

# Flask Deployment Web App Document

Name: Konstantinos Soufleros

Batch Code: LISUM30

Submission Date: 28/02/2024

Submitted To: GitHub

## 1. my\_flask\_app.py

```
from flask import Flask, render_template, request
import numpy as np
import pandas as pd
import pickle
from sklearn.preprocessing import StandardScaler

app = Flask(__name__)

# Set the static folder path
app.config['STATIC_FOLDER'] = 'static'

# Load the scaler
with open('scaler.pkl', 'rb') as scaler_file:
    scaler = pickle.load(scaler_file)

# Load the model
with open('gradient_boosting_model.pkl', 'rb') as model_file:
    model = pickle.load(model_file)

# Define the prediction route
@app.route('/', methods=['GET', 'POST'])
def predict():
    if request.method == 'POST':
        # Get the form data
        property_type = request.form['property_type']
        neighborhood = request.form['neighborhood']
        property_sqft = float(request.form['property_sqft'])
        bedrooms = int(request.form['bedrooms'])
        baths = int(request.form['baths'])

        # Load the column names of the encoded features
```

```

        with open('encoded_columns.pkl', 'rb') as f:
            encoded_columns = pickle.load(f)

        # Perform one-hot encoding for categorical variables
        type_encoded =
pd.get_dummies(pd.Series(property_type)).reindex(columns=encoded_columns, fill_value=0)
        sublocality_encoded =
pd.get_dummies(pd.Series(neighborhood)).reindex(columns=encoded_columns, fill_value=0)

        # Concatenate encoded features
        encoded_features = pd.concat([type_encoded,
sublocality_encoded], axis=1)

        # Scale the entire feature vector (both numerical and encoded
categorical features)
        input_features = np.array([[property_sqft, bedrooms,
baths]]) # Numerical features
        input_features_scaled =
scaler.transform(np.concatenate([input_features,
encoded_features.values], axis=1))

        # Make the prediction
        prediction = model.predict(input_features_scaled)
        output = round(prediction[0], 2)

        # Display the prediction result
        return render_template('index.html', prediction_text='House
price should be ${} '.format(output))

    return render_template('index.html')

if __name__ == '__main__':
    app.run(port=5000, debug=False)

```

## 2. index.html

```
3. <!DOCTYPE html>
4. <html lang="en">
5. <head>
6.     <meta charset="UTF-8">
7.     <meta name="viewport" content="width=device-width, initial-
      scale=1.0">
8.     <title>House Price Estimation in New York</title>
9.     <style>
10.         body {
11.             background-image: url('{{ url_for("static",
      filename="55443.jpg") }}');
12.             background-size: cover;
13.             background-repeat: no-repeat;
14.             background-attachment: fixed;
15.             display: flex;
16.             justify-content: center;
17.             align-items: center;
18.             height: 100vh;
19.             margin: 0;
20.         }
21.         .form-container {
22.             background-color: rgba(255, 255, 255, 0.8);
23.             padding: 20px;
24.             border-radius: 10px;
25.         }
26.     </style>
27. </head>
28. <body>
29.     <div class="form-container">
30.         <h1 style="text-align: center; color: #333;">House Price
      Estimation in New York</h1>
31.         <form id="prediction-form" style="text-align: center;">
32.             <label for="property-type">Type of Property:</label>
33.             <select id="property-type"
      name="property_type">
34.                 <option value="Condo for sale">Condo for sale</option>
35.                 <option value="House for sale">House for sale</option>
36.                 <option value="Townhouse for sale">Townhouse for
      sale</option>
37.                 <option value="Co-op for sale">Co-op for sale</option>
38.                 <option value="Multi-family home for sale">Multi-family
      home for sale</option>
39.                 <option value="For sale">For sale</option>
40.                 <option value="Contingent">Contingent</option>
41.                 <option value="Land for sale">Land for sale</option>
42.                 <option value="Foreclosure">Foreclosure</option>
43.             </select>

