**Web App deployment with Dash and Heroku**

**Cheat sheet**

**Created for Aberdeen Python User Group**

[Arturo Regalado](https://www.linkedin.com/in/arturoregalado/)

[@arturoregalado](https://twitter.com/arturoregalado)

9 February 2022

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Essential links:**

* <https://devcenter.heroku.com/articles/git>
* <https://dash.plotly.com/deployment>
* <https://dash.plotly.com/>
* <https://id.heroku.com/login>

**What is difficult about deploying a web app?**

Most problems occur because we are analysts or researchers, not full-stack developers or similar. It is difficult for us because:

* Working with the terminal feels odd.
* Need to step out of the excel mindset and the Jupyter Notebook mindset.
* Because you use Anaconda, DO NOT USE ANACONDA. USE PIP AND VENV, YOU WILL SEE WHY WHEN CREATING REQUIREMENTS.TXT FILE.
* **The more difficult part has nothing to do with Python.** Creating and deploying the first web app is painful because we need to learn HTML, CSS, and Git. You will see that many questions today are most likely related those tools. The python deployment is relatively straightforward with dash and plotly.

**Use Pycharm** – Alongside not using Anaconda it is the best advise I can give.

* All your coding can be done in the same interface. All means all: CSS, HTML, Python, Git.
* Work in the terminal and python console in the same interface.
* Easy structure and project management.
* All other benefits of pycharm like linter, syntax color, etc…

**Steps to deploy app and follow the base script for the talk:**

1. Create a Github and Heroku account.
2. Create project structure in PyCharm – create a new virtual environment and set a new interpreter. This step is very important as each app should be self-contained and all the packages downloaded specifically used for the project.
   1. Structure should have a data and assets folder as a minimum.
   2. In assets create stylesheet.css for styling.
3. Create app.py file. For small applications, the file is divided in two sections: first is any data loading, cleaning, pre-processing and get results ready to show in your app. Remember, the web app needs to be created with the reader in mind, but also for you as analyst to maintain easily.
   1. Make the work easy for you. Of course, loading the data, processing, etc… can make your code “slow”, but…if you are a data analyst chances are you are using this web app in a small scale. If you are deploying a large project, you will be part of a group with other expert developers. Their job is to make the code fast and deployable, yours is to quickly prototype a data solution.
4. Create layout for the app. See the target app we are trying to build.
   1. We require a heading, a instructions paragraph, a button to change parameters, and a space to show the plot.
   2. Use HTML Div Tags to separate each block of the layout. Here begins the tricky part. We are using python to write HTML via the dash core components modules.
5. Once the essential layout is done, we can now make the data processing and plot
   1. We are gonna show fake data with the number of students in University of Aberdeen and Robert Gordon University.
   2. The radio button will select the uni, and a bar plot will make the data. Useful to show how functions work and callback.
      1. When using callbacks there are two options: you can either create all code and not automate as much or take a bit more time but code all functions to be as automatic as possible.
      2. Note how in the callbacks the component id references the ids of the dash core components and how the property changes the properties of the dash core components.
6. When the core app is done - deploy using the following steps.

Prerequisites and initial files

* 1. Prerequisites: make sure git and Heroku CLI are installed <https://git-scm.com/book/en/v2/Getting-Started-Installing-Git> and <https://devcenter.heroku.com/articles/heroku-cli#download-and-install>
  2. Initialize git = git init
  3. Install gunicorn package using pip
  4. Create git ignore file with minimum files: venv \*.pyc .DS\_Store .env
  5. Create Procfile – Tricky it is important to create the file without any extension. The easiest way is to go to the terminal move to the project root directory and type: echo web: gunicorn app:server > Procfile
  6. Finally create the requirements file. Also very simple from the terminal, the trick is to use venv and not anaconda. The requirements file tells Heroku what packages it needs to download. Pip freeze does it automatically. Just type: pip freeze > requirements.txt
     1. This is why you do not want to use Anaconda (I will show example). If not using venv then the requirements file comes in a format that heroku does not recognize. You might need to check manually. But mistakes are less common.

Git and Heroku Remote – Also tricky but simple once you understand. The core idea is we are pushing a local GitHub repository into a Heroku remote repository that hosts our app.

* 1. Login to your Heroku account via terminal so git and Heroku know where to push your app. Use heroku login -i command to enter information inside the terminal. Once log in you can continue with the steps in the website links at the beginning. This step is normally omitted and put as a given, but for us not specialist can take some time to figure out.
  2. Now create app: heroku create set-name
  3. git add . # put the dot it is to add all files to git
  4. git commit -m ‘name-for-commit’
  5. git push heroku master # deploy code to heroku
     1. Troubleshooting. When deploying you will most likely encounter errors that have to do with the requirements text. You might need to clean some of the packages that are not actually used.

1. After deployment – lets change features and style.
   1. To update the code and redeploy you only need to git add . and commit again, then push heroku again.

**Intermediate challenge: You will now build and deploy your app adjusting the script I already gave to you.**

Adjustments:

1. Add data for North East Scotland College.
2. Change the input to dropdown instead of radio buttons. Tip search dash for dropdown component.
3. In the heading section, add a link to your twitter account and handle. Tip: search for a tag in HTML.
4. Style with CSS, put background colour to the title, change font size of instructions paragraph. For instructions paragraph also change background colour and font colour.