MAKE SCHOOL

Deployment

BEW 1.2

Agenda

MAKE SCHOOL

- Learning Outcomes
- Warm-Up
- What is Deployment?
- Deployment with Heroku
- Activity: Deploy your Project
- **BREAK**
- Lab Time

Learning Outcomes

MAKE SCHOOL

By the end of today, you should be able to...

1. Explain the difference between a PostgreSQL database & SQLite database.
2. Deploy a Flask application to Heroku using PostgreSQL.

Warm-Up

Warm-Up - Reflection Questions (6 minutes)

MAKE SCHOOL

In a group of 3, answer the following reflection questions:

1. What is the most interesting topic we have covered this term?
2. What is the most useful topic we have covered this term (and how do you plan to use it in your projects)?

MAKE SCHOOL

What is Deployment?

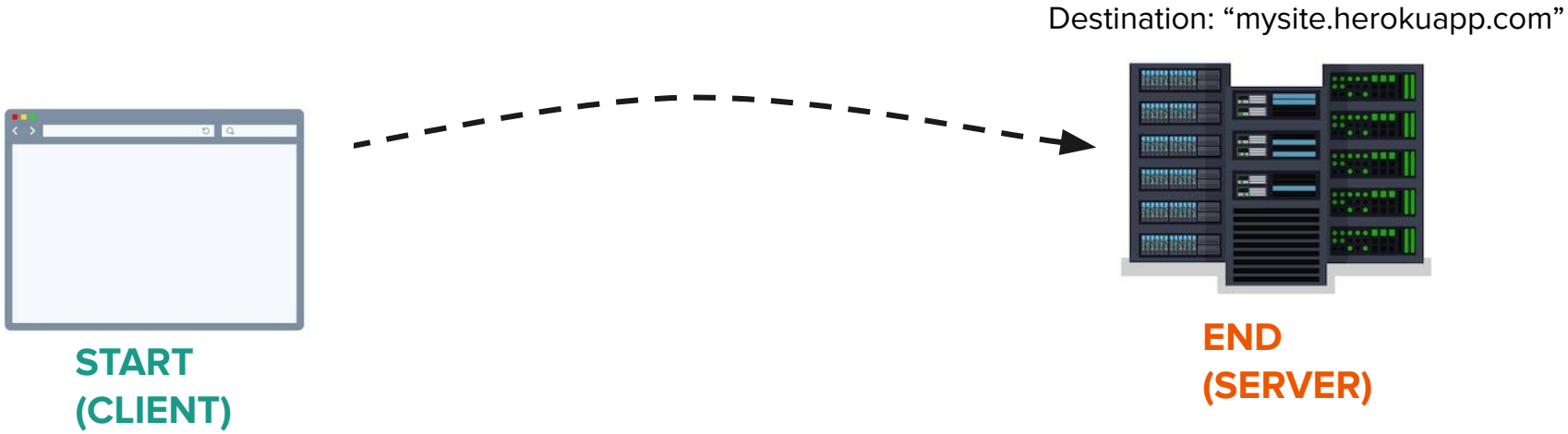
What is Deployment?

So far in this course, we've been running our applications on our local machines. This makes it easier to develop, debug, & test our work.

However, that means that only we can interact with it! The link won't work for anyone else.

What is Deployment?

Now, we're going to push our code to a **deployment environment**, which means it'll be hosted in a **datacenter**.



What is Deployment?

MAKE SCHOOL

Deployment in web development means pushing code changes from one deployment environment to another.

Development environments include:

- **Local**: Your own computer
- **Staging**: Intended for test users; usually available at a special staging URL
- **Production**: Intended for all users; usually available at a URL you release to the public

Why Deploy?

MAKE SCHOOL

Deploying your application to production can allow you to:

- Show your work to employers
- Show your work to friends or family
- Get test users
- Get real users / paying customers

What is Deployment?

MAKE SCHOOL

Since Make School doesn't own any datacenters (yet), we'll be using a service called **Heroku** to host our code.

"Heroku is a cloud platform that lets companies build, deliver, monitor and scale apps — we're the fastest way to go from idea to URL, bypassing all those infrastructure headaches."

MAKE SCHOOL

Types of Deployment Environments

Example: Facebook News Feed

Let's pretend you're an engineer at Facebook in 2006, and you've just completed a new feature - the News Feed. You've written all the code, and now you want to ship it to users! What do you do?

