**General Instructions**

Develop and deploy a data dashboard. The Web Development lesson has all of the information you need. If you are new to web development, you might have to go back to the concepts and rewatch some of the videos. The "deployment" parts of the lesson should be especially helpful. The video in that part of the lesson shows how to deploy a web app to Heroku. And the associated exercise has a complete, functioning web app with visualizations.

Most of the work will involve:

1. Wrangling your chosen data set to get the data in the format you want
2. Writing Python code to read in the data set and set up Plotly plots
3. Tweaking HTML so that the website has the design and information that you want.

We are providing a template that uses the [**Bootstrap library**](https://getbootstrap.com/) and [**Flask framework**](http://flask.pocoo.org/). The template is the same one used to build the app in the course except the name of the app has been changed. In the template, everything has the generic name "myapp" instead of "worldbankapp". The template is set up so that you can use pandas for loading the data and Python to create the dictionaries needed for plotly.

You'll only need to modify the following files:

* wrangle\_data.py
* index.html

Although the front-end is already set up for you, you should change the links and titles in index.html. If you want to add more visualizations or remove visualizations, you'll need to adjust the front-end code in index.html accordingly. That will involve adding or removing rows and columns in the HTML file.

For deployment, you can use a back-end service like **[Heroku](https://heroku.com/" \t "_blank)**.

**How to Build the App**

You'll find a workspace in the next part of the lesson. The workspace already contains the template code with a working web app. The web app has a back-end and front-end. Recall that you can run the web app from the workspace:

To run the app from the workspace, open a terminal and type env | grep WORK. Note the WORKSPACEDOMAIN and WORKSPACEID. To start the web app, type python myapp.py.

You can open a new browser window and go to the address: http://WORKSPACESPACEID-3001.WORKSPACEDOMAIN replacing WORKSPACEID and WORKSPACEDOMAIN with your values.

However, there is no data for the visualizations. You'll need to write a Python script that reads in the data files of your choosing and sets up the plots for Plotly. The process will be exactly the same as the one presented in the web development course.

**If you need to upload any files to the workspace, you can do so by clicking on the plus (+) sign and choosing "add file" or "add folder".**

The template code is also available on GitHub as part of the [**data scientist nanodegree term 2 repo**](https://github.com/udacity/DSND_Term2/tree/master/lessons/WebDevelopment).

Test your app in the workspace to make sure that everything is working. You'll see that if you start the app without modifying any of the code, the app currently works.

You should also save your work to a [**GitHub**](https://github.com/) or **[GitLab](https://about.gitlab.com/" \t "_blank)** repository so that you can use your code as part of your professional portfolio.

Once you're ready to deploy the app, don't forget to remove the app.run() line of code in the myapp.py file (In the web development lesson, myapp.py was called worldbank.py). You'll need to add a Procfile and requirements.txt file as well. Follow the instructions in the web development lesson to learn how to deploy the app from the classroom. And always comment your code :-)!

Also, at the end of this page you're reading, you'll find information about a more advanced version of the data dashboard that you can build.

**Steps**

Here is a reminder of the steps you'll need to do:

* find a data set or a few data sets that you're interested in
* explore and clean the data set
* put the data into a csv file or files - you can use pandas or spreadsheet software to do this
* upload your data sets to the correct folder
* write a Python script to read in the data set and set up the Plotly visualizations
* set up a virtual environment and install the necessary libraries to run your app
* run your web app locally to make sure that everything works
* deploy the app to Heroku or some other back-end service

**Where to Build the Web App**

We are providing a workspace containing a web app template. You can use this template to build and deploy your web app within the classroom.

The classroom has an Ubuntu Linux environment. Developing the app locally on macOS should be very similar. On a Windows machine, the commands are slightly different and you'll need to use the command prompt. This link contains a [**comparison of MS-DOS vs Linux commands**](https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/4/html/Step_by_Step_Guide/ap-doslinux.html).

To install the Heroku command line interface on a Windows machine, follow the instructions [**here**](https://devcenter.heroku.com/articles/heroku-cli) on the Heroku website.

**Advanced Version of the Exercise**

If you'd like an extra challenge, consider using an API to obtain your data. API stands for *Application Programming Interface*. An API provides a convenient way for two applications to communicate with each other. To be more concrete you can pull data directly from the World Bank API, clean the data in the back-end using pandas, and then display the results on your front-end. This would be instead of using a csv file for your data.

The benefit is that if the data ever changes, your web app will automatically have the correct data. Many companies provide APIs for accessing their data including Facebook, Twitter, Google among others. As an example, here is an API for pulling data about [**DVDs, movies, books, and games**](http://developer.rovicorp.com/docs).

After the workspace, you'll find a set of concepts that explain how to use the World Bank API. Go through that material if you'd like an extra challenge for building your web app.

# Here are a few issues that students have run into:

* remember to remove the the line app.run(host='0.0.0.0', port=3001, debug=True) in the myapp.py file before deploying to Heroku; however, to run the web app from within the Udacity workspace, you do need that line of code
* In the web development lesson, the Flask application file was called worldbank.py. In the template code, the same file is called myapp.py. So the Procfile should contain the line web gunicorn myapp:app.
* The template files can be deployed to Heroku. In other words, the template files already contain a working, deployable web app. If you're having trouble deploying your app to Heroku, try deploying the template files first. That should help with debugging because then you will know if an issue relates to how you are interacting with Heroku, or if there's a bug in your code.