Traveling Through a Secure API with Python
Hello!

Auth0

Pizza de Dados

Data Bootcamp

LinkedIn Learning

Jess Temporal
Sr. Developer Advocate
@jesstemporal
Travelling Through a Secure API with Python and Auth0

What we will build
What we will build

1. 4 endpoints with Flask
2. Some data manipulation with Pandas
3. Protect endpoints with Auth0
4. Deploy to Heroku
Show me the code!
Basic endpoints

1. GET /
2. GET /oops
Basic endpoints

- GET /
- GET /oops
Travelling Through a Secure API with Python and Auth0

Protected endpoints

1. GET /
2. GET /oops
3. GET /ping
4. POST /places
Travelling Through a Secure API with Python and Auth0

Protected endpoints

- GET /
- GET /oops
- GET /ping

```python
app.py > ...
18
19 @app.get('/ping')
20 def update_map():
21     created = create_map()
22     if created:
23         return render_template('index.html')
24     return redirect(url_for('oops'))
25
```
Travelling Through a Secure API with Python and Auth0

Protected endpoints

- GET /
- GET /oops
- GET /ping
- POST /places

```python
app.py > ...

26 27 @app.post('/places')
28 29 def new_place():
29 30     place_data = request.json
30 31     create_new_place(place_data)
31 32     created = create_map()
32 33     if created:
33 34         return render_template('index.html')
34 35     return redirect(url_for('oops'))
35```

Data Data Data Data Data
# Data

<table>
<thead>
<tr>
<th></th>
<th>name, latitude, longitude</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Graz, 47.0913253131379, 15.475817295884495</td>
</tr>
<tr>
<td>2</td>
<td>Taipei, 25.031224572388727, 121.56835539277961</td>
</tr>
<tr>
<td>3</td>
<td>São Paulo, -23.553587531516293, -46.64191552881332</td>
</tr>
<tr>
<td>4</td>
<td>Ribeirão Preto, -21.175419814826252, -47.80700041467615</td>
</tr>
<tr>
<td>5</td>
<td>Florianópolis, -27.599309445589657, -48.48817975644357</td>
</tr>
<tr>
<td>6</td>
<td>Salvador, -12.947243144620645, -38.436411301896534</td>
</tr>
<tr>
<td>7</td>
<td>Maceió, -9.641966176203784, -35.73846456565925</td>
</tr>
<tr>
<td>8</td>
<td>Belo Horizonte, -19.848295063505297, -43.934907510577496</td>
</tr>
<tr>
<td>9</td>
<td>Brasília, -15.765777242405598, -47.90688628657218</td>
</tr>
</tbody>
</table>
Loading data

```python
def load_data():
    github = Github(config['github'])

    repository = github.get_user().get_repo(config['repo'])
    file = repository.get_contents('places.csv')

    return pd.read_csv(StringIO(file.decoded_content.decode()))
```
Create a map

```python
maps.py > ...

def creates_standard_map():
    center = (13.133932434766733, 16.103938729508073)
    return folium.Map(location=center, zoom_start=3)
```
Add markers

```python
def adds_markers(my_map, dataset):
    for _, place in dataset.iterrows():
        folium.Marker(
            location=[place['latitude'], place['longitude']],
            popup=place['name'],
            tooltip=place['name']
        ).add_to(my_map)

    return my_map
```
Save updated data

def save_data(dataset):
    github = Github(config['github'])
    repository = github.get_user().get_repo(config['repo'])
    filename = 'places.csv'
    sha = repository.get_contents(filename).sha

    s = StringIO()
    dataset.to_csv(s, index=False)
    content = s.getvalue()

    commit_message = "Update data via PyGithub"
    repository.update_file(filename, commit_message, content, sha)
Putting all together

def create_map():
    try:
        df = load_data()
        my_map = creates_standard_map()
        marked_map = adds_markers(my_map, df)
        marked_map.save('templates/index.html')
    except:
        return False
    return True
Putting all together

def create_new_place(place_data):
    df = load_data()
    result = df.append(place_data, ignore_index=True)
    result = result.drop_duplicates(keep=False)
    save_data(result)
Protection!
Travelling Through a Secure API with Python and Auth0

Create an API in Auth0
Create an API in Auth0
Create an API in Auth0
Create an API in Auth0
Create an API in Auth0

1. Choose a JWT library
As your API will be parsing JWT formatted access tokens, you will need to setup these capabilities on your API.

You can navigate to jwtt.io and choose from there. Remember to pick a library that support your selected signing algorithm.

2. Configuring your API to accept RS256 signed tokens
Configure the library that will validate the access tokens in your API. Validating a token means that you are certain you can trust its contents.
Create an API in Auth0

