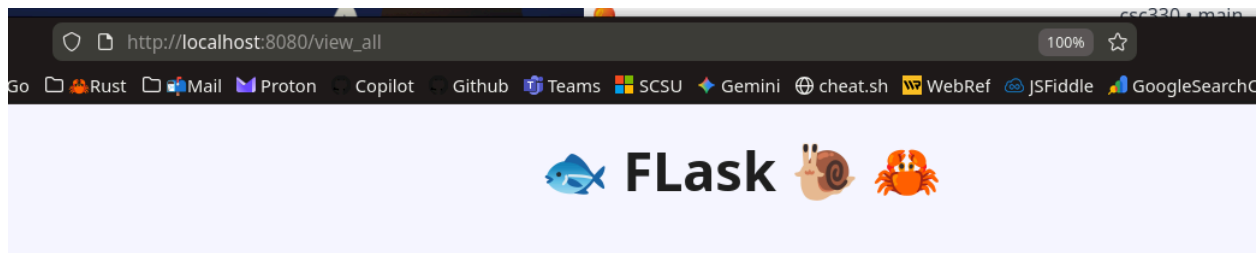


Keith Thomson

Activity 4 - Flask forms

After delete:



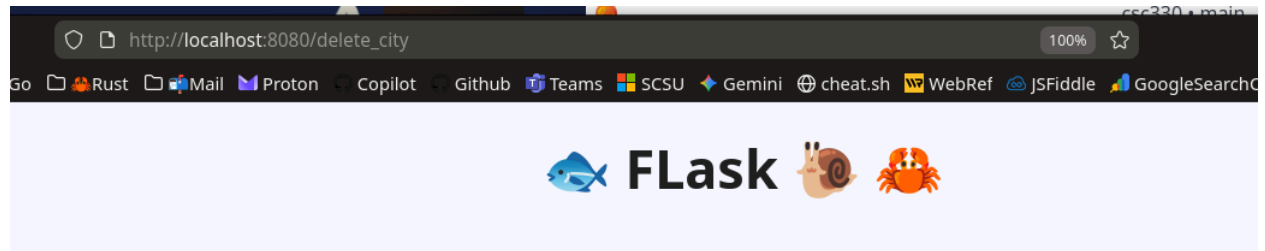
All Cities

City	Country	Population
San Francisco	United States	827,526
Newark	United States	758,344
Santiago	Chile	452,323

[Add More Cities](#) [Back to Home](#)

© 2026 Keith's Software Development Project

Github: [WhaleLogic](#)



Delete City

City Name:

Delete

[View All Cities](#) [Back to Home](#)

© 2026 Keith's Software Development Project

Github: [WhaleLogic](#)

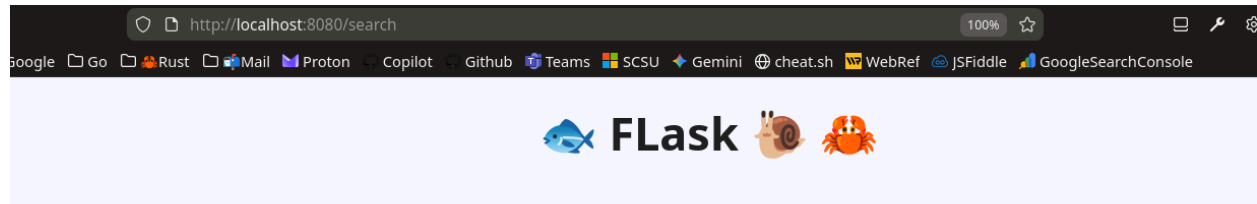
Google Docs - [Group Project Notes](#)

