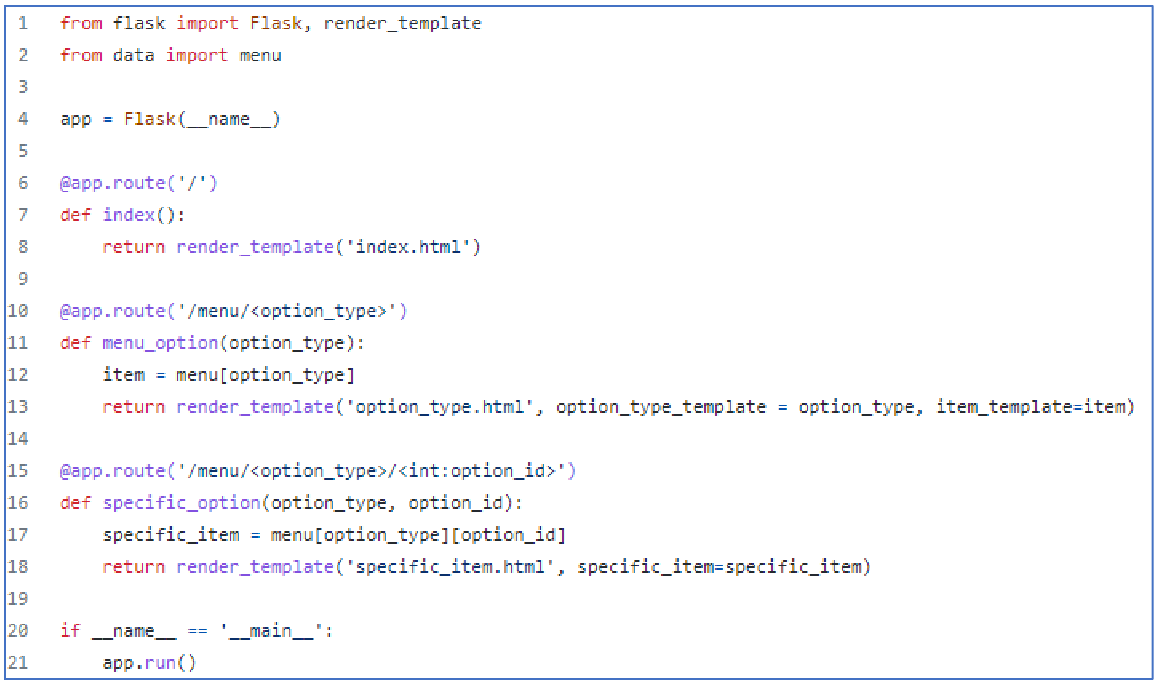
Rosal, Rona Leane, C. 2019-07307 LIS 161 – Project Documentation

Note: The live version of the app can be found by visiting: ronaleane.pythonanywhere.com

Github Link: https://github.com/ronaleane/lis161proj

Source Code

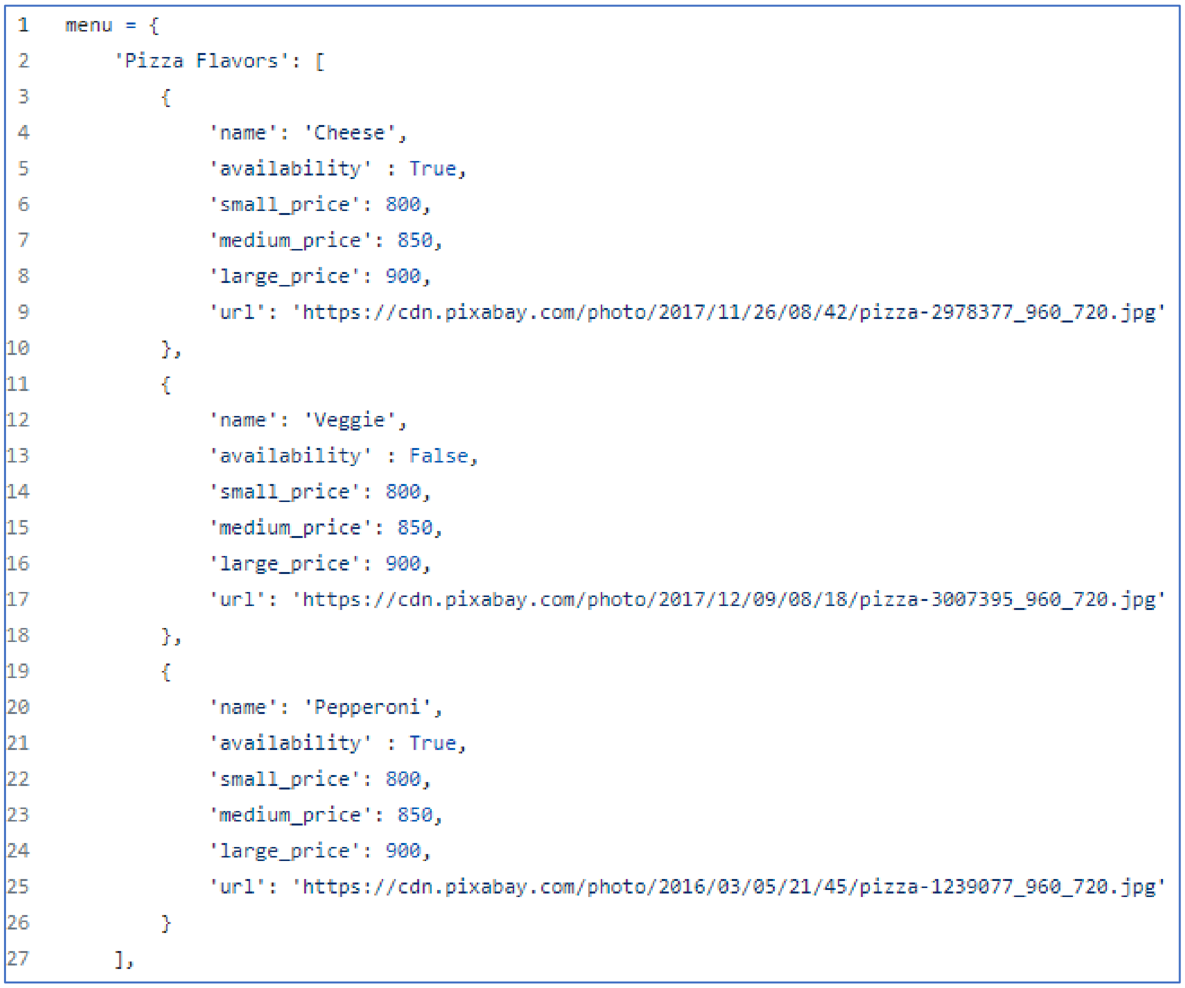
- Filename: flask_app.py
  - This is the main file for the flask application
  - This is named as such because it is the filename required by pythonanywhere.com



