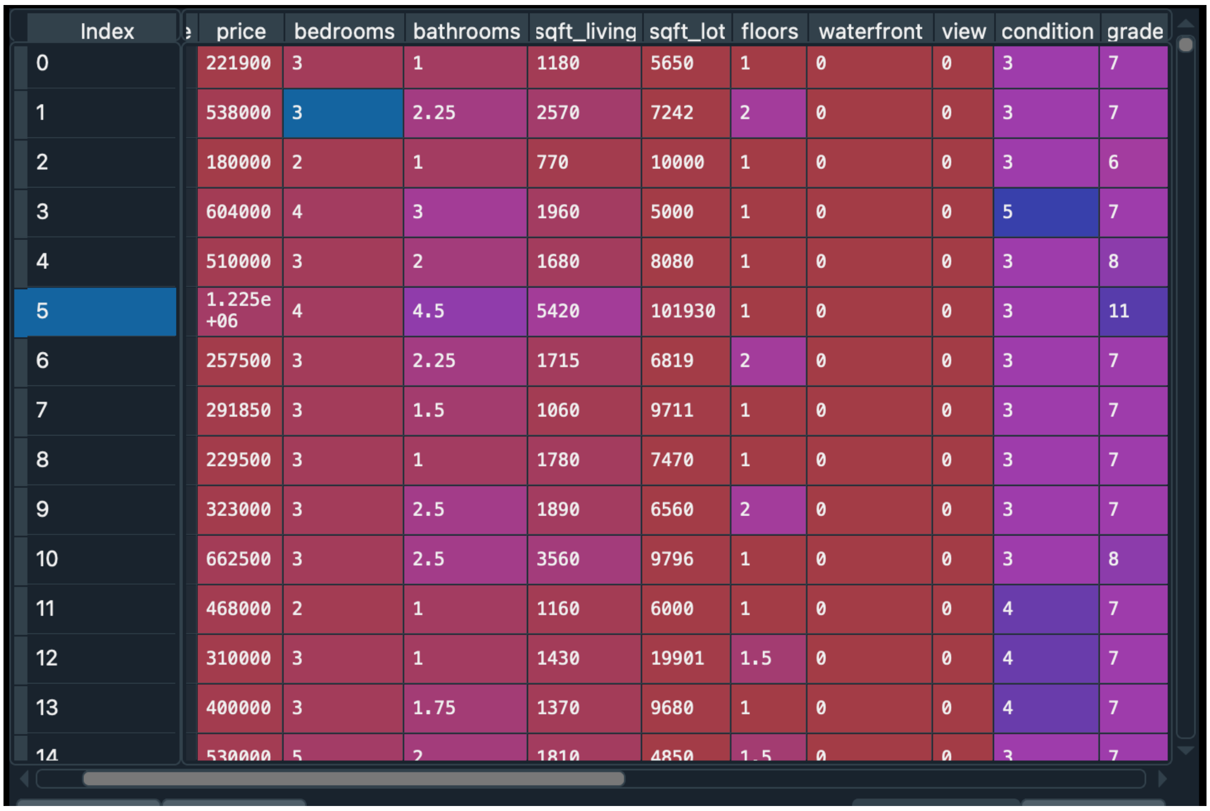
Name: İrem Tanrıverdi
Batch code: LISP01
Submission date: 20/03/2021
Submitted to: Data Glacier