Logs

```
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:8080
* Running on http://192.168.5.12:8080
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 102-959-545
127.0.0.1 - - [22/Feb/2026 14:45:21] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [22/Feb/2026 14:45:25] "GET /view_all HTTP/1.1" 200 -
127.0.0.1 - - [22/Feb/2026 14:45:27] "GET /population HTTP/1.1" 200 -
127.0.0.1 - - [22/Feb/2026 14:45:39] "POST /population HTTP/1.1" 302 -
127.0.0.1 - - [22/Feb/2026 14:45:39] "GET /population HTTP/1.1" 200 -
127.0.0.1 - - [22/Feb/2026 14:45:54] "POST /population HTTP/1.1" 302 -
127.0.0.1 - - [22/Feb/2026 14:45:54] "GET /population HTTP/1.1" 200 -
127.0.0.1 - - [22/Feb/2026 14:46:42] "POST /population HTTP/1.1" 302 -
127.0.0.1 - - [22/Feb/2026 14:46:42] "GET /population HTTP/1.1" 200 -
127.0.0.1 - - [22/Feb/2026 14:47:14] "GET /view_all HTTP/1.1" 200 -
127.0.0.1 - - [22/Feb/2026 14:47:20] "GET /population HTTP/1.1" 200 -
127.0.0.1 - - [22/Feb/2026 14:47:40] "POST /population HTTP/1.1" 302 -
127.0.0.1 - - [22/Feb/2026 14:47:40] "GET /population HTTP/1.1" 200 -
127.0.0.1 - - [22/Feb/2026 14:47:53] "POST /population HTTP/1.1" 302 -
127.0.0.1 - - [22/Feb/2026 14:47:53] "GET /population HTTP/1.1" 200 -
127.0.0.1 - - [22/Feb/2026 14:48:10] "GET /view_all HTTP/1.1" 200 -
127.0.0.1 - - [22/Feb/2026 14:48:14] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [22/Feb/2026 14:48:16] "GET /delete_city HTTP/1.1" 200 -
127.0.0.1 - - [22/Feb/2026 14:49:07] "POST /delete_city HTTP/1.1" 302 -
127.0.0.1 - - [22/Feb/2026 14:49:07] "GET /view_all HTTP/1.1" 200 -
127.0.0.1 - - [22/Feb/2026 14:50:18] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [22/Feb/2026 14:50:19] "GET /delete_city HTTP/1.1" 200 -
127.0.0.1 - - [22/Feb/2026 14:50:24] "POST /delete_city HTTP/1.1" 302 -
127.0.0.1 - - [22/Feb/2026 14:50:24] "GET /view_all HTTP/1.1" 200 -
127.0.0.1 - - [22/Feb/2026 14:50:34] "GET /view_all HTTP/1.1" 200 -
127.0.0.1 - - [22/Feb/2026 14:51:21] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [22/Feb/2026 14:51:23] "GET /delete_city HTTP/1.1" 200 -
127.0.0.1 - - [22/Feb/2026 14:51:39] "POST /delete_city HTTP/1.1" 302 -
127.0.0.1 - - [22/Feb/2026 14:51:39] "GET /view_all HTTP/1.1" 200 -
```

python3.14 + Sun Feb 22 14:52

/search



Search for City Population

City Name:

Search

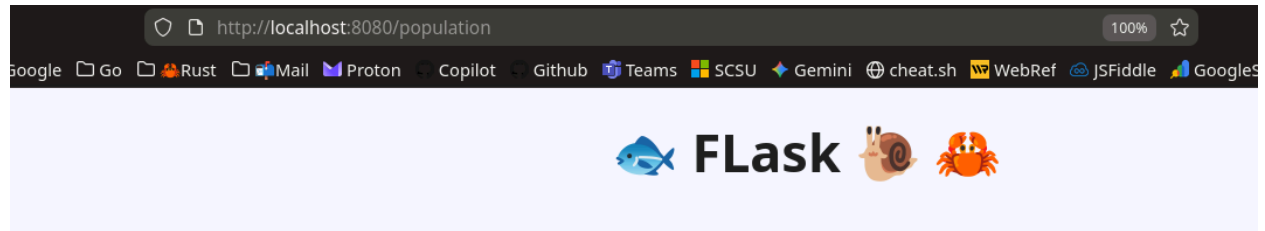
[Add More Cities](#) [Back to Home](#)