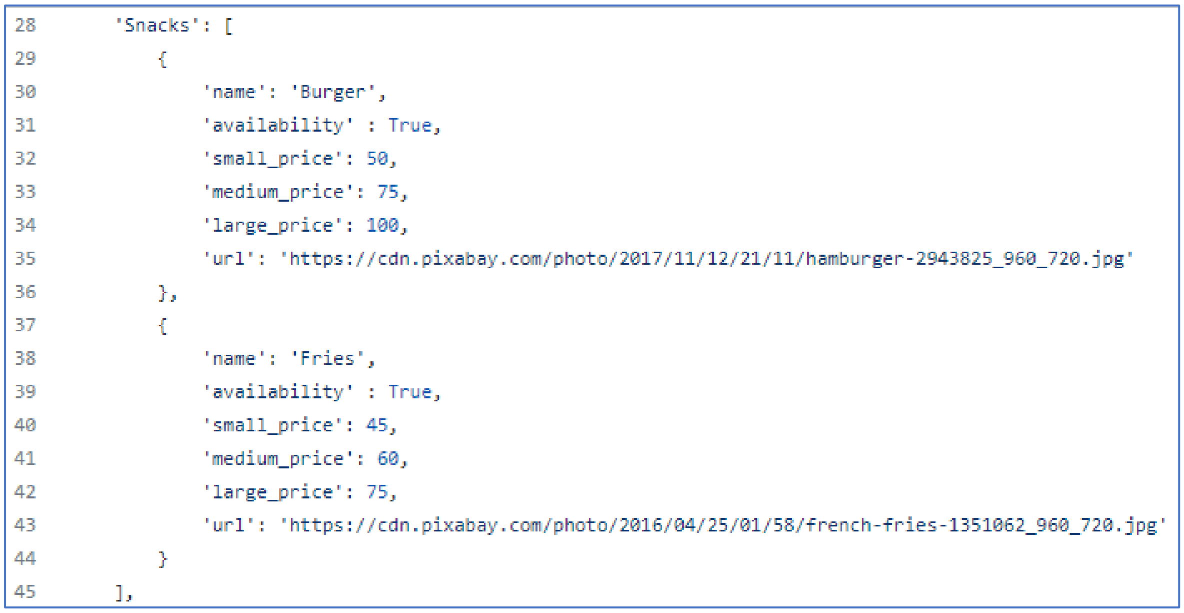
```
1  from flask import Flask, render_template
2  from data import menu
3  
4  app = Flask(__name__)
5  
6  @app.route('/')
7  def index():
8      return render_template('index.html')
9  
10 @app.route('/menu/<option_type>')
11 def menu_option(option_type):
12     item = menu[option_type]
13     return render_template('option_type.html', option_type_template = option_type, item_template=item)
14 
15 @app.route('/menu/<option_type>/<int:option_id>')
16 def specific_option(option_type, option_id):
17     specific_item = menu[option_type][option_id]
18     return render_template('specific_item.html', specific_item=specific_item)
19 
20 if __name__ == '__main__':
21     app.run()
```

  - Lines 1-2: Importing Flask and the render_template method. Flask is the python web framework used in this project. The method render_template is needed to render the html files found in the templates folder. Another imported file is called 'data', a python file containing all necessary information for the program. Menu is an object inside data.py that contains all data.
  - Line 4: Storing the instance of the flask application to a variable called 'app'
  - Lines 6-8: Defining the index or home route and function. The function renders the file 'index.html'.
  - Lines 10-13: Defining the route and function for the types of item in the menu. The variable item refers to the options, 'Pizza Flavors', 'Snacks', and 'Drinks' which will be found in the data.py file.
  - Lines 15-18: Defining the route and function for the specific options for each type of item in the menu. The 'item' is the python list that serves as the value for each of the keys: 'Pizza Flavors', 'Snacks', and 'Drinks'. This python list contains dictionaries, each containing the details about each specific version of the item type.
  - Lines 20-21: Ensuring that the program running is the main program

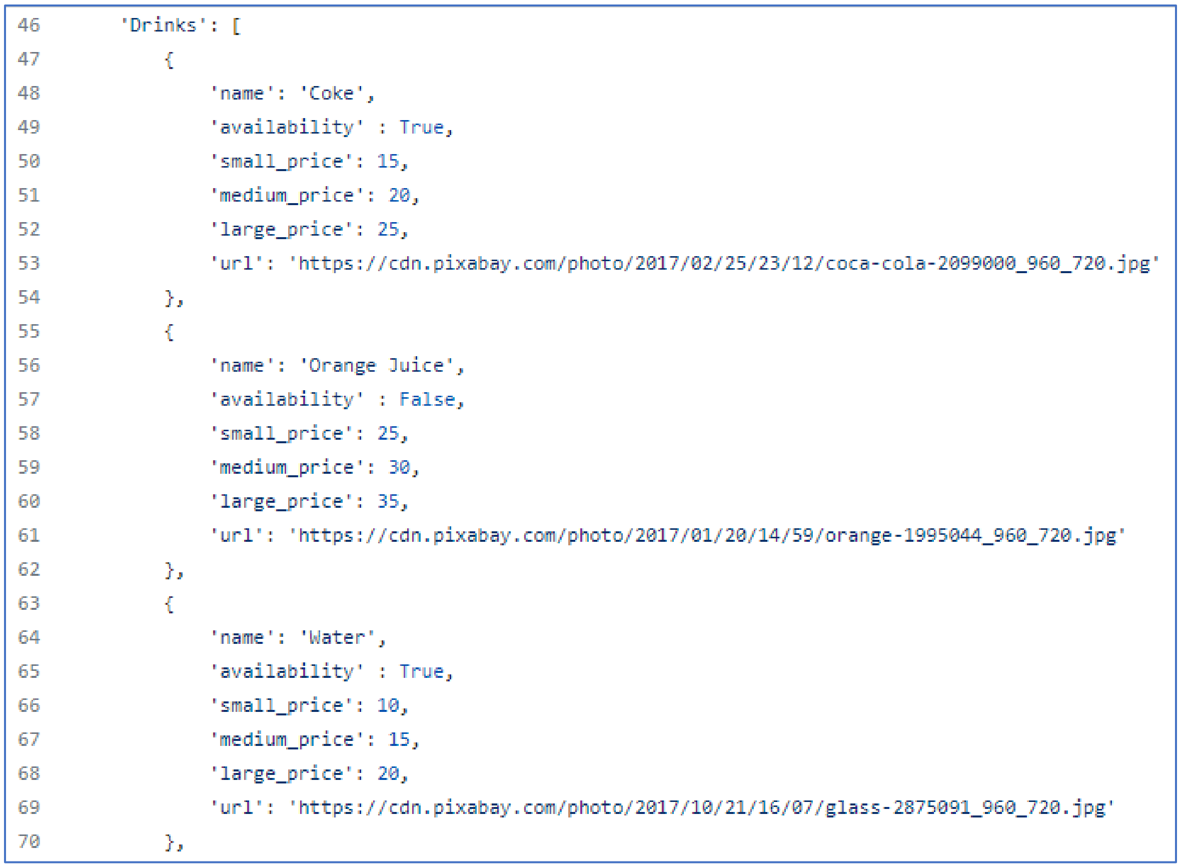
- Filename: data.py
    - This is the file that contains the data used for the program.