Example: Facebook News Feed

MAKE SCHOOL

Step 1: Test changes (using automated & manual testing) locally to make sure they work on your machine

Step 2: Push changes to a small number of users (such as **beta users**) to catch any remaining bugs

Step 3: Push to all users. Feature is launched!

Environments

MAKE SCHOOL

There are **three types of environments** we typically refer to in the software development lifecycle. Large companies (such as Google, Facebook, and yes, Make School!) use all three, and sometimes more.

1. **Development**
2. **Staging** ← **we'll use this one today**
3. **Production**

Development

MAKE SCHOOL

Your **development environment** is the one you've been using all along: your code is hosted on your own machine, and only you can run it. This is also called the "local environment". It is...

- Where **all** code updates occur
- Where you will run your tests, debug, & fix any issues
- Easier & quicker to read error messages here
- Any changes made will not affect the live website.
- Example: `http://localhost:5000`

Staging

MAKE SCHOOL

The **staging environment** is where we will push to today.

- All code lives on a server, not your machine.
- As similar to production as possible.
- Used in industry to beta test & catch any final issues before rolling out to production.
- Example: `https://projectname-test.herokuapp.com`

Production

MAKE SCHOOL

The **production environment** is the one that is rolled out to customers. It is the highest priority environment.

- Where we "**go live**", "**launch**", or "**ship**" our website.
- Real, live people will find any bugs you missed while coding!
- Example: `https://www.projectname.com`

Differences between environments

MAKE SCHOOL

In general, we want our environments to be **as similar as possible**. This will allow us to debug any issues we find in production using our local environment.

However, a few minor differences are allowed:

- The values of **environment variables** & **configuration settings** - e.g. the location of your database
- Different **data** in your local database vs. production

Addendum: Virtual environment??

MAKE SCHOOL

However, do **not** get these confused with your **virtual environment** - which is not a type of deployment environment at all!!!

A **virtual environment** is merely a container into which you can install any Python packages (a "sandbox"). It is used for local development so that you can keep your projects completely separate.

MAKE SCHOOL

Postgres Database

Postgres Database

MAKE SCHOOL

So far in this course, we've been using a SQLite database, because it's easy to set up & use.

However, it doesn't work well with Heroku, because Heroku does not guarantee that changes made to `database.db` will persist. (Basically, it can delete your data.)

So, we need a database that will persist over time.

Postgres Database

MAKE SCHOOL

PostgreSQL is a powerful, open source object-relational database system that uses and extends the SQL language combined with many features that safely store and scale the most complicated data workloads. (Source)

It works well with Heroku and is easy to set up!

