Deployment with Heroku

Heroku

- a platform as a service, runs on Amazon Web Services
- easier and faster to deploy, but far less customization

Installing Heroku

- Make sure homebrew is installed <https://brew.sh/>
- Sign up for an account on Heroku <https://signup.heroku.com/>
- Install the Heroku CLI

```
$ brew install heroku/brew/heroku
```

Installing gunicorn

When we deploy an application in production, we will always want to use a server that is production ready and not meant for just development.

The server we will be using is gunicorn so let's make sure we:

```
(venv) $ pip install gunicorn
```

Ensuring a correct requirements.txt

- Heroku needs to know our dependencies!
- Make sure you `pip freeze > requirements.txt`

Adding a Procfile

- When we push code to Heroku, we need to tell Heroku what command to run to start the server.
- This command must be placed in a file called ***Procfile***.
- Make sure this filename does not have any extension and begins with capital P.

```
$ echo "web: gunicorn app:app" > Procfile
```

Adding a runtime.txt

- To make sure you are using a certain version of Python on Heroku, add a file called ***runtime.txt*** and specify the version of Python you want to use.

```
$ echo "python-3.7.2" > runtime.txt
```

Creating your Heroku app

- Login to your heroku account
- Create an application and make sure you have a correct remote.
- Push your code to the new remote and make sure you have a worker.
- Open your heroku app!

```
$ heroku login
$ heroku create NAME_OF_APP
$ git remote -v           # make sure you see heroku
$ git push heroku master  # make sure you've added & commited!
$ heroku open
```

Debugging a Heroku application

It's **never** going to work perfectly the first time. Make sure you look at the server logs to debug!

To see what went wrong, check out the server logs:

```
$ heroku logs --tail
```

Environment Variables

Since we're on a different server, we need different environment variables values:

```
$ heroku config:set SECRET_KEY=nevertell FLASK_ENV=production
$ heroku config    # see all your environment variables
```

```
import os

# use secret key in production or default to our dev one
app.config['SECRET_KEY'] = os.environ.get('SECRET_KEY', 'shh')
```

Adding a Postgres Database

In order to use a production database, we need Heroku to make one:

```
$ heroku addons:create heroku-postgresql:hobby-dev
$ heroku config   # you should see DATABASE_URL
```

Making sure you connect to the correct database

Now that we have a postgres database, we need to make sure that we are connecting to the correct database when in production!

```
import os

app.config['SQLALCHEMY_DATABASE_URI'] = os.environ.get(
    'DATABASE_URL', 'postgresql:///flask-heroku')
```

Connecting to *psql*

```
$ heroku pg:psql
```

Running a SQL file on Heroku

```
$ heroku pg:psql < data.sql
```

Running commands on your production server

```
$ heroku run python seed.py
```

Heroku hints

- Make sure you've added and committed before pushing to production
- If things break **ALWAYS** go to ***heroku logs –tail*** and see what's breaking
- If CSS or JS is not loading, check the Chrome console — make sure you're serving over HTTPS

> **Note: Further Reading**
>
> There are a number of helpful guides on the Heroku Dev Center that walk you step-by-step through deploying applications in the technology of your choice. Guides for Python projects can be found here <https://devcenter.heroku.com/categories/python-support>.