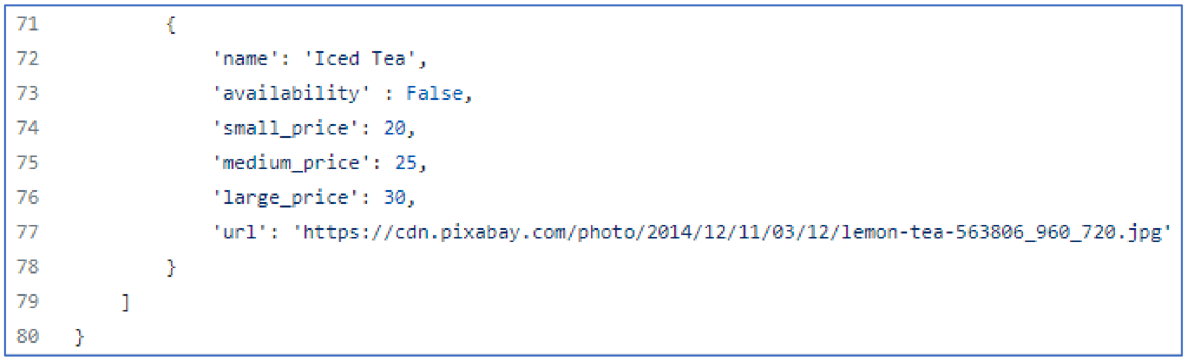
```python
menu = {
    'Pizza Flavors': [
        {
            'name': 'Cheese',
            'availability' : True,
            'small_price': 800,
            'medium_price': 850,
            'large_price': 900,
            'url': 'https://cdn.pixabay.com/photo/2017/11/26/08/42/pizza-2978377_960_720.jpg'
        },
        {
            'name': 'Veggie',
            'availability' : False,
            'small_price': 800,
            'medium_price': 850,
            'large_price': 900,
            'url': 'https://cdn.pixabay.com/photo/2017/12/09/08/18/pizza-3007395_960_720.jpg'
        },
        {
            'name': 'Pepperoni',
            'availability' : True,
            'small_price': 800,
            'medium_price': 850,
            'large_price': 900,
            'url': 'https://cdn.pixabay.com/photo/2016/03/05/21/45/pizza-1239077_960_720.jpg'
        }
    ],
```



```python
    'Snacks': [
        {
            'name': 'Burger',
            'availability' : True,
            'small_price': 50,
            'medium_price': 75,
            'large_price': 100,
            'url': 'https://cdn.pixabay.com/photo/2017/11/12/21/11/hamburger-2943825_960_720.jpg'
        },
        {
            'name': 'Fries',
            'availability' : True,
            'small_price': 45,
            'medium_price': 60,
            'large_price': 75,
            'url': 'https://cdn.pixabay.com/photo/2016/04/25/01/58/french-fries-1351062_960_720.jpg'
        }
    ],
```



```
46      'Drinks': [
47          {
48              'name': 'Coke',
49              'availability' : True,
50              'small_price': 15,
51              'medium_price': 20,
52              'large_price': 25,
53              'url': 'https://cdn.pixabay.com/photo/2017/02/25/23/12/coca-cola-2099000_960_720.jpg'
54          },
55          {
56              'name': 'Orange Juice',
57              'availability' : False,
58              'small_price': 25,
59              'medium_price': 30,
60              'large_price': 35,
61              'url': 'https://cdn.pixabay.com/photo/2017/01/20/14/59/orange-1995044_960_720.jpg'
62          },
63          {
64              'name': 'Water',
65              'availability' : True,
66              'small_price': 10,
67              'medium_price': 15,
68              'large_price': 20,
69              'url': 'https://cdn.pixabay.com/photo/2017/10/21/16/07/glass-2875091_960_720.jpg'
70          },
```

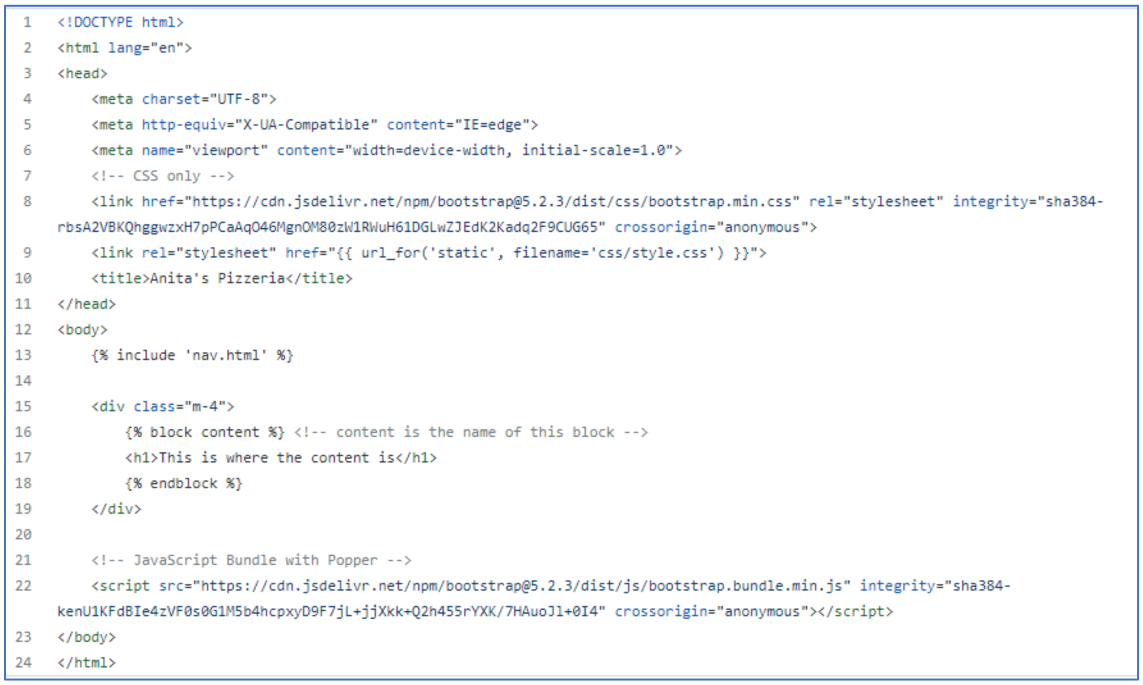


```
71          {
72              'name': 'Iced Tea',
73              'availability' : False,
74              'small_price': 20,
75              'medium_price': 25,
76              'large_price': 30,
77              'url': 'https://cdn.pixabay.com/photo/2014/12/11/03/12/lemon-tea-563806_960_720.jpg'
78          }
79      ]
80  }
```

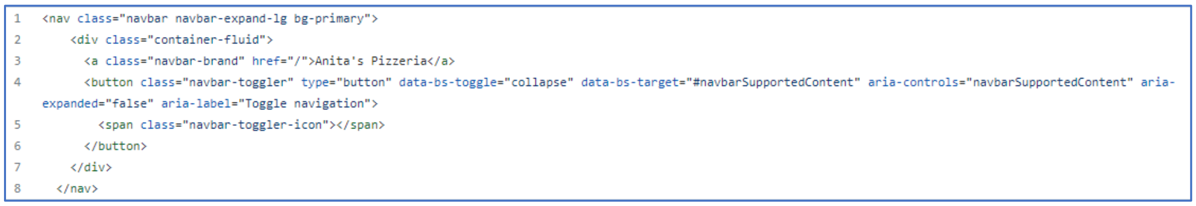
- Line 1: Creating the dictionary called 'menu'. It contains all the needed information for the program.
- Lines 2-25: Contains all the details for the product called 'Pizza Flavors'. 'Pizza Flavors' is an 'item' referred to in line 12 of the main application. Each dictionary inside the list (which is the value corresponding to key 'Pizza Flavors') refers to the 'specific_item' in line 17 of the main application.
- Lines 28-43: Contains all the details for the product called 'Snacks'. 'Snacks' is an 'item' referred to in line 12 of the main application. Each dictionary inside the list (which is the value corresponding to key 'Snacks') refers to the 'specific_item' in line 17 of the main application.
- Lines 46-77: Contains all the details for the product called 'Drinks'. 'Drinks' is an 'item' referred to in line 12 of the main application. Each dictionary inside the list

(which is the value corresponding to key ‘Drinks’) refers to the ‘specific_item’ in line 17 of the main application.