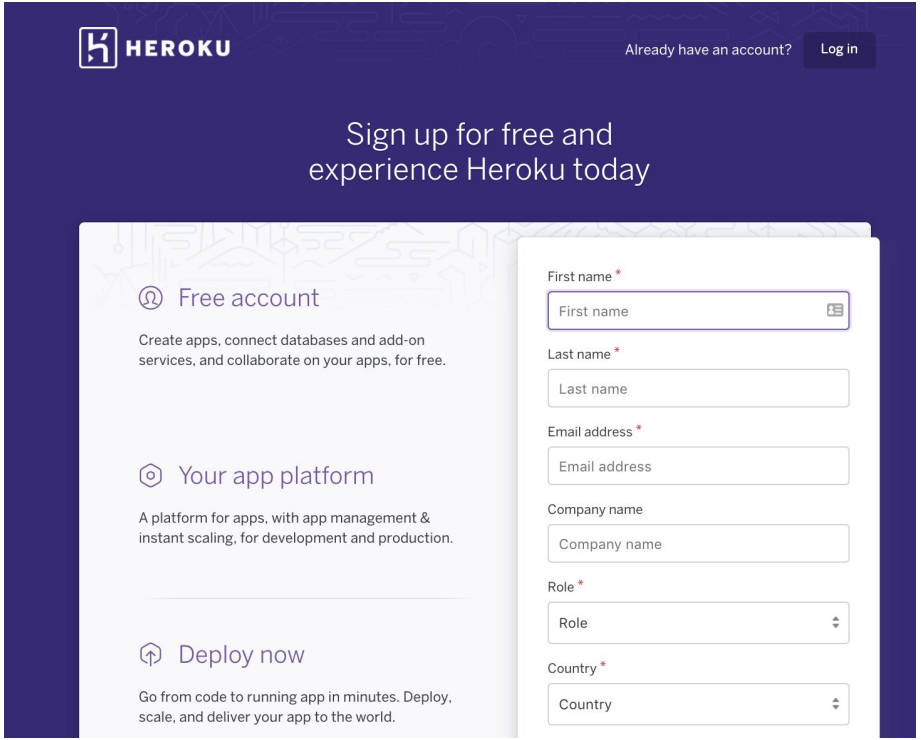
MAKE SCHOOL

Walk-Through: Deployment

Heroku

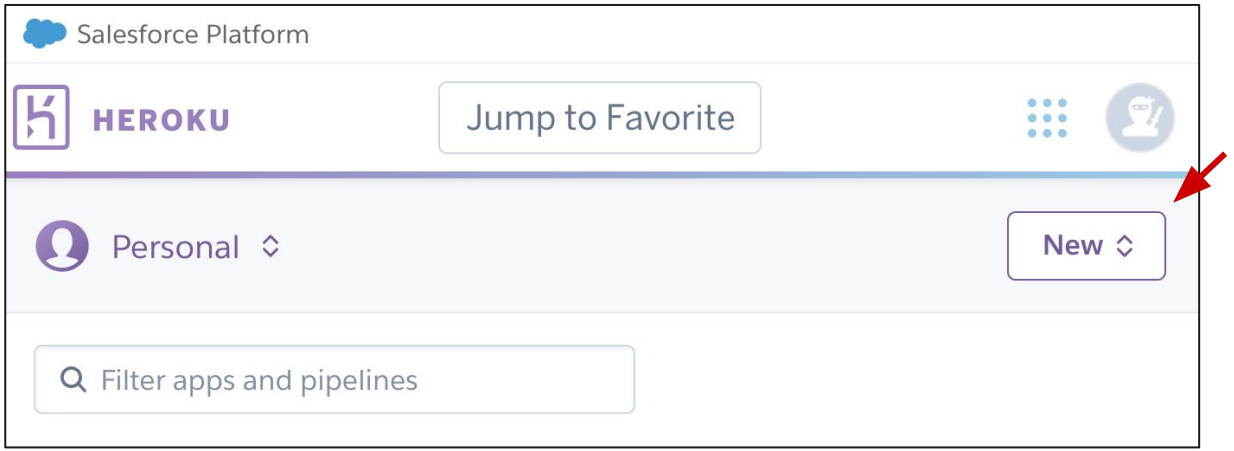
MAKE SCHOOL

Sign up for a Heroku account:



Heroku

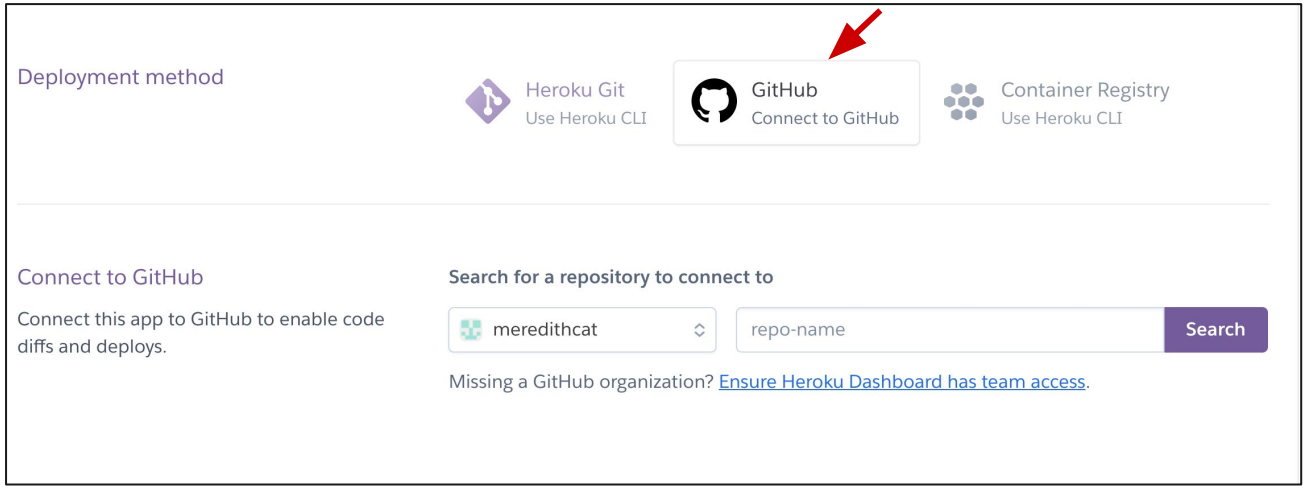
Go to dashboard.heroku.com/apps and create a new app:



