Baking Recipes - Coursework Report

Joanna Mein
40316854@live.napier.ac.uk
Edinburgh Napier University - Advanced Web Technologies (SET09103)

Abstract

The aim of the coursework was to develop a web application for an online catalogue using Python Flask. This report covers the design of the web-app including URL hierarchy, enhancements, critical evaluation and personal evaluation.

Keywords – Python, Flask, Bootstrap, Jinga2, JSON, Recipes, Recipe Catalogue, Baking, Catalogue, Occasion, Cooking Time

1 Introduction

Baking Recipes is a python flask web-app run in the browser that allows users to browse recipes. If they want something a bit more specific they can view recipes based on the occasion or the cooking time. If users are looking for recipes based on occasion they can choose from three categories:

- Halloween
- Bonfire Night
- Christmas

If users are interested in finding recipes based on the cooking time they can choose from four categories:

- Less than 10 minutes
- 10-20 minutes
- 20-30 minutes
- 30-40 minutes

2 Design

2.1 Languages

This web-app has been built using Python Flask with the use of HTML templates, CSS, JavaScript and Json. The main running of the web-app is done using Python Flask through Levinux, Terminal or similar command line software.

2.1.1 Python Flask

The structure and running of the web-app has been developed in Python Flask using Terminal on Mac, it can also be run using Levinux and other command line software. Flask is the framework which allow websites to be built in python and run in any web browser. Using Python Flask lets developers create web-apps without the need to write lines and lines of code. Features can normally be executed in just a few lines of code.

2.1.2 Jinga2 HTML Templates

The Layout and structure of the app has been created using jinga2 HTML templates which are rendered in the index.py file. For example, the Halloween page of this web-app is rendered as below using one simple line of code:

Listing 1: Render Template Example

```
1  @app.route('/recipes/halloween/')
2  def halloween():
3      return render_template('halloween.html')
4
```

2.1.3 JSON

JSON has been used to import recipes from a json file in to the relevant pages of the web-app. Json was chosen to hold all the data instead of a text file because it only needs one file which can hold all the recipes required for the web-app, whereas there would need to be multiple text files. Json is called in the index.py file as below:

Listing 2: JSON in index.py File

```
1  recipes = []
2  with open('recipes.json', 'r') as f:
3   recipes = json.load(f)
4   f.close()
5
6  p = {}
7  for item in recipes:
8   if item['name'] == "Rocky Road Crunch Bars":
9     print item
10    p = item
11
```

An example of the JSON file which stores all the data about each individual recipe is as below:

Listing 3: JSON File Storing Recipe Data

```
1  [{
2      "name": "Rocky Road Crunch Bars",
3      "cooking": "less than 10 mins",
4      "ingredients": ["125g/4.5oz soft unsalted butter",
5      "300g/10.5oz best-quality dark chocolate, broken into ↩
       pieces"
6  ],
7      "methods": [
8      "Heat the butter, chocolate and golden syrup in a heavy-↩
       based saucepan over a gentle heat. Remove from the heat, ↩
       scoop out about 125ml/4.5fl oz of the melted mixture and ↩
       set aside in a bowl.",
9      "Place the biscuits into a plastic freezer bag and crush ↩
       them with a rolling pin until some have turned to crumbs ↩
       but there are still pieces of biscuit remaining.",
10     "Fold the biscuit pieces and crumbs into the melted ↩
       chocolate mixture in the saucepan, then add the ↩
       marshmallows."
11   }]
```

12

The last step of implementing The JSON is to called it in the HTML template so that the recipes can be displayed on the correct web page.

Listing 4: JSON in HTML Template

```
<ul>
    {% for ingredient in rocky.ingredients %}
        <li>{{ ingredient }}</li>
    {% endfor %}
</ul>

```

2.2 URL Hierarchy

The design of the URL hierarchy has been developed with the users in mind so that they can easily navigate their way around the web-app. Users can move around using the Navigation bar which is located at the top of the screen and also the side navigation for more specific navigation. The page the user is on is reflected in the URL. For example, if the user is on the recipes page then the URL is "http://localhost:5000/recipes/". This makes it clear that the user is on the recipes page. Below is an example of how the URL is set in Python.

Listing 5: Recipes URL Hierarchy in Python Flask

```
@app.route('/recipes/')
def recipes():
    return render_template('recipes.html')

```

The hierarchy of the web-app is as below:

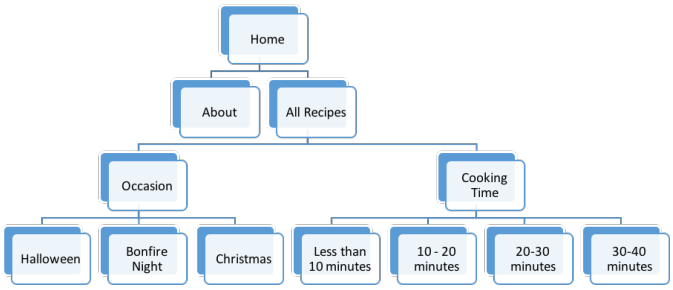

Figure 1: **URL Hierarchy** - Baking Recipes Web-app

2.3 Styling

2.3.1 Bootstrap

To style each page of the web-app Bootstrap has been used. Using this tool helps to makes the overall design consistent on every page. It also allows for easy layout design using the style sheet provided by Bootstrap. Custom styling has been implemented in this web-app to make it more appealing for the users. One of the other reasons Bootstrap has been used is because using the classes created it makes the pages responsive for all devices.