© 2026 Keith's Software Development Project

Github: [WhaleLogic](#)

Google Docs - [Group Project Notes](#)

/population



City Population Form

Add City

City:

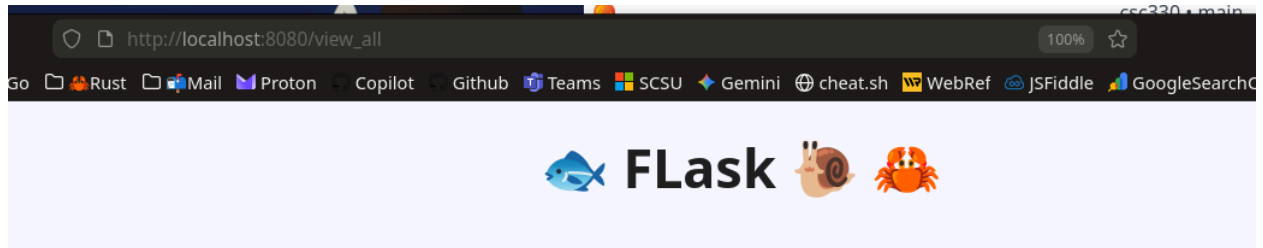
Population:

Country:

Save

[Back to Home](#) [View All Cities](#)

/view_all

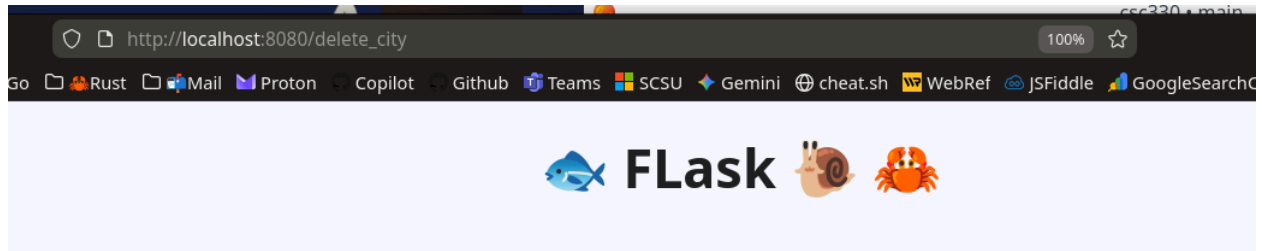


All Cities

City	Country	Population
Boston	Unites States	3,422,334
San Fransicso	United States	827,526
Newark	United States	758,344
Santiago	Chile	452,323

[Add More Cities](#) [Back to Home](#)

/delete_city



Delete City

City Name:

Delete

[View All Cities](#) [Back to Home](#)

© 2026 Keith's Software Development Project

Github: [WhaleLogic](#)

Google Docs - [Group Project Notes](#)

view_cities.html

```
forms.py X routes.py X population.htm X search.html X base.html X view_cities.ht X
1 {% extends "base.html" %}
2
3 {% block title %}View All Cities{% endblock %}
4
5 {% block content %}
6 <div class="container">
7   <h1>All Cities</h1>
8
9   {% if cities %}
10    <table>
11      <thead>
12        <tr>
13          <th>City</th>
14          <th>Country</th>
15          <th>Population</th>
16        </tr>
17      </thead>
18      <tbody>
19        {% for city in cities %}
20          <tr>
21            <td>{{ city.name }}</td>
22            <td>{{ city.country }}</td>
23            <td>{{ "{:,}".format(city.population) }}</td>
24          </tr>
25        {% endfor %}
26      </tbody>
27    </table>
28  {% else %}
29    <p style="text-align: center; color: #666; margin: 20px 0;">No cities added yet. Go to the Population form to add cities.</p>
30  {% endif %}
31
32  <div class="content-links">
33    <a href="{{ url_for('population') }}">Add More Cities</a>
34    <a href="{{ url_for('home') }}">Back to Home</a>
35  </div>
36 </div>
37 {% endblock %}
```

