BEND Webcast: Deploying your Flask App with PostgreSQL on Heroku

- Date & Time: April 20, 10:00 AM (PDT)
- Overview: Deploying your Flask App with PostgreSQL on Heroku
- Presenters: Rahul

What is Heroku?

Heroku is a platform as a service (PaaS) that enables developers to build and run applications entirely in the cloud and without the need for servers or administration

** PaaS (Platform as a Service) provides computing platforms which typically includes operating system, programming language execution environment, database, web server etc. Examples: AWS Elastic Beanstalk, Windows Azure, Heroku, Force.com, Google App Engine, Apache Stratos. So, it's kind of service where you get already setup environment to run your apps.

- It supports languages such as Ruby, Java, Node.js, Clojure, Python, PHP, Perl, and Scala.

Installation

Heroku toolbelt is a command line software client which helps in managing apps. It is available for all the major platform. Currently, this is the best way to interact with the heroku services.

Please visit the download page link to get the required version of the toolbelt.

Once the heroku toolbelt has been installed, you can use heroku command line to interact with its API's. You will first need to enter the heroku login credentials.

```
$ heroku login
Enter your Heroku credentials.
Email: rahul.ranjan@udacity.com
Password:
```

Example Project

We have created a sample app so that you can follow up along and later on.
Download here

Prepare App

I will be deploying the sample `commento` app on heroku which allows user to add comments and shows all the comments in a table view.

```
$ git clone https://github.com/rahulrrixe/commento.git
$ cd commento
```

Create App on Heroku

You will need to create the application on heroku and copy it's URL. If it is created using command line then you don't need to copy the URL. If not, then you have to add `heroku` in your git remote using command below.

```
$ git remote add heroku https://git.heroku.com/commento.git
```

It is similar to what we do on github i.e. before uploading any project we need to create it.

```
$ heroku create commento
Creating commento in organization heroku... done, stack is cedar-14
http://commento.herokuapp.com/ | https://git.heroku.com/commento.git
Git remote heroku added
```

```
* Note: The project name should be unique. If it gives error Name is already taken, then try with some other name.
```

Procfile

A Procfile is a mechanism for declaring what commands are run by your application's dynos on the Heroku platform. It follows the process model. You can use a `Procfile` to declare various process types, such as multiple types of workers, a singleton process like a clock, or a consumer of the Twitter streaming API. Consider this as a `app.yaml` file but much simpler.

For more details follow official documentation.

In our case, I need dynos(Linux Container) to run the `app.py` file and so Procfile content would be

```
web: python app.py
```

Procfile basically follows a specific template for defining commands and process i.e.

```
<process type>: <command>
```

and for our case `web` is a process and command is `python app.py`.

```
* Note:  It should be named Procfile exactly, and not anything else. For example, Procfile.txt is not valid. The file should be a simple text file.
```

Declare app dependencies

As you are deploying the app on a new linux container and so you will need to create the proper environment for running the app and to achieve this on heroku we will need to provide the packages list on which our app is depended upon.

For python app, it's the `` `requirements.txt` ``. To generate it, use following commands.

```
$ pip freeze > requirements.txt
```

```
* Note this not a best way to do it as it enlist all the packages installed in your
environment and not directly related to your app
```

Deployment

Once your `Procfile` and `requirements.txt` is ready and added to the git, you can go ahead and deploy it.

```
$ git add Procfile
$ git add requirements.txt
$ git commit -m "Add Procfile and requirements.txt for heroku deployment"
$ git push heroku master
```

Adding Postgres Database

The `heroku` add-on marketplace has a large number of data stores, from Redis and MongoDB providers, to Postgres and MySQL but only Postgres is available for free with 10 MB storage.

Some applications will have a small, free postgres database provisioned by default. You can check this by running

```
$ heroku pg:info
```

It depends on the buildpack whether a database is provisioned automatically. You can provision the dev database manually with following commands

```
$ heroku addons:create heroku-postgresql:hobby-dev
```

Once the database is setup we should promote it such that the DATABASE_URL environment variable will be set:

You will need to restart the app to update the configuration.

```
$ heroku ps:scale web=0
$ heroku ps:scale web=1
```

Resources:

Official Documentation: https://devcenter.heroku.com/articles/getting-started-with-python
Heroku Command Cheatsheet: http://ruten.ca/2012/02/15/heroku-cheatsheet-useful-heroku-commands-reference/