# CS50 Section. Week 9. 11/1/16.

*Tuesdays 4-5:30 PM, CGIS S-040*

Nicholas Boucher nboucher@college.harvard.edu

# Important links

- CS50 Discuss: https://cs50.harvard.edu/discuss
- CS50 Style Guide: https://manual.cs50.net/style/

# Section Agenda

1. Python Decorators
2. Flask
3. Jinja
4. SQL

# Python Decorators

Python, like a handful of other modern languages, has a construct known as decorators. A *decorator* is a function that modifies the behavior of other functions, typically applying some extra functionality thereto. Under the hood, decorators pass the function they are applied to as an argument to another function which then wraps it with more functionality.

Examples of decorators can be seen throughout pset7 with code such as the following:

```
@app.route("/buy", methods=["GET", "POST"])
@login_required
def buy():
    """Buy shares of stock."""
    return apology("TODO")
```

The two main decorators to know are: * `@app.route()` * Specifies how a webpage can be reached, both with its path relative to the domain and its protocol * The two protocol (`method=[]`) options are GET and POST. GET is typically used for simply requesting a URL while POST is usually used for transmitting some data to the server, such as a form. * `@login_required` * Specifies that a user must be logged-in for the page to load.

# Flask

Flask is a Python-based web microframework that automates the process of building simple web apps. By microframework, we means a small set of code written by someone else and imported into your application to handle common tasks automatically - in this case, serving the website you are writing.

As is the purpose of frameworks in general, you do not need to know *how* flask does what it does, just how to use it. The following is an example of a very basic usage of Flask:

```
from flask import Flask

app = Flask(__name__)

@app.route("/")
def index():
    return "You are at the index!"
```

To run an application written with flask, you need only type `flask run` from within the directory containing the target `application.py` (which is the file containing code like that above). Then click on **CS50 IDE** in the top left corner of the IDE, followed by **Web Server**. When you make changes to your code, you do not need to restart the web server in order for the changes to take effect! This is thanks to the fact that Python is an *interpreted* language instead of a *compiled* language.

# Jinja

Jinja is aa Python-inspired templating language that allows one to generate complex webpages via templates rendered with programmatic logic. Essentially, Jinja is the difference between Flask generating strings that are displayed in a user's browser and generating full webpages.

Examples of Jinja can bee seen in the `/templates` folder of the ps7 distro code. Here, the base template is `layout.html` and each other page (i.e. `login.html` and `apology.html`) extends the base template.

When a Jinja page is rendered via a call to `render_template()` (which is conveniently wrapped for you in the `apology` function), the logical blocks contained within `{% %}` are evaluated and rendered HTML is returned.

# SQL

SQL stands for *Structured Query Language*. It is a very simple language that is used to interact with databases.

Within pset 7, we have provided an implementation of *SQLlite* for you. This stores your databases

in `.db` files within your application's working directory (but this is already setup for you).

There are 4 basic commands that you will need to know in SQL:

1. `UPDATE` - Update data in a database table

```
# update table, changing values in particular columns
UPDATE table SET col1 = val1, col2 = val2, ...

# update table, changing col1 to val1 where "name" equals "identifier"
UPDATE table SET col1 = val1 WHERE identifier = "name"
```

2. `INSERT INTO` - Insert certain values into a table

```
# insert into table the row of values
INSERT INTO table VALUES values

# insert into table under columns col1 & col2, val1 & val2
INSERT INTO table (col1, col2) VALUES (val1, val2)
```

3. `SELECT` - Select values to view

```
# select a column from table to compare/ view
SELECT col FROM table WHERE col = "identifier"

SELECT * FROM table # select all columns from a table
```

4. `DELETE` - Delete from table

```
# delete a row from table
DELETE FROM table WHERE col = "identifier"
```

In Python, SQL statements can be executed via `db.execute()` within your code. Thus, you have the power to work with databases in your code!

Finally, you can access the visual "Database Manager" (phpMyAdmin) via the `phpliteadmin DATABASE_FILE` from the command line, and then selecting **CS50 IDE** in the top left corner of the IDE, followed by **phpMyAdmin**.