Templates

- Filename: base.html
    - 

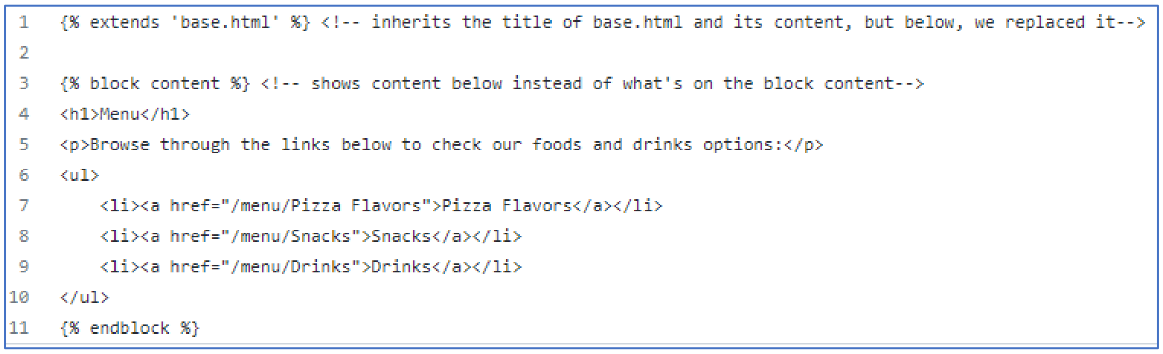
```
1   <!DOCTYPE html>
2   <html lang="en">
3   <head>
4       <meta charset="UTF-8">
5       <meta http-equiv="X-UA-Compatible" content="IE=edge">
6       <meta name="viewport" content="width=device-width, initial-scale=1.0">
7       <!-- CSS only -->
8       <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-
    rbsA2VBKQhggwzxH7pPCaAqO46MgnOM80zW1RWuH61DGLwZJEdK2Kadq2F9CUG65" crossorigin="anonymous">
9       <link rel="stylesheet" href="{{ url_for('static', filename='css/style.css') }}">
10      <title>Anita's Pizzeria</title>
11  </head>
12  <body>
13      {% include 'nav.html' %}
14
15      <div class="m-4">
16          {% block content %} <!-- content is the name of this block -->
17          <h1>This is where the content is</h1>
18          {% endblock %}
19      </div>
20
21      <!-- JavaScript Bundle with Popper -->
22      <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/js/bootstrap.bundle.min.js" integrity="sha384-
    kenU1KFdBIe4zVF0s0G1M5b4hcpxyD9F7jL+jjXkk+Q2h455rYXK/7HAuoJl+0I4" crossorigin="anonymous"></script>
23  </body>
24  </html>
```

    - This is the html file that shows how the pages of the web app will be formatted.
    - Line 8: Using Bootstrap for styling.
    - Line 9: Using the file called ‘style.css’ as another way to style the html files.
    - Line 10: When visiting the app, the tab for each page will be titled ‘Anita’s Pizzeria’.
    - Line 12: The base file will include nav.html. Since this is the parent of the inheritance, all child files will include the contents of nav.html.
    - Lines 15-19: This is the content block for the content of the pages. In case a page does not replace the code inside the block, the one in line 17 will be shown.
    - Lines 21-22: Importing Bootstrap JS Bundle with Popper
- Filename: nav.html
    - 

```
1   <nav class="navbar navbar-expand-lg bg-primary">
2       <div class="container-fluid">
3         <a class="navbar-brand" href="/">Anita's Pizzeria</a>
4         <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-
    expanded="false" aria-label="Toggle navigation">
5           <span class="navbar-toggler-icon"></span>
6         </button>
7       </div>
8     </nav>
```

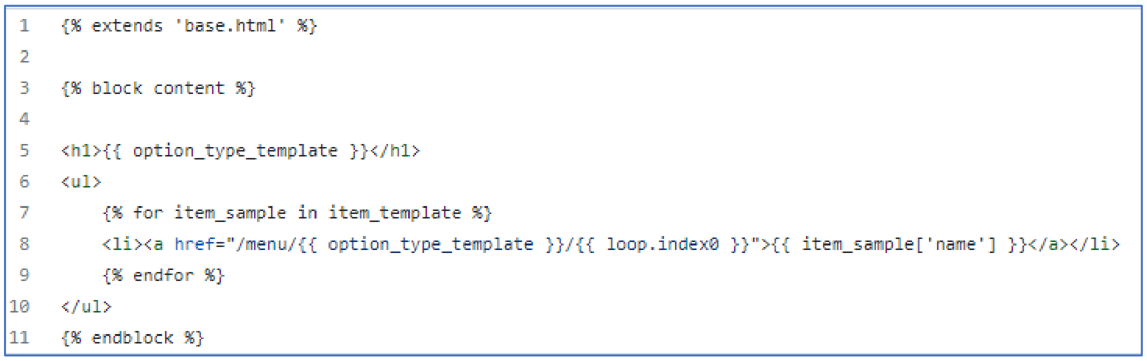
    - This is the file that adds a navigation bar to the website.
    - Line 3: The brand’s name for the navbar is ‘Anita’s Pizzeria’. Clicking it will direct the user to the homepage or index.
- Filename: index.html
    - This is the file corresponding to the landing page of the site.

    - This is rendered in line 8 of the main application.



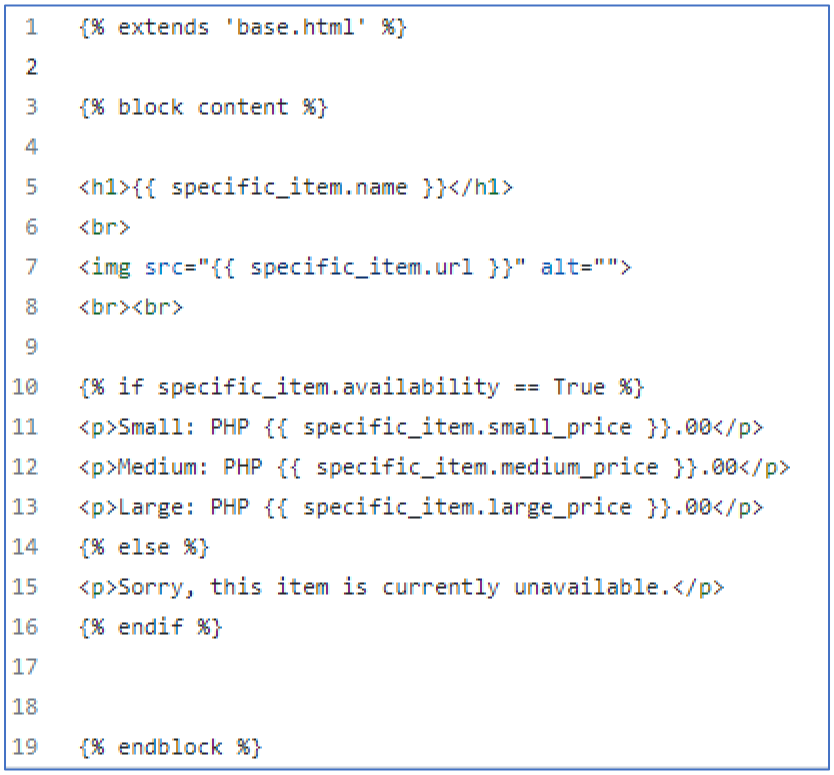
```
1   {% extends 'base.html' %} <!-- inherits the title of base.html and its content, but below, we replaced it-->
2
3   {% block content %} <!-- shows content below instead of what's on the block content-->
4   <h1>Menu</h1>
5   <p>Browse through the links below to check our foods and drinks options:</p>
6   <ul>
7       <li><a href="/menu/Pizza Flavors">Pizza Flavors</a></li>
8       <li><a href="/menu/Snacks">Snacks</a></li>
9       <li><a href="/menu/Drinks">Drinks</a></li>
10  </ul>
11  {% endblock %}
```