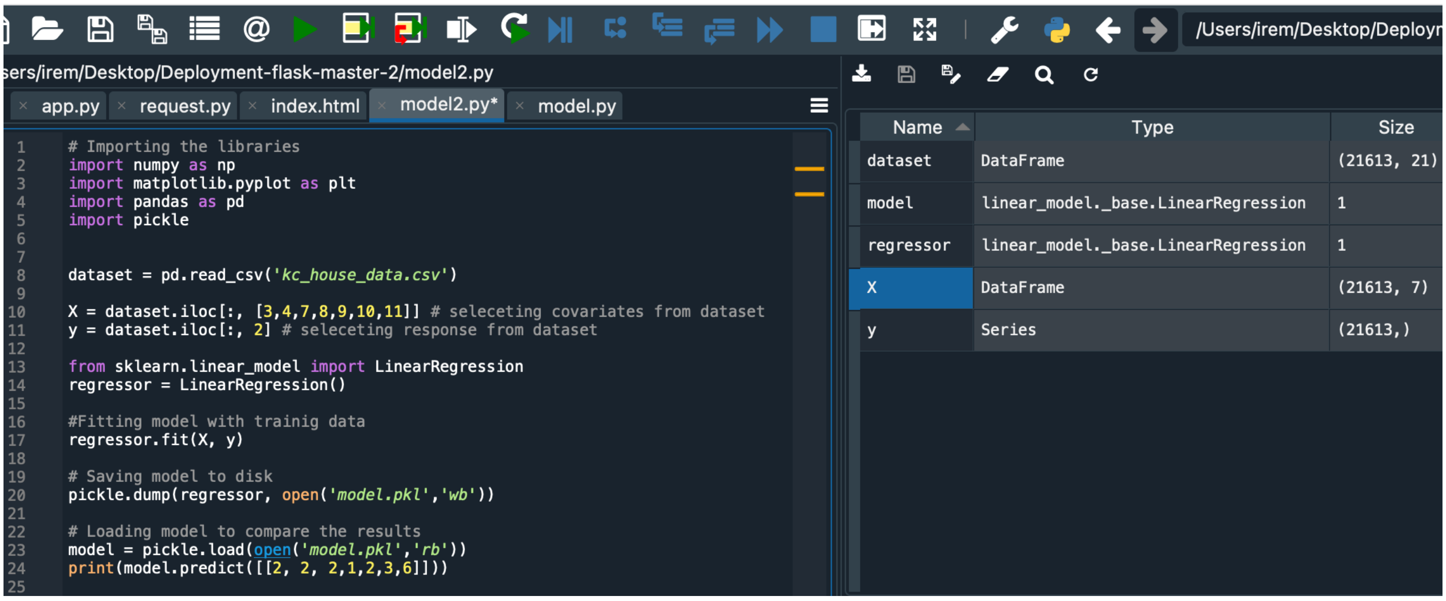
Name of the dataset is "kc_house_data.csv" taken from Kaggle. This dataset includes information about characteristics of the houses like how many bathrooms, bedroom, etc. and what is the price of these house.



Index	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	view	condition	grade
0	221900	3	1	1180	5650	1	0	0	3	7
1	538000	3	2.25	2570	7242	2	0	0	3	7
2	180000	2	1	770	10000	1	0	0	3	6
3	604000	4	3	1960	5000	1	0	0	5	7
4	510000	3	2	1680	8080	1	0	0	3	8
5	1.225e+06	4	4.5	5420	101930	1	0	0	3	11
6	257500	3	2.25	1715	6819	2	0	0	3	7
7	291850	3	1.5	1060	9711	1	0	0	3	7
8	229500	3	1	1780	7470	1	0	0	3	7
9	323000	3	2.5	1890	6560	2	0	0	3	7
10	662500	3	2.5	3560	9796	1	0	0	3	8
11	468000	2	1	1160	6000	1	0	0	4	7
12	310000	3	1	1430	19901	1.5	0	0	4	7
13	400000	3	1.75	1370	9680	1	0	0	4	7
14	530000	5	2	1810	4850	1.5	0	0	3	7

This is the snapshot of the dataset.