Heroku

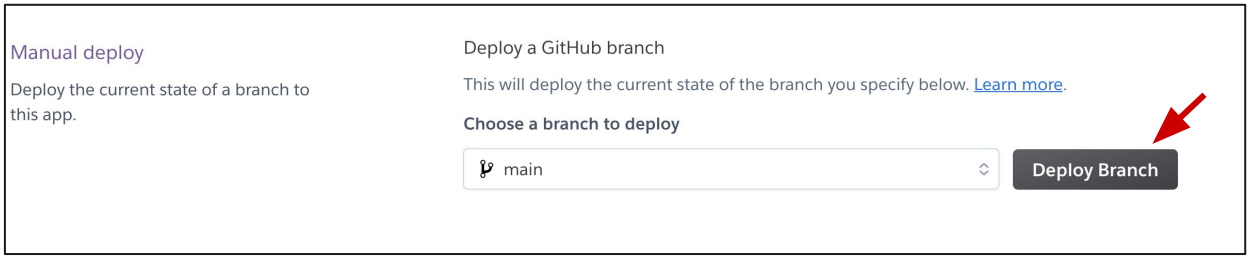
MAKE SCHOOL

Choose "Deployment Method: GitHub" and choose your repository:



MAKE SCHOOL

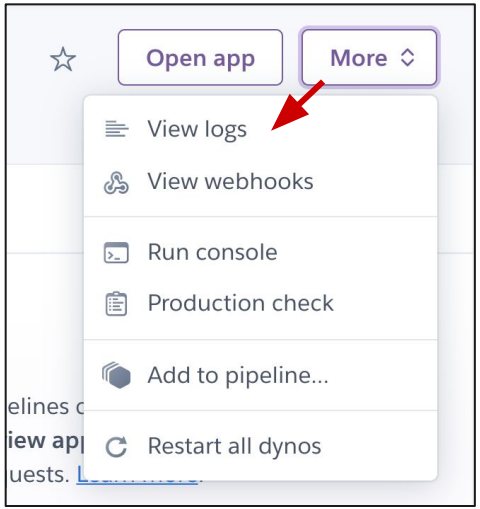
Click on "Manual Deploy -> Deploy Branch":



Heroku

Try opening your application! It won't run successfully the first time - and that's ok. Let's see what went wrong.

Click on "More -> View logs" to see the logs:



Add Procfile

MAKE SCHOOL

Let's add a Procfile to tell our app how to run.

Create a file in the root directory named `Procfile` (no file extension) and give it the following contents:

```
web: gunicorn -w 1 app:app --preload
```

Then add `gunicorn` to your requirements.txt file, and try deploying again.