1 %> main

population.html

```
forms.py X routes.py X population.htm X search.html X base.html X view_cities.ht X
7 <h1>City Population Form</h1>
8
9 <h2>Add City</h2>
10 <form method="POST" action="{{ url_for('population') }}">
11     {{ form.hidden_tag() }}
12
13     <div>
14         {{ form.city.label }}
15         {{ form.city(placeholder="Enter city name") }}
16         {% if form.city.errors %}
17             <ul class="errors">
18                 {% for error in form.city.errors %}
19                     <li>{{ error }}</li>
20                 {% endfor %}
21             </ul>
22         {% endif %}
23     </div>
24
25     <div>
26         {{ form.population.label }}
27         {{ form.population(placeholder="Enter population") }}
28         {% if form.population.errors %}
29             <ul class="errors">
30                 {% for error in form.population.errors %}
31                     <li>{{ error }}</li>
32                 {% endfor %}
33             </ul>
34         {% endif %}
35     </div>
36
37     <div>
38         {{ form.country.label }}
39         {{ form.country(placeholder="Enter country name") }}
40         {% if form.country.errors %}
41             <ul class="errors">
42                 {% for error in form.country.errors %}
43                     <li>{{ error }}</li>
44                 {% endfor %}
45             </ul>
46         {% endif %}
47     </div>
48 </div>
main
```

Search.html

```

forms.py X routes.py X population.htm X search.html X base.html X view cities.ht X
1 {% extends "base.html" %}
2
3 {% block title %}Search Cities{% endblock %}
4
5 {% block content %}
6 <div class="container">
7   <h1>Search for City Population</h1>
8
9   <form method="POST" action="{{ url_for('search') }}">
10     {{ form.hidden_tag() }}
11
12     <div>
13       {{ form.city.label }}
14       {{ form.city(placeholder="Enter city name") }}
15       {% if form.city.errors %}
16         <ul class="errors">
17           {% for error in form.city.errors %}
18             <li>{{ error }}</li>
19           {% endfor %}
20         </ul>
21       {% endif %}
22     </div>
23
24     <div>
25       {{ form.submit() }}
26     </div>
27   </form>
28
29   {% if city %}
30     <div class="content-card">
31       <h2>Search Results</h2>
32       <table>
33         <thead>
34           <tr>
35             <th>City</th>
36             <th>Country</th>
37             <th>Population</th>
38           </tr>
39         </thead>
40         <tbody>
41           <tr>
42             <td>{{ city.name }}</td>
43             <td>{{ city.country }}</td>
44             <td>{{ "{:,}".format(city.population) }}</td>
45           </tr>
46         </tbody>
47       </table>
48     </div>
49   {% elif request.method == 'POST' %}
50     <p style="text-align: center; color: #666; margin: 20px 0;">No city found with that name.</p>
51   {% endif %}
52
53   <div class="content-links">
54     <a href="{{ url_for('population') }}">Add More Cities</a>
55     <a href="{{ url_for('home') }}">Back to Home</a>
56   </div>
57 </div>
in

```

Source Code

forms.py

```
from flask_wtf import FlaskForm
from wtforms import StringField, IntegerField, SubmitField
from wtforms.validators import DataRequired

class PopulationForm(FlaskForm):
    city = StringField('City:', validators=[DataRequired()])
    country = StringField('Country:', validators=[DataRequired()])
    population = IntegerField('Population: ',
validators=[DataRequired()])
    submit = SubmitField('Save')

class SearchForm(FlaskForm):
    city = StringField('City Name:', validators=[DataRequired()])
    submit = SubmitField('Search')

class DeleteCityForm(FlaskForm):
    city = StringField('City Name:', validators=[DataRequired()])
    submit = SubmitField('Delete')
```

routes.py

```
from flask import render_template, redirect, url_for, request, flash
from app.forms import PopulationForm, SearchForm, DeleteCityForm
from app import cities, database
from app.city import City
import os
import requests
import random
```

