SEO Tech Developer

Data with ORMs and Deploying

What you will be able to do:

- Create a relational database connected to a website
- Use ORMs to setup a SQL table in Python
- Deploy a server-backed website
- Setup auto-deployment using Webhooks

SEO Tech Developer

Data with ORMs

SQLAlchemy (main.py)

- Now that we have made a user registration form, we need to store our usernames and passwords in a database
- Flask supports SQLAlchemy
  - `from flask_sqlalchemy import SQLAlchemy`
- Connect to your database

  -
    ```
    app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///site.db’
    db = SQLAlchemy(app)
    ```

SEO Tech Developer

ORMs

- Up until now we have been doing SQL queries on databases
- In some use cases, each database record (row) can be nicely represented as an object
- **O**bject-**R**elational **M**apping is a way to nicely map database columns to object attributes

SEO Tech Developer

SQLAlchemy ORM support (main.py)

```
class User(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    username = db.Column(db.String(20), unique=True, nullable=False)
    email = db.Column(db.String(120), unique=True, nullable=False)
    password = db.Column(db.String(60), nullable=False)

    def __repr__(self):
        return f"User('{self.username}', '{self.email}')"

with app.app_context():
    db.create_all()
```

SEO Tech Developer

Saving data on form submission (main.py)

```
@app.route("/register", methods=['GET', 'POST'])
def register():
    form = RegistrationForm()
    if form.validate_on_submit():
        user = User(username=form.username.data, email=form.email.data, password=form.password.data)
        db.session.add(user)
        db.session.commit()
        flash(f'Account created for {form.username.data}!', 'success')
        return redirect(url_for('home'))
    return render_template('register.html', title='Register', form=form)
```

SEO Tech Developer

Test database (terminal)

Run your Flask app and submit a form successfully.

```
export FLASK_APP=main_app_file_name

flask shell

>>> from app_py_file_name import User

>>> User.query.all()
```

You should see the newly added user printed out!

SEO Tech Developer

SEO Tech Developer

Deploying

Deploying

- For others to access your flask website, you need to **deploy** it to a **hosting service**.
- Flask is not a static website, it has a server backend, which makes hosting more complicated.

- We will use pythonanywhere.com
  - You can create *multiple* FREE accounts with the same email
  - Note: your username becomes part of the URL so choose something related to your PROJECT

Requirements.txt

- Create a requirements.txt file:

```
click==8.1.3
email_validator==2.0.0.post2
Flask==2.3.2
flask_behind_proxy==0.1.1
flask_sqlalchemy==3.0.5
Flask-WTF==1.1.1
GitPython==3.1.31
itsdangerous==2.1.2
Jinja2==3.1.2
MarkupSafe==2.1.3
pycodestyle
pytest
Werkzeug==2.3.6
```

- You can always check what you have installed using `pip3 list`
- Push `requirements.txt` to your project Github before moving on

SEO Tech Developer

Setup PythonAnywhere.com

- Create account – remember to make username project name
- Click “Web” and click “Add a new web app” on left menu
- Click “Consoles” and Start a new “Bash” console
  - Clone Github repo (use HTTPS URL) on command line
  - Delete any starter code that was generated for you
  - `cd` into repo
  - `pip3 install -r requirements.txt --user`

SEO Tech Developer

Setup PythonAnywhere.com

- Under “Web” in “Code” section
  - Change Source Code path to `home/project/repo`
  - Change Working directory to `home/project`
  - In WSGI configuration, update:
    - `project_home = '/home/project/repo'`
    - `from app import app as application  # noqa`
      - Change from `flask_app` to main python file name

  - Try reloading and visiting your flask application

SEO Tech Developer

Auto-Deployment

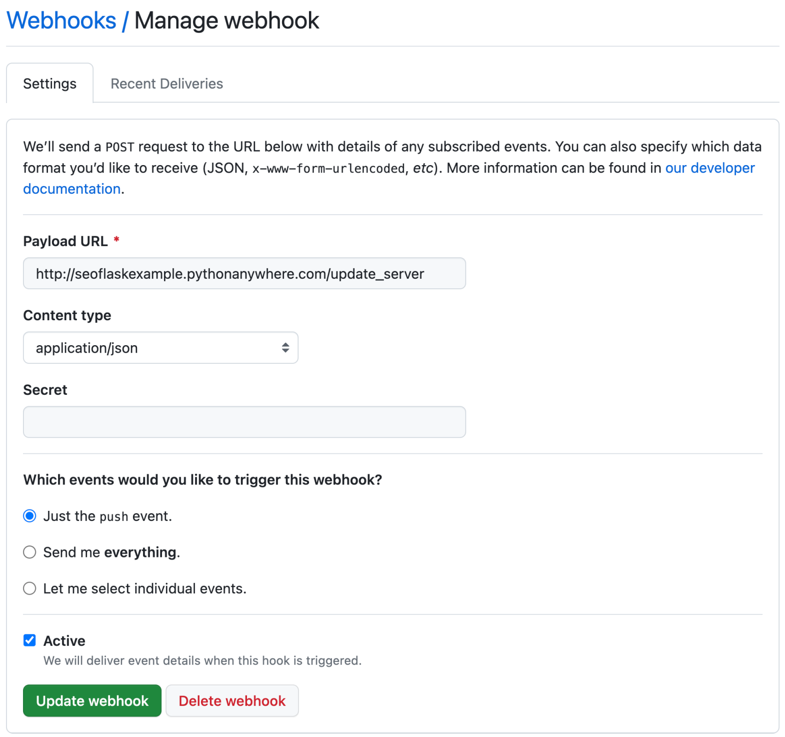
- It would be annoying to have to log in, pull and reload every time you pushed to Github

- We can use GitHub Webhooks to auto-deploy pushes to PythonAnywhere

SEO Tech Developer

Setup GitHub

- Go to your project GitHub repo > Settings > Webhooks
- Click "Add webhook"
- Enter Payload URL
  - http://projectname.pythonanywhere.com/update_server
- Choose application/JSON Content type



We'll send a POST request to the URL below with details of any subscribed events. You can also specify which data format you'd like to receive (JSON, x-www-form-urlencoded, *etc*). More information can be found in our developer documentation.

Payload URL *
http://seoflaskexample.pythonanywhere.com/update_server

Content type
application/json

Secret

Which events would you like to trigger this webhook?
[✓ Just the push event.]
Send me **everything**.
Let me select individual events.

[✓ Active]
We will deliver event details when this hook is triggered.

SEO Tech Developer

Setup in your python code (main.py)