Check requirements.txt

MAKE SCHOOL

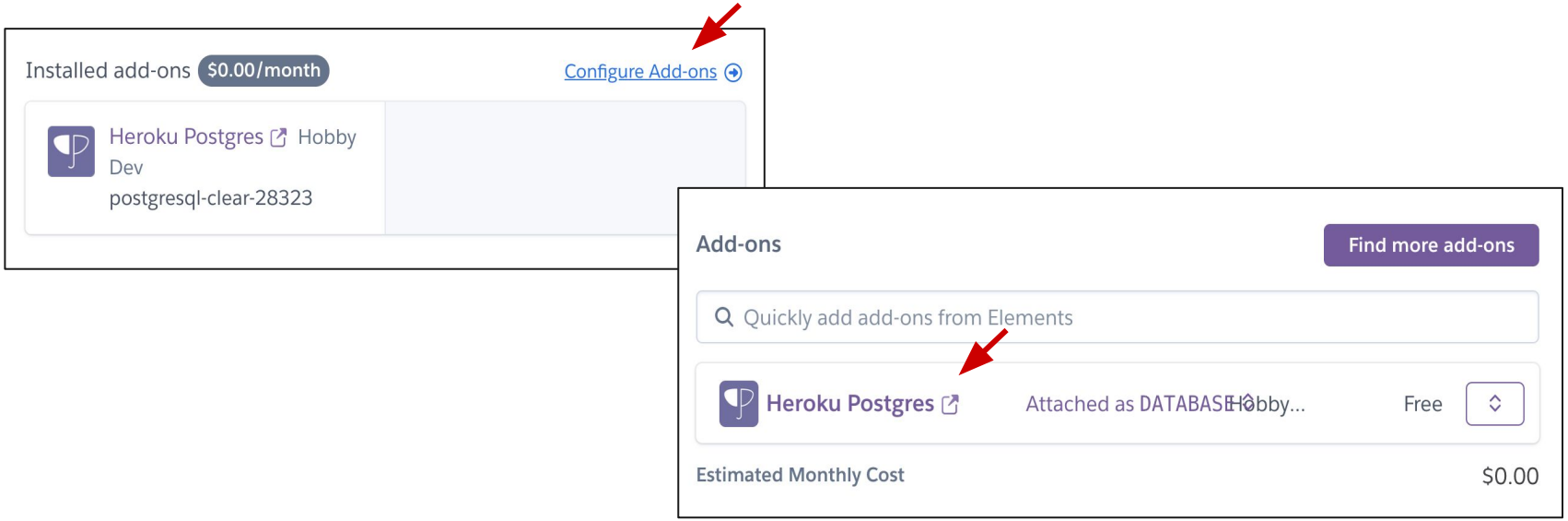
If you have issues with requirements.txt, try copying these contents and add any other project requirements you may have:

```
bcrypt==3.2.0
cffi==1.14.5
click==7.1.2
Flask==1.1.2
Flask-Bcrypt==0.7.1
Flask-Login==0.5.0
Flask-SQLAlchemy==2.4.4
Flask-WTF==0.14.3
gunicorn==20.1.0
itsdangerous==1.1.0
Jinja2==2.11.3
MarkupSafe==1.1.1
psycopg2==2.8.6
pycparser==2.20
python-dotenv==0.15.0
six==1.16.0
SQLAlchemy==1.3.23
SQLAlchemy-Utils==0.36.8
Werkzeug==1.0.1
WTForms==2.3.3
```

Postgres

Next up -- another error! This time, we need to switch out the database.

Go to "Overview -> Configure Add-ons" and search for "Heroku Postgres":



Postgres

MAKE SCHOOL

Now, we need to tell our application how to find the Postgres database!

Go to `app/config.py` and update the following line:

```
class Config(object):
    """Set environment variables."""


    SQLALCHEMY_DATABASE_URI = os.getenv('DATABASE_URL')
```

Then update your `.env` file to reflect the new variable name:

```
DATABASE_URL=sqlite:///database.db
```

Postgres

MAKE SCHOOL

Also, you'll need to add the `psycopg2` package to your `requirements.txt` file.

If you're using a virtual environment, you can instead install it with `pip3 install psycopg2` and `pip freeze > requirements.txt`.

Deploy

MAKE SCHOOL

Now, try deploying again. If you followed all of the steps, it should work!

Woo-hoo!!

If not, try using the logs to debug further.

Activity (50 minutes)

MAKE SCHOOL

Choose a project or homework assignment to deploy.

Follow this video to create a Heroku app and connect it to your GitHub repository.

Keep in mind that there are some differences between your code and the video. This is good practice for being able to understand other people's code and integrate it into your own!

MAKE SCHOOL

Break - 10 min

MAKE SCHOOL

Course Feedback Survey

Feedback Survey

MAKE SCHOOL

Please complete the course feedback survey: link

MAKE SCHOOL

Lab Time

Final assessment & project

MAKE SCHOOL

Cumulative final assessment on Wed/Thurs

You do not have to take the final if you scored >=70% on Quizzes 1 and 2.

Final project - please submit:

1. Code
2. Video & deployment

Last day to submit will be this Friday, the 14th