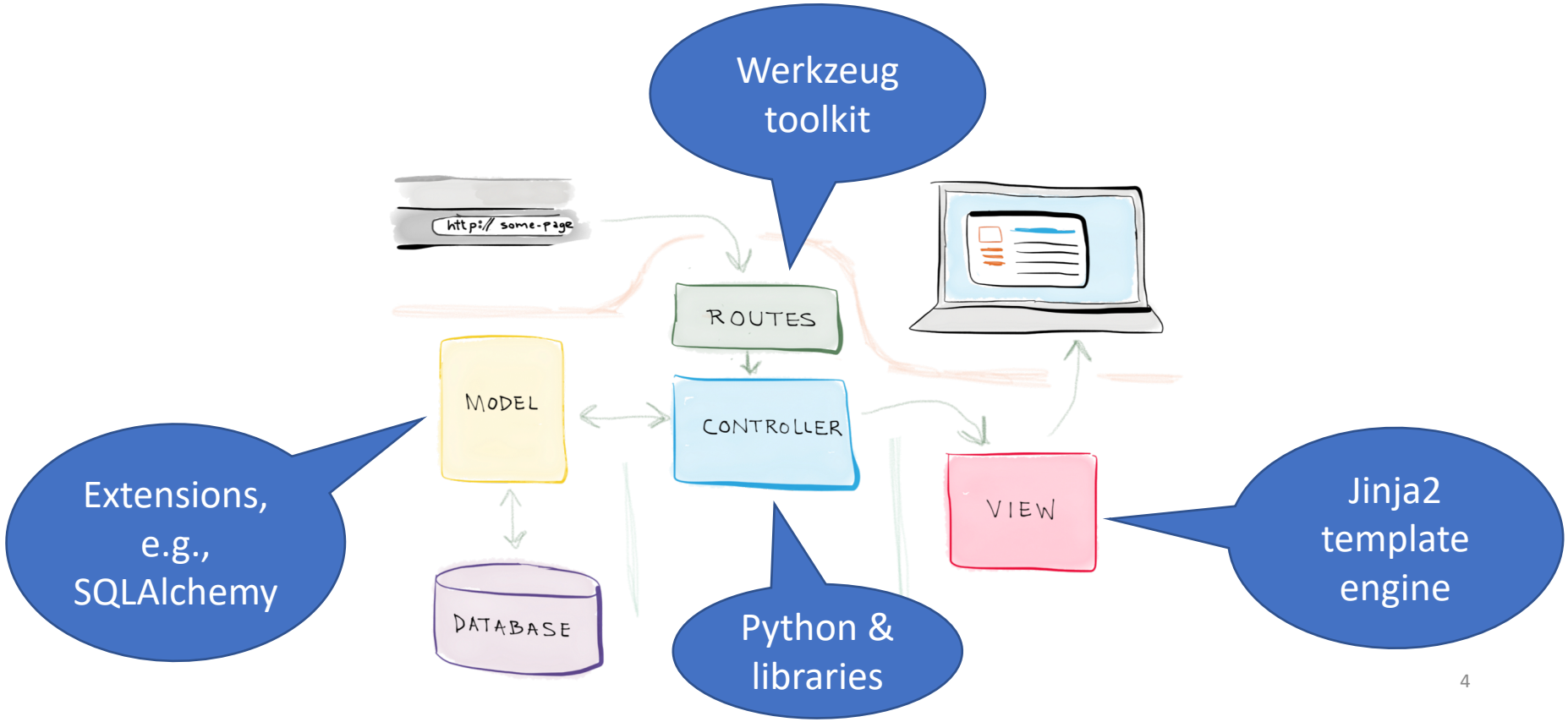
```
$ pip install -r requirements.txt
```

  - Deactivate:

```
$ deactivate
```

What is Flask?

- Flask is a micro web framework written in Python that supports MVC.
- Flask has a built-in development server and debugger.
- Flask is highly extensible via various extensions.



See README: https://github.com/udmis/songbase

Setup and Run a Simple Flask App

1. Setup virtual environment and activate
2. Install required packages using pip
3. Setup the basic app and routing (e.g., in songbase.py)