```
curl --request POST 
  --url https://jesstemporal.us.auth0.com/oauth/token 
  --header "content-type: application/json" 
  --data '{"client_id": "Fsavnk78br80xuZuhKhvs8usJn9bIUBF1", "client_secret": "A0xxqHl:NDJLqR3QZMTX_mc7p0x380C"}' 
```

In this example, `client_id` and `client_secret` are the ones from the Where Have I Been (Test Application) application.
Update the code

```python
# app.py

cross_origin(allow_headers=["Content-Type", "Authorization"], requires_auth)
```
Create an API in Auth0

Create the JWT validation decorator

Add a decorator which verifies the Access Token against your JWKS.

```python
# /server.py

# Format error response and append status code

def get_token_auth_header():
    """Obtains the Access Token from the Authorization Header"
    """
```
Travelling Through a Secure API with Python and Auth0

Create an API in Auth0

```bash
curl --request POST \ 
  --url https://jess-temporal.us.auth0.com/oauth/token \ 
  --header "content-type: application/json" \ 
  --data '{"client_id": "Foam70b8B8KZzoJhWKhvaASJ9b1UBF1", "client_secret": "AQxxQ4NJDyQR3QIMTX암CgZpOm3BCO"}'
```

In this example, `client_id` and `client_secret` are the ones from the Where Have I Been (Test Application) application. You can change these values with any from your other authorized applications.

**Response**

```
{
  "access_token": "eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiO
  "token_type": "Bearer"
}
```

You can inspect how this token is built at jwt.io
Deploy time!
Update the code

Procfile

web: gunicorn app:app
Travelling Through a Secure API with Python and Auth0

Update the code

```
[packages]
Flask = "2.0.0"
Flask-Cors = "3.0.10"
PyGithub = "1.55"
python-jose = "3.3.0"
folium = "0.12.1"
gunicorn = "20.1.0"
pandas = "1.2.5"

[requires]
python_version = "3.6"
```
Create a new app in Heroku

Install the Heroku CLI
Download and install the [Heroku CLI](https://devcenter.heroku.com/articles/getting-started-with-the-command-line)

If you haven’t already, log in to your Heroku account and follow the prompts to create a new SSH public key.

```
$ heroku login
```

Create a new Git repository
Initialize a git repository in a new or existing directory

```
$ cd my-project/
$ git init
$ heroku git:remote -a peaceful-sea-89241
```

Deploy your application
Commit your code to the repository and deploy it to Heroku using Git.

```
$ git add .
$ git commit -am "make it better"
$ git push heroku master
```
Let’s see how it looks!
Travelling Through a Secure API with Python and Auth0
Travelling Through a Secure API with Python and Auth0
Travelling Through a Secure API with Python and Auth0
Travelling Through a Secure API with Python and Auth0
Travelling Through a Secure API with Python and Auth0

GET Home
POST Get Auth Token
GET Generate New Map
POST Create New Place

```
1: {
2:  "name": "São José",
3:  "latitude": "-27.597724533718875",
4:  "longitude": "-48.61495221317449"
5: }
6
```

Dashboard / WHIB Talks Edition

POST _base_url places

Send 200 OK 2.37 s 12 KB

JSON Bearer Query Header

Preview

Header 1

Cookie Timeline

Map of Florianópolis with markers for São José.
That’s all for today!

The code