* Create the web framework (flask)
* Set up page to display text and boxes for input fields (HTML)
* Set up page to take in inputs (flask)
* Set up page to return a result (pickle)
  + Pickle module in python is useful for pushing python content to the web app
  + Chart.js for graphs
    - Looks like you can also use straight up json/html to create at least a table (maybe not a graph)

**SETTING UP WEB PAGE**

RESOURCE: https://www.youtube.com/watch?v=uU8-Ik8rxuw&list=PL3W4xRdnQJHXzyqvV4oQiq8B\_6LXNPHE9&index=1

* Since flask is a framework it’s better to install flask in a virtual environment
  + Pip install virtualenv
  + Install flask in virtual environment
  + In virtual environment create file app.py
    - Import Flask
    - Define routes

RESULT: Blank web page with some text

**MAKING WEB PAGE PRETTY**

RESOURCE:

* <https://web.stanford.edu/group/csp/cs21/htmlcheatsheet.pdf>
* <https://www.youtube.com/watch?v=4L3tJmAdF98&list=PL3W4xRdnQJHXzyqvV4oQiq8B_6LXNPHE9&index=2>
* Will use flask module called render\_template
  + Searches for folder with html templates
  + We will need to create this templates file in our project directory
    - Inside the templates folder create html files that create the structure for each route (page)
    - Then in the app.py file we call these html pages using the function render\_template
* Need bootstrap as well
  + An open source framework from ?? that provides user interface components to clean and create attractive web pages that are compatible with modern browsers
  + This has a bootstrap has a flask extension
    - Pip install flask-bootstrap

So far I have figured out how to set up the webpage and manipulate its structure.

Now need to figure out how to incorporate python objects into flask/HTML.

**PASSING PYTHON OBJECTS TO APP – PICKLE**

RESOURCES:

* https://machinelearningmastery.com/a-gentle-introduction-to-serialization-for-python/#:~:text=Serialization%20refers%20to%20the%20process,the%20reverse%20process%20of%20deserialization.

The pickle module is used for serializing and de-serializing python objects

[Serialization](https://hazelcast.com/glossary/serialization/): The process of converting a data object into a series of bytes which saves the state of the object in a form that more easily transmittable when delivering another data storage platform like an application or some other destination.

* Pickle.dump() vs. Pickle.dumps()
  + *Dump -*  writes the pickled object to an open file
  + *Dumps –* returns the pickles object as bytes

De-serialization: The reverse process which constructs a data structure or object from a series of bytes

* This process recreates the object making the data easier to read and modify as a native structure in a programming language
* Pickle.load()
* Pickle.loads()

Jsonify could also be a resource in place of pickle

* Jsonify will return javascript objects while pickle will store files as bytes

Files

**Jobscraper.py**

RESOURCES:

* <https://flask.palletsprojects.com/en/2.2.x/api/>
* Before\_request()
* After\_request()
* [Redirect()](https://pythonbasics.org/flask-redirect-and-errors/)
* Url\_for

**Home.html**

**Recommender.html**

**Summary.html**