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


@app.route('/')
def index():
    # return HTML
    return "<h1>this is the index page!<h1>"
```

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

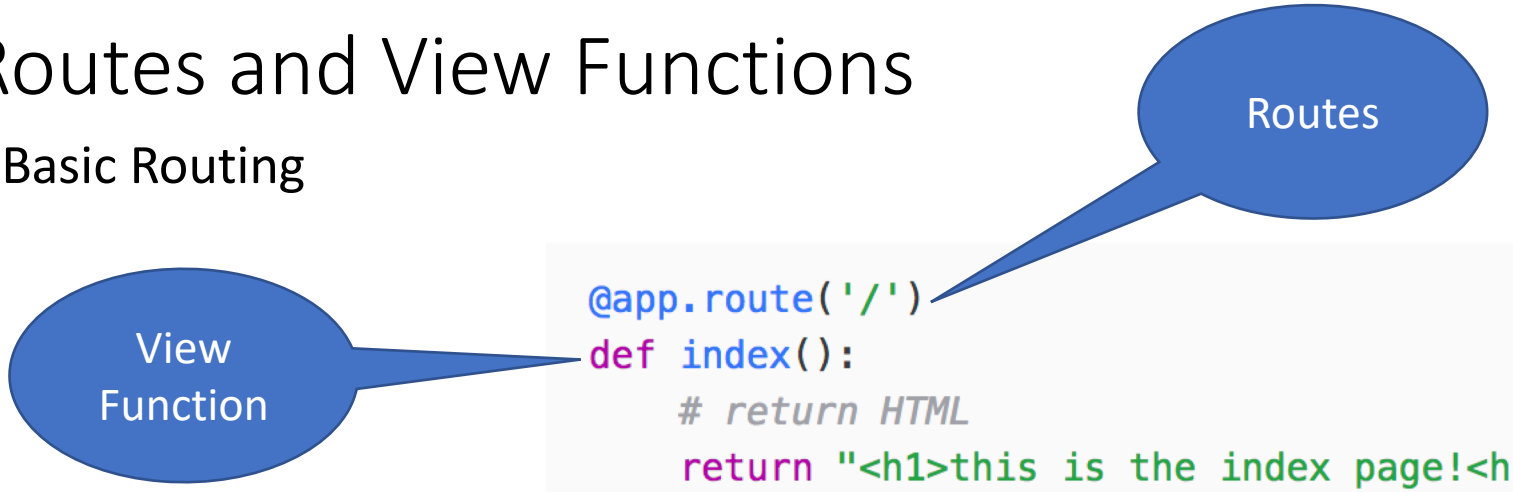
What this if does: https://goo.gl/Pc39w8

4. Start up the server

```
(venv) wifi-roaming-128-4-186-49:songbase harrywang$ python songbase.py
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

Routes and View Functions

- Basic Routing



- Routing with variables (with strong formatting and single/double quotes example)

```
@app.route('/users/<string:name>/')
def get_user_name(name):
    # return "hello " + name
    return "Hello %s, this is %s" % (name, 'administrator')


@app.route('/songs/<int:id>/')
def get_song_id(id):
    # return "This song's ID is " + str(id)
    return "Hi, this is %s and the song's id is %d" % ('administrator', id)
```

In-class exercise

- Setup basic Flask server
- Create a few routes

Flask Extensions

- Flask-Script extension provides support for writing external scripts in Flask, such as running a development server with debugger, a customised Python shell, scripts to set up your database, etc.
- For example, activating debugger and reloader, or running dev server with different port require changing the source code:

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

```
if __name__ == '__main__':
    app.run(host='0.0.0.0', port=80)
```

- With Flask-Script, create manage.py:

```
from flask_script import Manager
from songbase import app

manager = Manager(app)
```

```
if __name__ == "__main__":
    manager.run()
```

- Then, you can

```
(venv) wifi-roaming-128-4-186-49:songbase harrywang$ python manage.py runserver -d
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
```

Jinja2

- Jinja2 is a modern and designer-friendly templating language for Python.
- Separate business logic from presentation using render_template()
- /templates folder for HTML template files

```
from flask import Flask, render_template
```

```
@app.route('/')
def index():
    # return HTML
    # return "<h1>this is the index page!<h1>"
    return render_template('index.html')
```

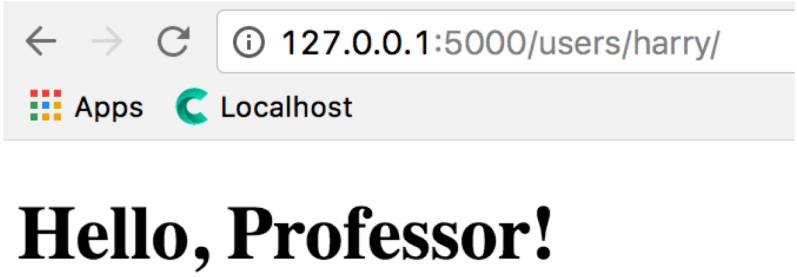
```
@app.route('/users/<string:name>/')
def get_user_name(name):
    # return "hello " + name
    # return "Hello %s, this is %s" % (name, 'administrator')
    return render_template('user.html', name=name)
```