    - Line 1: It inherits the contents of base.html
    - Lines 3-11: The content of the homepage. It has a heading, a description, and 3 hyperlinks connecting to each item type on the menu. These are ‘Pizza Flavors’, ‘Snacks’, and ‘Drinks’, shown as list items.
- Filename: option_type.html
    - This is the file corresponding to the possible options per item type in the menu.
    - This file is rendered in line 13 of the main application.



```
1   {% extends 'base.html' %}
2
3   {% block content %}
4
5   <h1>{{ option_type_template }}</h1>
6   <ul>
7       {% for item_sample in item_template %}
8       <li><a href="/menu/{{ option_type_template }}/{{ loop.index0 }}">{{ item_sample['name'] }}</a></li>
9       {% endfor %}
10  </ul>
11  {% endblock %}
```

    - Line 1: It inherits the contents of base.html
    - Lines 3-11: This is the content for the pages for each item type in the menu.
    - Line 5: From line 13 of the main application, this refers to the names of the item types in the menu. These are ‘Pizza Flavors’, ‘Snacks’, and ‘Drinks’.
    - Lines 6-10: Create a list. The first part of each list item is an href attribute. The loop.index0 will loop through the contents of the item types. The second part is the clickable part of the link.
        - As an example, item_sample for the ‘Pizza Flavors’ are the three dictionaries inside the list. ‘item_template’ is the list the contains these three. One ‘item_sample[‘name’]’ is ‘Cheese’, which can be seen in line 4 of the data.py file.
        - Each of the ‘item_sample[‘name’]’ will be shown as a list item in the page.
- Filename: specific_item.html
    - This file will provide the item view for each specific product. An example is the ‘Cheese’ pizza flavor and its details.
    - This file is rendered in line 18 of the main application.

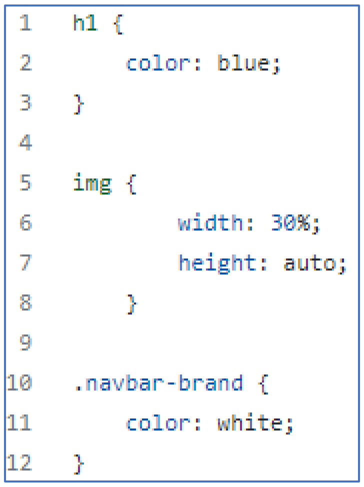


```
1   {% extends 'base.html' %}
2
3   {% block content %}
4
5   <h1>{{ specific_item.name }}</h1>
6   <br>
7   <img src="{{ specific_item.url }}" alt="">
8   <br><br>
9
10  {% if specific_item.availability == True %}
11  <p>Small: PHP {{ specific_item.small_price }}.00</p>
12  <p>Medium: PHP {{ specific_item.medium_price }}.00</p>
13  <p>Large: PHP {{ specific_item.large_price }}.00</p>
14  {% else %}
15  <p>Sorry, this item is currently unavailable.</p>
16  {% endif %}
17
18
19  {% endblock %}
```

- Line 1: It inherits the contents of base.html
- Lines 3-19: Content of the item view.
- Note: The explanations for lines 5-16 will use the product 'Cheese' for example. Details for the item can be found in lines 4-9 of the data.py file. Note that the variable 'specific_item' is the same with 'item_sample' that is used in the option_type.html file.
  - Line 5 shows the value of the key 'name' which is 'Cheese'.
  - Line 7 shows the image that can be found by following the link/value of the key 'url'.
  - Line 10 will check first the value of the key 'availability'. In this case, the value is 'True'.
  - Line 11-14 will be shown since the value is 'True'. Else, the page will show the paragraph in line 15.

Static File

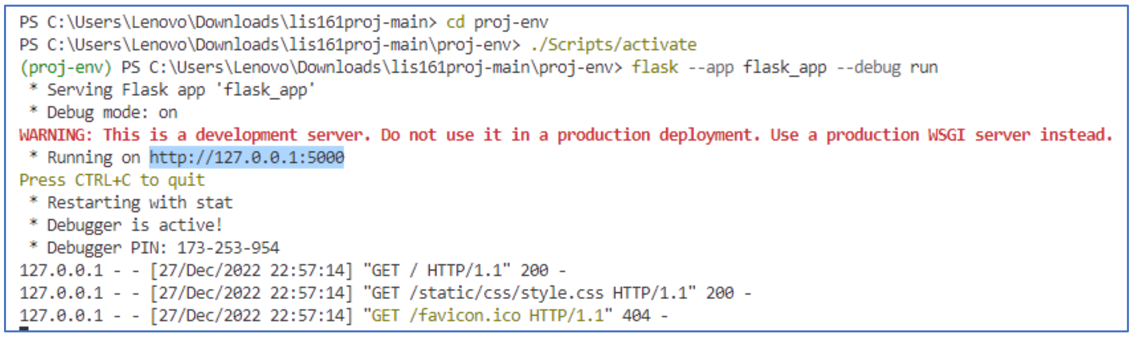
- Filename: style.css
  - This file contains the styling of some of the html elements for the program

- 

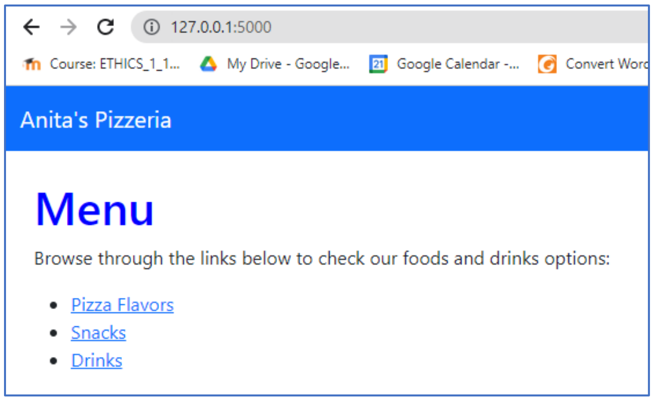
```
1   h1 {
2       color: blue;
3   }
4
5   img {
6           width: 30%;
7           height: auto;
8       }
9
10  .navbar-brand {
11      color: white;
12  }
```

- As explained in the base.html file, these will be applicable for all the pages of the web app.
- Lines 1-3: Setting the color of the h1 element to blue
- Lines 5-8: Setting the width of the image to 30% and its height to auto
- Lines 10-12: Setting the color of the navbar-brand to white. This is because the navbar's color is primary/blue. This makes the text pop out.

Project Requirements

- R1
  - Note: I downloaded the zip file from the github link https://github.com/ronaleane/lis161proj . I then opened the extracted folder in VS Code.
  - 

```
PS C:\Users\Lenovo\Downloads\lis161proj-main> cd proj-env
PS C:\Users\Lenovo\Downloads\lis161proj-main\proj-env> ./Scripts/activate
(proj-env) PS C:\Users\Lenovo\Downloads\lis161proj-main\proj-env> flask --app flask_app --debug run
 * Serving Flask app 'flask_app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 173-253-954
127.0.0.1 - - [27/Dec/2022 22:57:14] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [27/Dec/2022 22:57:14] "GET /static/css/style.css HTTP/1.1" 200 -
127.0.0.1 - - [27/Dec/2022 22:57:14] "GET /favicon.ico HTTP/1.1" 404 -
```

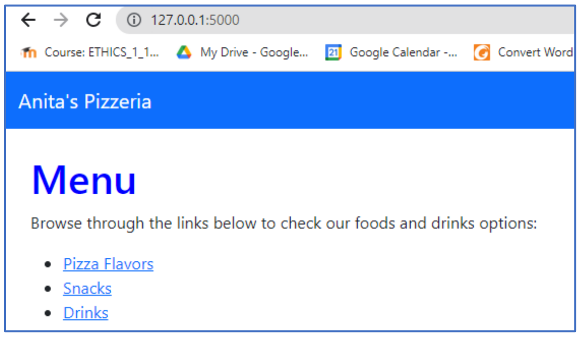

127.0.0.1:5000
Course: ETHICS_1_1... My Drive - Google... Google Calendar -... Convert Word