```
from flask import Flask, render_template, url_for, flash, redirect, request
import git

...

@app.route("/update_server", methods=['POST'])
def webhook():
    if request.method == 'POST':
        repo = git.Repo('/home/projectname/github-repo-name')
        origin = repo.remotes.origin
        origin.pull()
        return 'Updated PythonAnywhere successfully', 200
    else:
        return 'Wrong event type', 400
```

SEO Tech Developer

Push to GitHub, pull via PythonAnywhere console, and reload site!

Setup PythonAnywhere.com

- Inside same Bash console in repo
  - Setup git hook
    - `cd .git/hooks`
    - `touch post-merge`
    - `nano post-merge`
    - Add and save file:

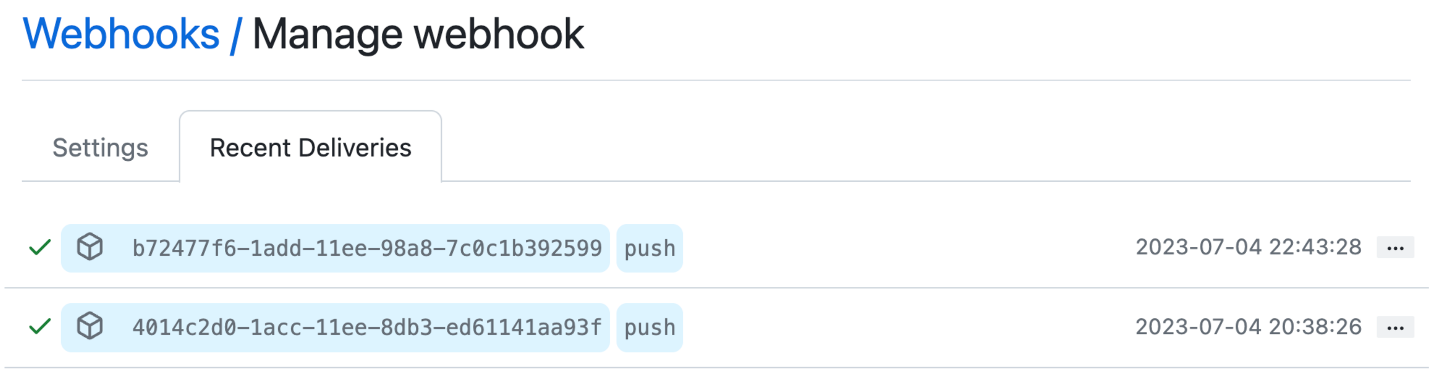
      -
        ```
        #!/bin/sh
        touch /var/www/projectname_pythonanywhere_com_wsgi.py
        ```
    - Make the file executable:
      - `chmod +x post-merge`

SEO Tech Developer

Try it out!

- You should be able to push a small change (like in your .css file) and visit your PythonAnywhere site to see the changes

- You should be able to check if your pushes are successful on GitHub > Settings > Webhooks > Recent Deliveries


Webhooks / Manage webhook

```
✓ b72477f6-1add-11ee-98a8-7c0c1b392599 push    2023-07-04 22:43:28
✓ 4014c2d0-1acc-11ee-8db3-ed61141aa93f push    2023-07-04 20:38:26
```

SEO Tech Developer

What questions do you have about…

- Creating a relational database connected to a website
- Using ORMs to setup a SQL table in Python
- Deploying a server-backed website
- Setting up auto-deployment using Webhooks

SEO Tech Developer

SEO Tech Developer

Thank you!