Template Syntax

- `{% ... %}` for Statements
- `{{ ... }}` for Expressions to print to the template output
- `{# ... #}` for Comments not included in the template output

- If Statement

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>User Page</title>
  </head>
  <body>
    {# This is comment in template #}
    {% if name=="harry" %}
          <h1>Hello, Professor!</h1>
    {% else %}
          <h1>Hello, {{ name }}!</h1>
    {% endif %}
  </body>
</html>
```

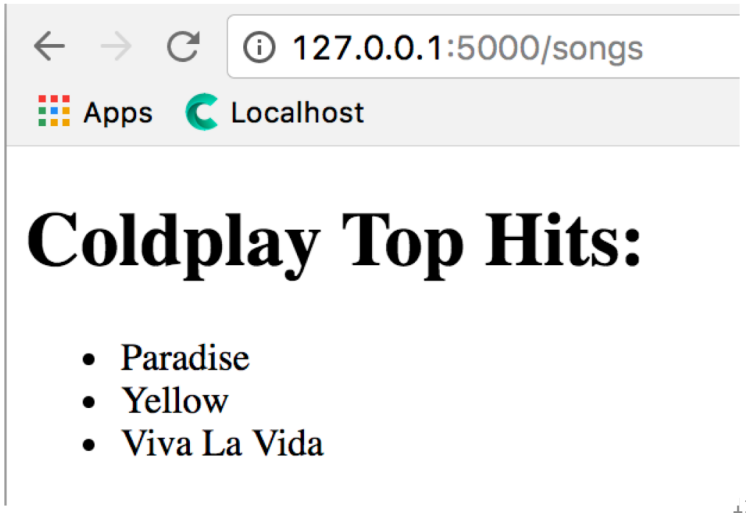


Template Syntax (cont.)

- For Loop

```python
@app.route('/songs')
def show_all_songs():
    songs = [
        'Paradise',
        'Yellow',
        'Viva La Vida'
    ]
    return render_template('song-all.html', songs=songs)
```

```
<body>
  <h1>Coldplay Top Hits:</h1>
  {# This is a loop #}
  <ul>
    {% for song in songs %}
        <li>{{song}}</li>
    {% endfor %}
  </ul>
</body>
```

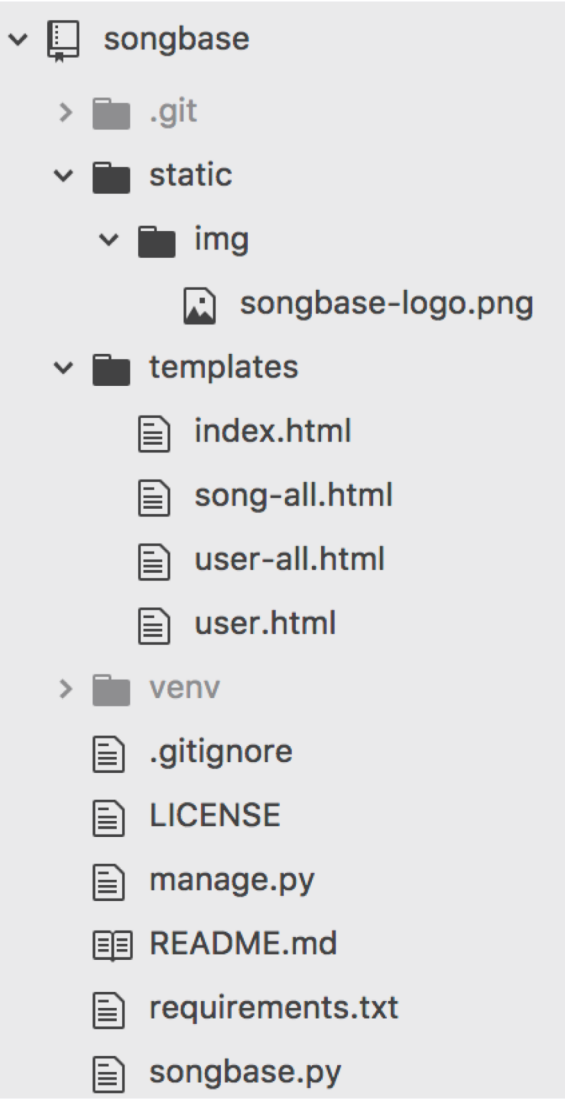


Static Files

- /static folder for static files, such as css, js, images, etc.
- url_for() function is used to refer to the static files

```
<body>
  <h1>All Users</h1>
  <img src="{{ url_for('static', filename='img/songbase-logo.png') }}"
   alt="SongBase">
</body>
```

Project Folder Structure



See code at SongBase:
https://github.com/udmis/songbase

Use `git checkout v1` to see the code