```

```
44.         <option value="Pending">Pending</option>
45.         <option value="Coming Soon">Coming Soon</option>
46.         <option value="Mobile house for sale">Mobile house for
sale</option>
47.         <option value="Condop for sale">Condop for sale</option>
48.     </select>
49.     <br><br>
50.     <label for="neighborhood">Neighborhood:</label>
51.     <select id="neighborhood" name="neighborhood">
52.         <option value="New York">New York</option>
53.         <option value="Staten Island">Staten Island</option>
54.         <option value="Manhattan">Manhattan</option>
55.         <option value="Brooklyn">Brooklyn</option>
56.         <option value="Bronx">Bronx</option>
57.         <option value="Jackson Heights">Jackson Heights</option>
58.         <option value="Elmhurst">Elmhurst</option>
59.         <option value="Woodside">Woodside</option>
60.         <option value="Rego Park">Rego Park</option>
61.         <option value="Forest Hills">Forest Hills</option>
62.         <option value="Briarwood">Briarwood</option>
63.         <option value="Queens">Queens</option>
64.         <option value="Flushing">Flushing</option>
65.         <option value="Woodhaven">Woodhaven</option>
66.         <option value="Jamaica">Jamaica</option>
67.         <option value="Richmond Hill South">Richmond Hill
South</option>
68.         <option value="Whitestone">Whitestone</option>
69.         <option value="Ridgewood">Ridgewood</option>
70.         <option value="Rosedale">Rosedale</option>
71.         <option value="Ozone Park">Ozone Park</option>
72.         <option value="Springfield Gardens">Springfield
Gardens</option>
73.         <option value="Far Rockaway">Far Rockaway</option>
74.         <option value="Bellerose">Bellerose</option>
75.         <option value="Bayside">Bayside</option>
76.         <option value="Astoria">Astoria</option>
77.         <option value="Floral Park">Floral Park</option>
78.         <option value="Fresh Meadows">Fresh Meadows</option>
79.         <option value="Howard Beach">Howard Beach</option>
80.         <option value="Cambria Heights">Cambria Heights</option>
81.         <option value="Rockaway Park">Rockaway Park</option>
82.         <option value="East Elmhurst">East Elmhurst</option>
83.         <option value="Little Neck">Little Neck</option>
84.         <option value="Long Island City">Long Island City</option>
85.         <option value="Corona">Corona</option>
86.         <option value="Kew Gardens">Kew Gardens</option>
87.         <option value="Brownville">Brownville</option>
88.         <option value="Brooklyn Heights">Brooklyn Heights</option>
```

```

89.         <option value="Maspeth">Maspeth</option>
90.         <option value="Queens Village">Queens Village</option>
91.         <option value="Richmond Hill">Richmond Hill</option>
92.         <option value="Middle Village">Middle Village</option>
93.         <option value="Saint Albans">Saint Albans</option>
94.         <option value="Kensington">Kensington</option>
95.         <option value="South Ozone Park">South Ozone Park</option>
96.         <option value="Canarsie">Canarsie</option>
97.         <option value="College Point">College Point</option>
98.         <option value="Bedford Stuyvesant">Bedford
    Stuyvesant</option>
99.         <option value="Hollis">Hollis</option>
100.        <option value="Malba">Malba</option>
101.        <option value="Glen Oaks">Glen Oaks</option>
102.        <option value="Douglaston">Douglaston</option>
103.        <option value="Sunnyside">Sunnyside</option>
104.        <option value="Arverne">Arverne</option>
105.        <option value="Glendale">Glendale</option>
106.        <option value="New Hyde Park">New Hyde Park</option>
107.        <option value="Crown Heights">Crown Heights</option>
108.        <option value="Old Mill Basin">Old Mill
    Basin</option>
109.        <option value="Beechhurst">Beechhurst</option>
110.        <option value="Roosevelt Island">Roosevelt
    Island</option>
111.        <option value="Kew Gardens Hills">Kew Gardens
    Hills</option>
112.        <option value="Stuyvesant Heights">Stuyvesant
    Heights</option>
113.        <option value="Belle Harbor">Belle Harbor</option>
114.        <option value="East Flatbush">East Flatbush</option>
115.        <option value="Kew Garden Hills">Kew Garden
    Hills</option>
116.        <option value="Ditmas Park">Ditmas Park</option>
117.        <option value="Brighton Beach">Brighton
    Beach</option>
118.        <option value="Prospect Lefferts Gardens">Prospect
    Lefferts Gardens</option>
119.        </select>
120.        <br><br>
121.        <label for="property-sqft">Property Sqft:</label>
122.        <input type="number" id="property-sqft"
    name="property_sqft" min="0">
123.        <br><br>
124.        <label for="bedrooms">Bedrooms:</label>
125.        <input type="number" id="bedrooms" name="bedrooms"
    min="0">
126.        <br><br>

```

```

127.         <label for="baths">Baths:</label>
128.         <input type="number" id="baths" name="baths" min="0">
129.         <br><br>
130.         <button type="submit">Compute Estimated
    Price</button>
131.     </form>
132.     <!-- Display prediction result here -->
133.     <div id="prediction-result"></div>
134. </div>
135.
136.     <script>
137.         const form = document.getElementById('prediction-form');
138.
139.         form.addEventListener('submit', (event) => {
140.             event.preventDefault();
141.
142.             // Get the form data
143.             const propertyType =
    document.getElementById('property-type').value;
144.             const neighborhood =
    document.getElementById('neighborhood').value;
145.             const propertySqft =
    document.getElementById('property-sqft').value;
146.             const bedrooms =
    document.getElementById('bedrooms').value;
147.             const baths = document.getElementById('baths').value;
148.
149.             // Send the data to the server using a POST request
150.             fetch('/predict', {
151.                 method: 'POST',
152.                 headers: {
153.                     'Content-Type': 'application/json'
154.                 },
155.                 body: JSON.stringify({
156.                     property_type: propertyType,
157.                     neighborhood: neighborhood,
158.                     property_sqft: propertySqft,
159.                     bedrooms: bedrooms,
160.                     baths: baths
161.                 })
162.             })
163.             .then(response => response.json())
164.             .then(data => {
165.                 // Display the prediction result
166.                 document.getElementById('prediction-
    result').innerHTML = data.prediction_text;
167.             });
168.         });

```

```

169.
170.         </script>
171.     </body>
172. </html>

```

### 3. Result



