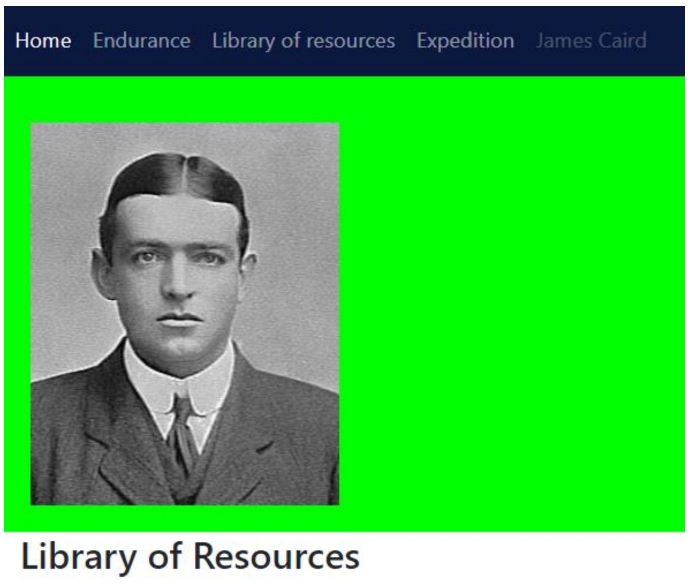
Flask Python (6) - Displaying Data

Step 1: Edit your *flask_app.py* file to include a new route to library. This will link to a *library.html* template.

Step 2: Create the library template *library.html* based on *base.html* and has the heading **Library of Resources**.

Step 3: Update your navigation bar to add a link to this page and check it works. It should look like this:



Library of Resources

Step 4: We are now going to retrieve the data from the database and list the books we have entered on the library resources page.

Importing modules

We have a directory called library and inside we have a file called model.py which contains our database functions. We need to import these into our flask_app.py file. We do this in two steps.

Step 1: Create a file called *__init__.py* in the library directory. This is a blank file, but it tells python that the files can be included as modules. For information when a directory is imported this file will be run so objects can be created here and then used if you need them.

Step 2: Put the following code at the top of your *flask_app.py* file:

```
from library.models import get_books
```

This allows us to use the `get_books()` function we wrote in *models.py* to retrieve data from our database.

Using the function

Step 1: We can now add a new route to the library and hence we amend our code in the library function in *flask_app.py*. It will look like this:

```
@app.route('/library')

def library():

    all_books = get_books()

    return render_template('library.html', all_books = all_books)
```

What we have done here is to create a variable `all_books` and assign to it the list of books we get when running `get_books().` We now have a list of our books.

Note that the `render_template` call now has a parameter `all_books` and we are sending the details of the books to *library.html*

Passing Parameters

Step 1: We have now passed a list of books from our database to the library page. To retrieve them we use the following syntax:

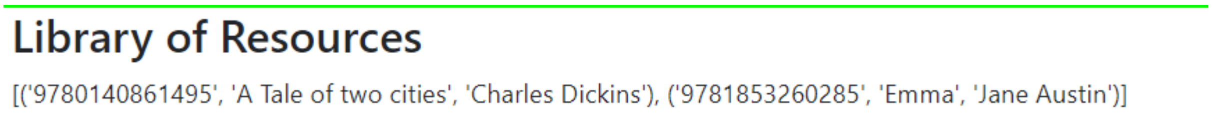
```
{{all_books}}
```

On your page the Jinja template engine will now look for the data passed to it when the template was rendered and display the `all_books` value (i.e. your book list of tuples).

Step 2: Your *library.html* should now look like this:

```
<div class="container w-50">
      <h3> Library of Resources </h3>
      {{ all_books }}
</div>
```

And it will render as follows on the web page:



Library of Resources

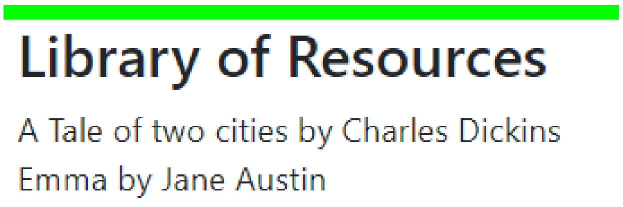
```
[('9780140861495', 'A Tale of two cities', 'Charles Dickins'), ('9781853260285', 'Emma', 'Jane Austin')]
```

Looping through a list

Step 1: This display isn't very user friendly so you can now loop through the list of books using this code:

```
<div class="container-fluid">

      <h3> Library of Resources </h3>

      {% for e in all_books %}

            {{e[1]}} by {{e[2]}}<br>

      {% endfor %}

</div>
```

Here we use the for loop (the for statement) provided by Jinja to find each element in our list and list the specific elements of the tuple. Here is the result:



Library of Resources

A Tale of two cities by Charles Dickins
Emma by Jane Austin