Anita's Pizzeria

Menu

Browse through the links below to check our foods and drinks options:

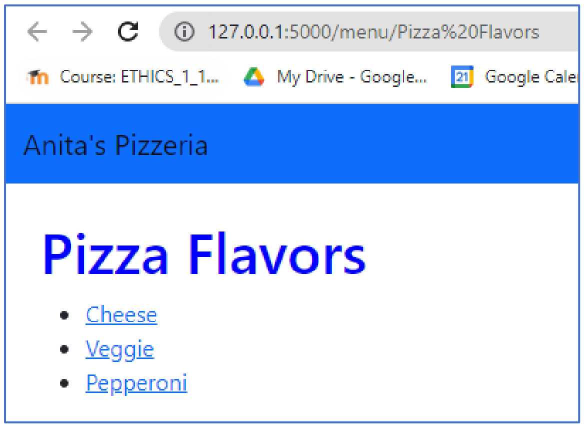
- Pizza Flavors
- Snacks
- Drinks

- R2
    - List view
        - 
          127.0.0.1:5000
          Course: ETHICS_1_1... My Drive - Google... Google Calendar -... Convert Word

          Anita's Pizzeria

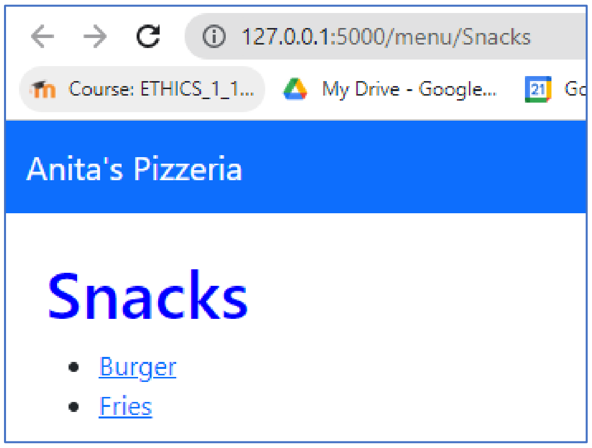
          Menu

          Browse through the links below to check our foods and drinks options:

          - Pizza Flavors
          - Snacks
          - Drinks
        - 
          127.0.0.1:5000/menu/Pizza%20Flavors
          Course: ETHICS_1_1... My Drive - Google... Google Cale

          Anita's Pizzeria

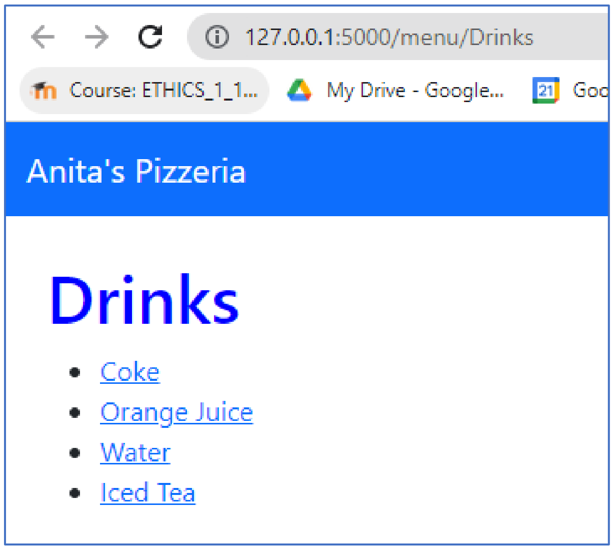
          Pizza Flavors

          - Cheese
          - Veggie
          - Pepperoni
        - 
          127.0.0.1:5000/menu/Snacks
          Course: ETHICS_1_1... My Drive - Google... Go

          Anita's Pizzeria

          Snacks

          - Burger
          - Fries



127.0.0.1:5000/menu/Drinks

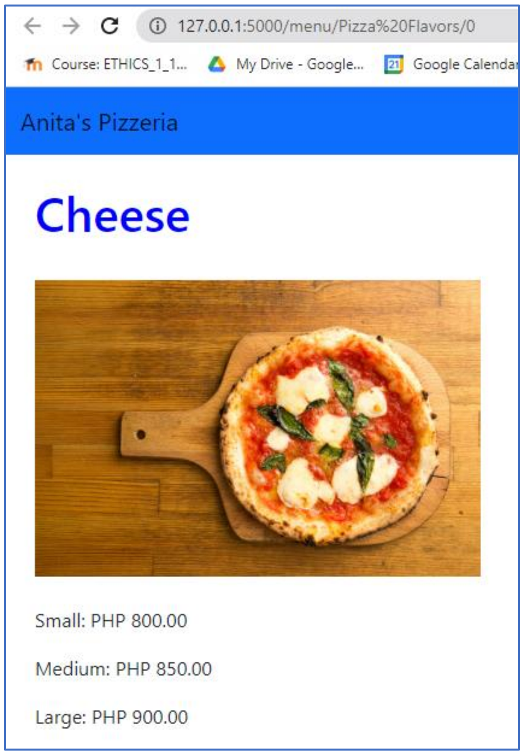
Course: ETHICS_1_1... My Drive - Google...