2.3.2 Imagery

Throughout the web-app there are banner images on every page to make it more interesting and eye catching for the user. The images selected are all related to baking and on the individual recipe pages the banner image reflects what the recipe looks like. Images were taken from Unsplash[1] and BBC Food[2].

2.3.3 Colour Scheme

The colour scheme implemented was selected using Coolor [3]. This online tool generates a combination of five colours which are complementary to each other. Using tools like this means that designers and developers avoid having colours which do not complement each other. The colour scheme used in this particular web-app is green, red, white and grey. The specific colours which have been used are:

- Green - #1C5253
- Red - #F26A681
- Grey - #D9CED2
- White - #FFFFFF

3 Enhancements

3.0.1 Add Recipes

A feature which could be added to improve the web-app would be to allow users to add recipes. Having this feature would enable the users to add their own recipes to the site to share with others. Having this feature would also add an element of community between users. It would also provide an easy way to share the recipes easily with friends and family by uploading it to the site and sharing a link with those who want it.

3.0.2 Contact Form

Another feature which could improve the web-app would be to add a contact form so that users can get in touch if they have any issues or questions related to the web-app. This would be a valuable feature because if users were having difficulty finding recipes then they would be able to send a message with their question or query for someone to get back to them. If the feature to add recipes was in place, then they could fill out the contact form with a question perhaps relating to how they add a recipe.

3.0.3 Search Bar

Another feature which could be added is a search bar. Having this feature allows more user interaction because it would let the users search for recipes in a variety of different ways. For example, they could search using specific keywords, recipe names, ingredients, User, etc. The possibilities are endless what users could search for on the web-app.

4 Critical Evaluation

The main feature of the web-app is the ability to using the side navigation, find recipes by occasion and cooking time. The developer feels this feature works well because it is a simple navigation menu which has drop downs so that when the user selects occasion they can then choose from Halloween, Bonfire Night or Christmas to find the recipes they want to make. Although they feel this feature works well it could be made better by adding in a page which displays all the occasions and another page which displays all the cooking times. This feature would allow the user who selected Cooking Times in the side navigation to be taken to a page which has all the cooking times which have recipes available.

The developer feels the individual recipe pages have a simple and clean look to them and the layout chosen makes it recognisable that they are recipe. One think that doesn't work so well is that users can only see what the recipes looks like from the banner image and on some pages you can't see the whole image. It might have been better to put an image next to the ingredients to show the full image of the recipe.

5 Personal Evaluation

When creating this web-app there were a few challenges to face. These included learning Python Flask, Bootstrap and importing data from a JSON file to the HTML templates.

5.0.1 Python Flask

To get familiar with Python Flask I worked my way through the Workbook provided[4] which gave examples of the skills and knowledge required in order to complete the coursework. I also used Learn Python the Hard Way[5] which provided examples of things you can do with flask to try out. I also used Code Academy[6] which provided interactive examples of the things which can be done with Flask. Using a combination of these three tools I feel I now have a better understanding of how Python Flask works and how to build a web-app using Flask.

5.0.2 Bootstrap

Before starting the development and design of this web-app I had never used Bootstrap so it was a tool which was completely new to me, although it was a tool which I was keen to learn. However, to get familiar with it I read through the documentation[7] provided on the bootstrap website which enabled me to learn what all the different classes do. It also provided examples of ways to style and layout web pages in a modern way. These examples gave me ideas of the sort of layout I wanted to use for my web-app to keep it clean and modern looking.

5.0.3 JSON in Python Flask

JSON was something else which I had never use, so again it is another skill which I gained by completing the coursework. To help me learn how JSON works and the ways it can be used to bring data into an HTML page I looked at some of the documentation[8] on the JSON website. This gave me some background information about what JSON is and what it can be used for. I then looked at examples python website which helped me to understand more about how JSON can be used with Python Flask. I also looked at examples[9] of how to structure my JSON file. Once I had a better understanding of how it worked I used Stack Overflow[10] to look at examples of how others had implemented it into Flask and Jinga2 Templates to make it work for the information I was displaying.

5.0.4 Overall Personal Reflection

Overall I feel the developing of the web-app went well without too many challenges are problems. Although challenges were faced I was able to overcome them easily with the help from articles and website online. I also feel the skills and knowledge I have gained through this development of the web-app will help me later on in university and my career after university.

References

[1] "Unsplash," October 2017.

[2] "Bbc food," October 2017.

[3] "Coolors," October 2017.

[4] S. Wells, "Notes & workbook 2017-2018," September 2017.

[5] Z. Shaw, "Learn python the hard way," October 2017.

[6] "Codeacademy," October 2017.

[7] "Bootstrap," October 2017.

[8] "Introducing json," October 2017.

[9] M. Wanyoike, "10 example json files," October 2017.

[10] "How to iterate over a list of list in jinja," October 2017.