There is an SQLite database version that is currently commented out. The app runs using an in-memory version right now that deletes the city objects as like the assignment requires.

```
CITIES = ["San Diego", "New York", "Miami", "Las Vegas", "Seattle", "Denver", "New Haven"]
```

```
def get_weather_data():
    # This function uses the CITIES list above and is separate from
    the City class and database code. It is used in the /temps and
    /plot_temps routes to get weather data from the Open weather API.
    api_key = os.getenv('WEATHER_API_KEY')
    results = {}
    for city in CITIES:
        url =
f'http://api.openweathermap.org/data/2.5/weather?q={city}&appid={api_k
ey}&units=imperial'
        try:
            response = requests.get(url).json()
            results[city] = response['main']['temp']
        except:
            results[city] = 0
    return results
```

```
def init_app(app):
    @app.route('/')
    def home():
        return render_template('index.html', title="Keith's Flask
App")
```

```
@app.route('/temps')
def temps():
    data = get_weather_data()
    return render_template('temps.html', weather_data=data)
```

```

@app.route('/plot_temps')
def plot_temps():
    data = get_weather_data()
    return render_template('chart.html',
                           labels=list(data.keys()),
                           values=list(data.values()))

@app.route('/fruits')
def fruits():
    fruits = ['Apple', 'Banana', 'Cherry', 'Dragonfruit']
    r = random.choice(fruits)
    return f"Random fruit: {r}"

@app.route('/population', methods=['GET', 'POST'])
def population():
    form = PopulationForm()

    if form.validate_on_submit():
        city_name = form.city.data
        population_count = form.population.data
        country_name = form.country.data

        # In-memory version
        city_obj = City(city_name, population_count, country_name)
        cities.append(city_obj)

        # Database version (comment out for in-memory)
        # database.insert_city(city_name, population_count,
country_name)

    return redirect(url_for('population'))

```



```
        return render_template('population.html', form=form,
title="Population Dashboard")
```

```
@app.route('/view_all')
def view_all():
    # In-memory version
    city_objects = cities

    # Database version (comment out to use in-memory version)
    # db_cities = database.get_all_cities()
    # city_objects = [City(row['city'], row['population'],
row['country'] if row['country'] else '') for row in db_cities]

    return render_template('view_cities.html',
cities=city_objects)
```

```
@app.route('/search', methods=['GET', 'POST'])
def search():
    form = SearchForm()
    city_found = None

    if form.validate_on_submit():
        search_name = form.city.data

        # In-memory version (uncomment for in-memory)
        for city in cities:
            if city.name.lower() == search_name.lower():
                city_found = city
                break

        # Database version (comment out for in-memory)
        # db_cities = database.get_all_cities()
        # for row in db_cities:
        #     if row['city'].lower() == search_name.lower():
```

```

        #             city_found = City(row['city'],
row['population'], row['country'] if row['country'] else '')
        #             break

    return render_template('search.html', form=form,
city=city_found)

@app.route('/delete_city', methods=['GET', 'POST'])
def delete_city():
    form = DeleteCityForm()

    if form.validate_on_submit():
        city_name = form.city.data

        # In-memory version (uncomment for in-memory)
        for i, city in enumerate(cities):
            if city.name.lower() == city_name.lower():
                cities.pop(i)
                break

        # Database version (comment out for in-memory)
        # db_cities = database.get_all_cities()
        # for row in db_cities:
        #     if row['city'].lower() == city_name.lower():
        #         database.delete_city(row['id'])
        #         break

    return redirect(url_for('view_all'))

return render_template('delete_city.html', form=form)

```