Anita's Pizzeria

Drinks

- Coke
- Orange Juice
- Water
- Iced Tea

- Item View



127.0.0.1:5000/menu/Pizza%20Flavors/0

Course: ETHICS_1_1... My Drive - Google... Google Calenda

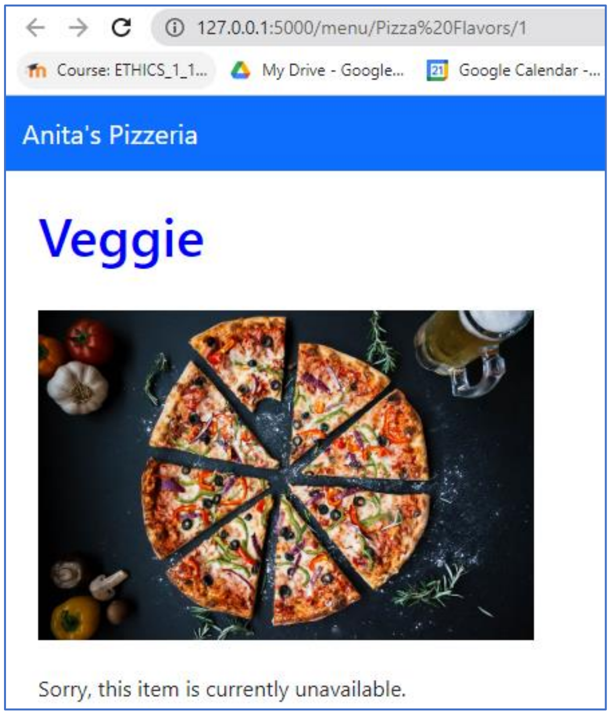
Anita's Pizzeria

Cheese

Small: PHP 800.00

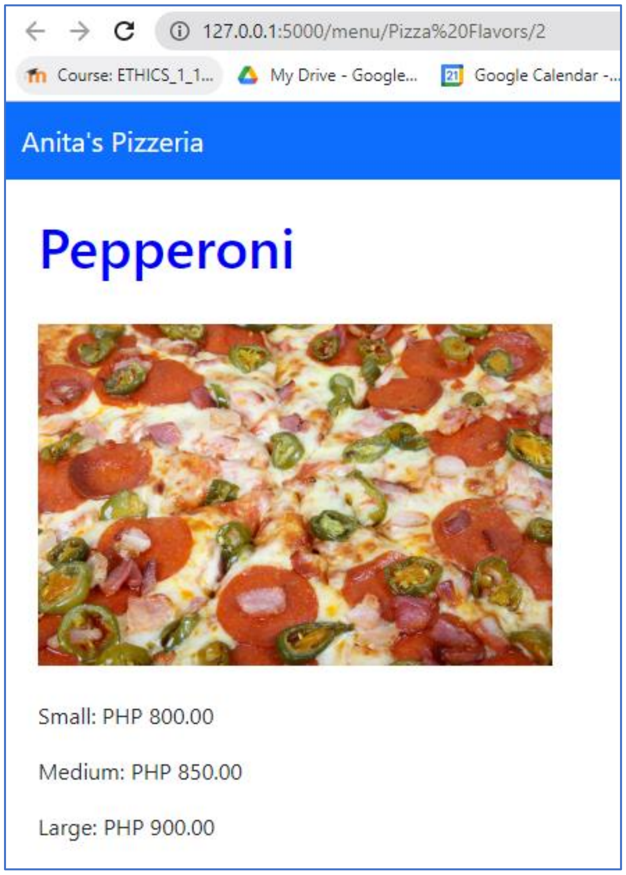
Medium: PHP 850.00

Large: PHP 900.00

- 
  127.0.0.1:5000/menu/Pizza%20Flavors/1

  Anita's Pizzeria

  Veggie

  Sorry, this item is currently unavailable.
- 
  127.0.0.1:5000/menu/Pizza%20Flavors/2

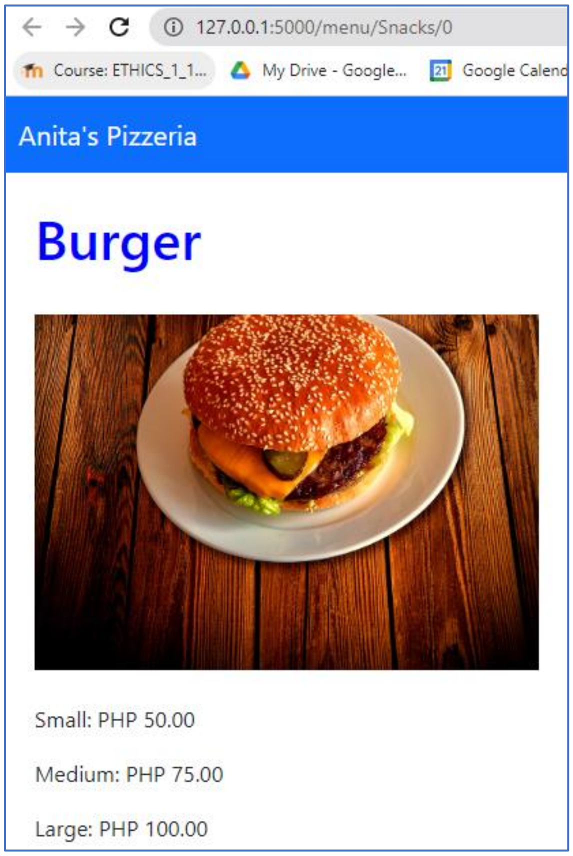
  Anita's Pizzeria

  Pepperoni

  Small: PHP 800.00

  Medium: PHP 850.00

  Large: PHP 900.00

- 

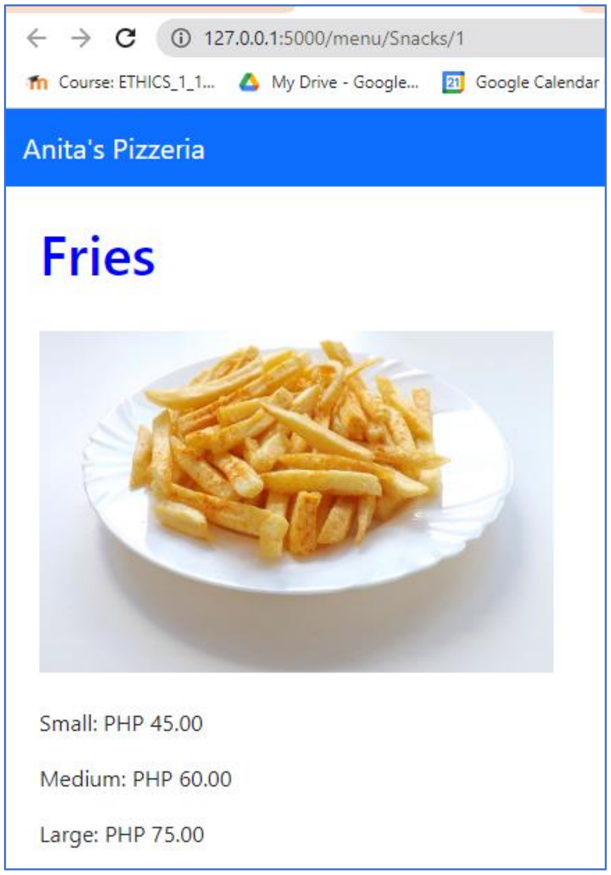
Anita's Pizzeria

Burger

Small: PHP 50.00

Medium: PHP 75.00

Large: PHP 100.00

- 

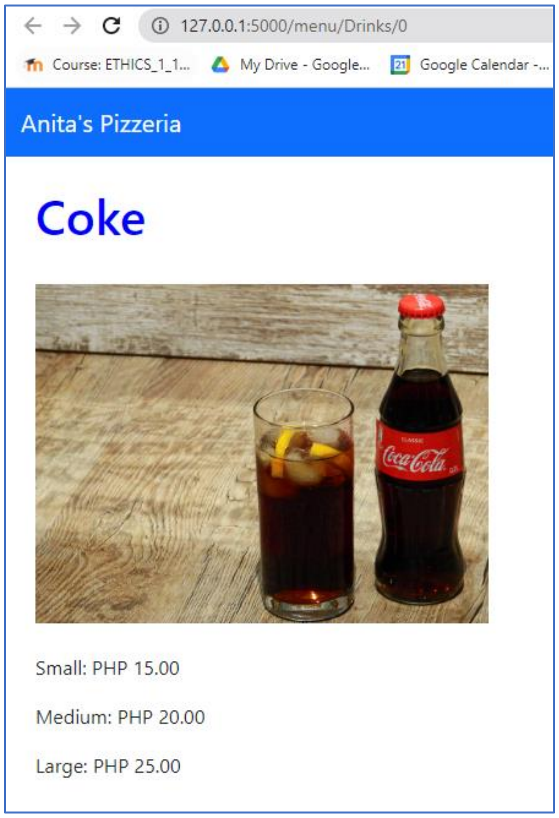
Anita's Pizzeria

Fries

Small: PHP 45.00

Medium: PHP 60.00

Large: PHP 75.00


127.0.0.1:5000/menu/Drinks/0

Course: ETHICS_1_1... My Drive - Google... Google Calendar -...

Anita's Pizzeria

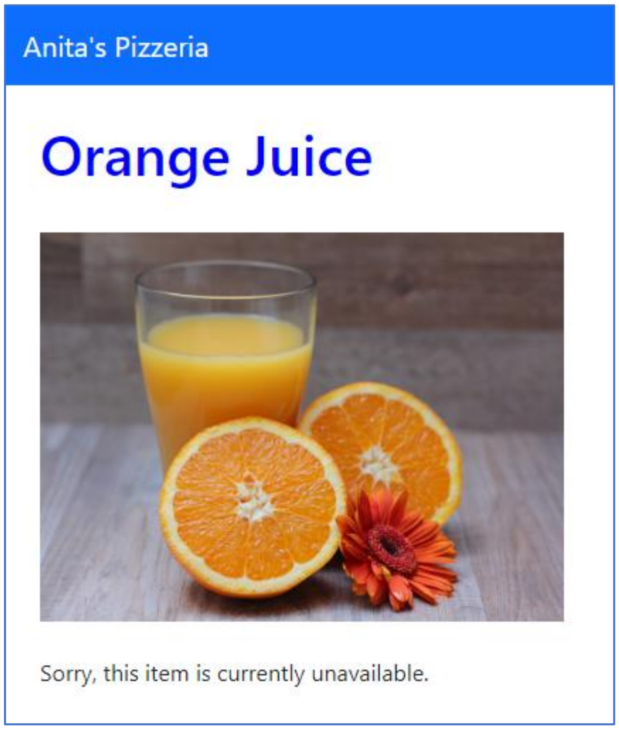
Coke

Small: PHP 15.00

Medium: PHP 20.00

Large: PHP 25.00

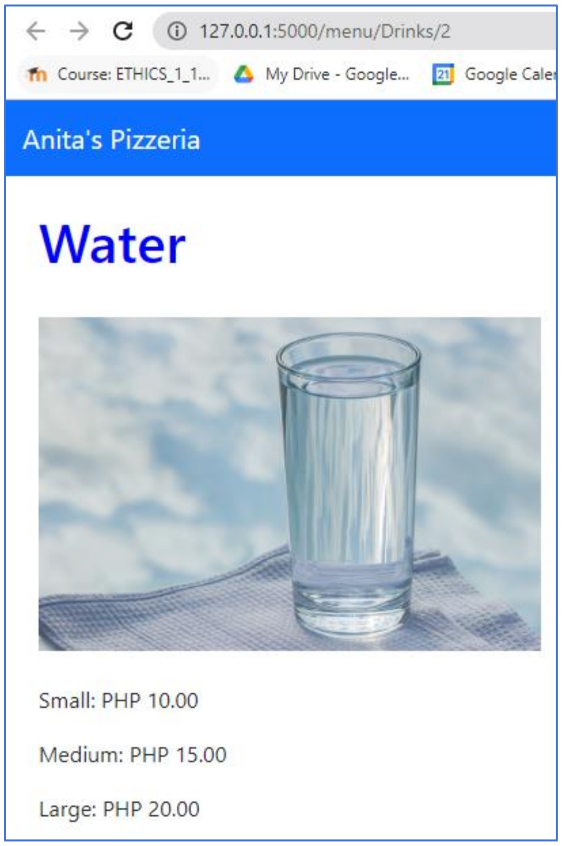
-


Anita's Pizzeria

Orange Juice

Sorry, this item is currently unavailable.

-

▪ 
127.0.0.1:5000/menu/Drinks/2

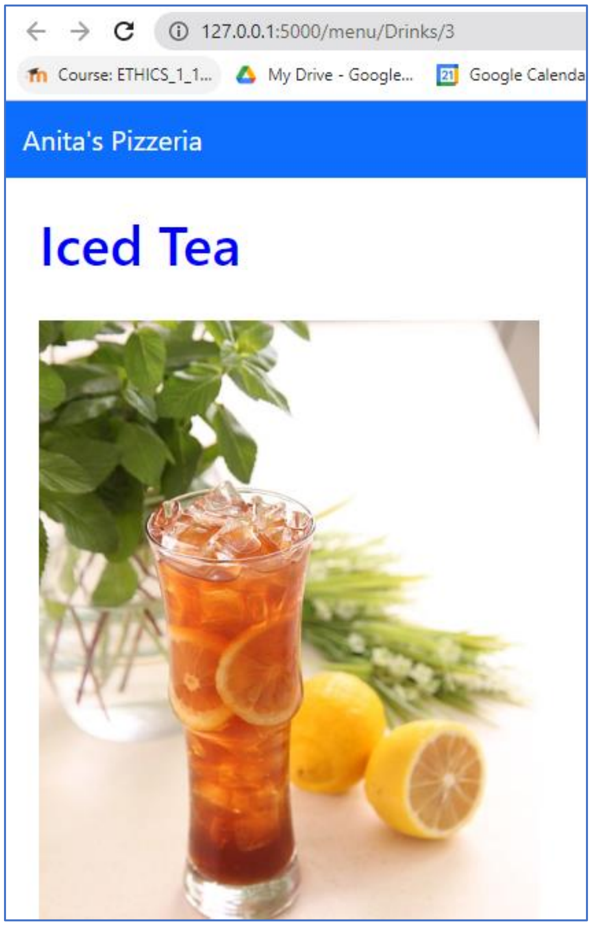
Anita's Pizzeria

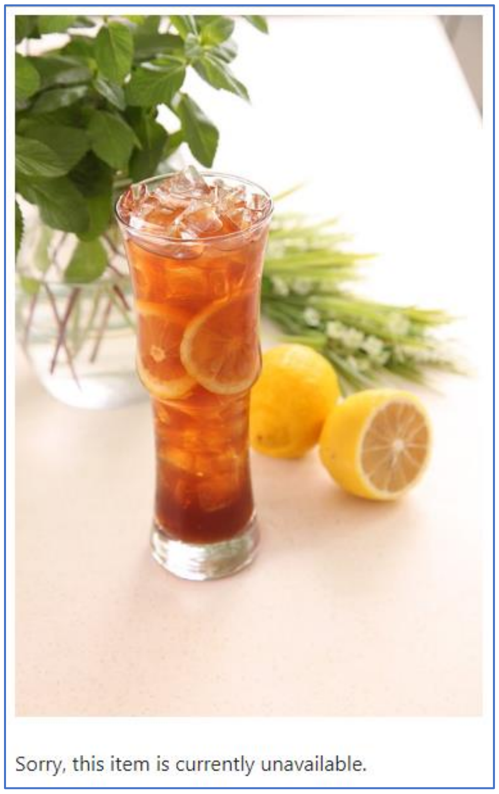
Water

Small: PHP 10.00

Medium: PHP 15.00

Large: PHP 20.00

- 
  127.0.0.1:5000/menu/Drinks/3

  Anita's Pizzeria
  Iced Tea
- 
  Sorry, this item is currently unavailable.

- R3
  - The navigation to all the pages (aside from the homepage) can be found in the list view.
  - The navigation to the homepage is via the navbar-brand.
- R4
  - The data is stored in data.py
- R5
  - Text type: str (for all the text that can be viewed in the website, as well as the str that can be found in data.py)
  - Numeric Type:
    - int (the value of the ‘small_price’, ‘medium_price’, and ‘large_price’ variables)
    - float (though not considered as float in the python file, as viewed in the website, the prices on the item view are shown as floats)
  - Sequence Type: list (the value paired with the keys ‘Pizza Flavors’, ‘Snacks’, and ‘Drinks’
  - Mapping Type: dict (the objects inside the list corresponding to the mentioned keys above, as well as the object ‘menu’ that contains all the data)
  - Boolean Type: True/False (value provided for the key ‘availability’)
  - Image (the images in the item view or the ones that can be accessed by following the specific_item[‘url’])
- R6
  - The ‘render_template()’ method and the templates in the ‘templates’ folder show that flask templating is employed.
- R7
  - A static file called ‘style.css’ is used.
  - The parent file is ‘base.html’ while the other templates except nav.html inherit the contents of the base file. The contents of ‘nav.html’ is included in ‘base.html’ which is also inherited by the other/child templates.