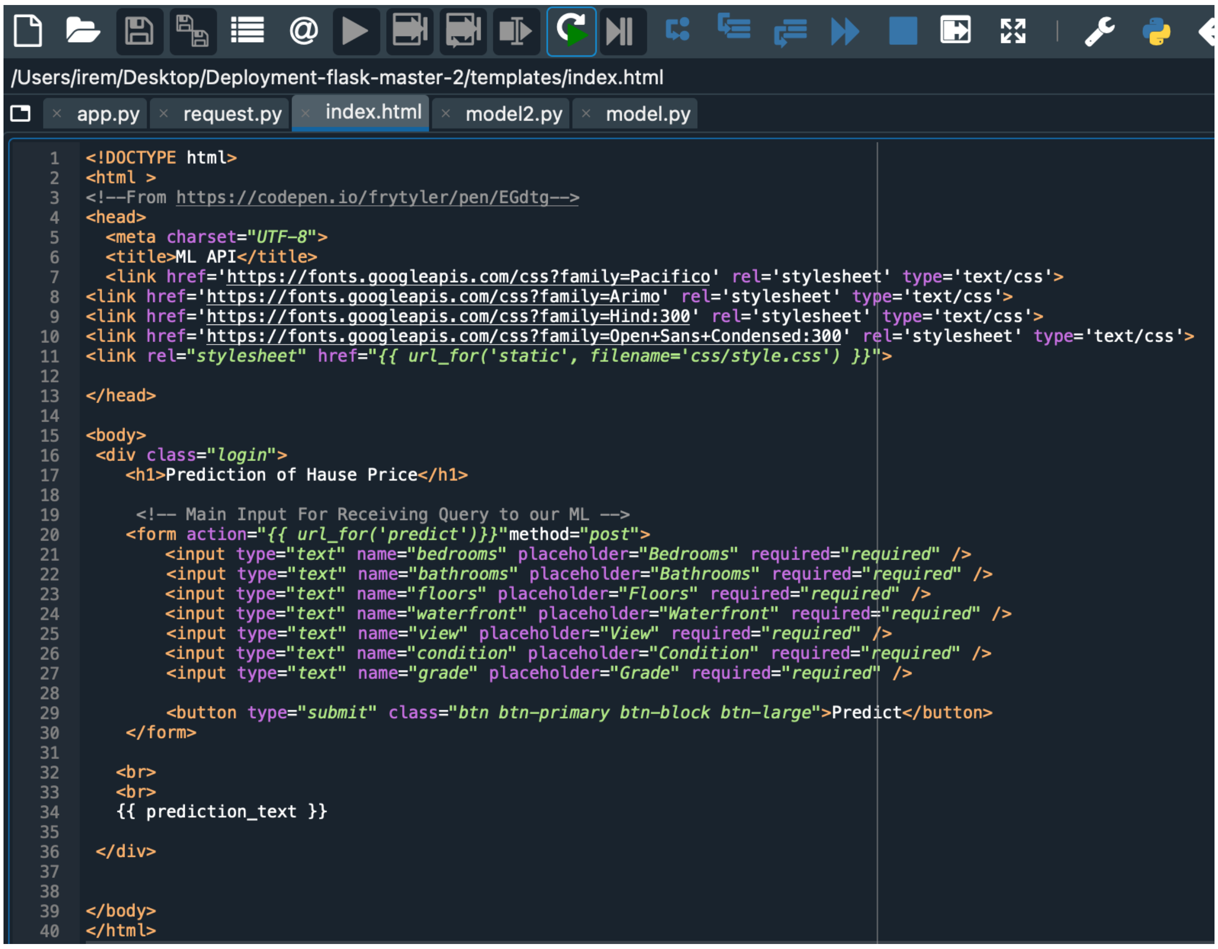
Waterfront: There is waterfront=1 ; There is not waterfront=0


/Users/irem/Desktop/Deployn
sers/irem/Desktop/Deployment-flask-master-2/model2.py
app.py
request.py
index.html
model2.py*
model.py

```python
# Importing the libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import pickle


dataset = pd.read_csv('kc_house_data.csv')

X = dataset.iloc[:, [3,4,7,8,9,10,11]] # seleceting covariates from dataset
y = dataset.iloc[:, 2] # seleceting response from dataset

from sklearn.linear_model import LinearRegression
regressor = LinearRegression()

#Fitting model with trainig data
regressor.fit(X, y)

# Saving model to disk
pickle.dump(regressor, open('model.pkl','wb'))

# Loading model to compare the results
model = pickle.load(open('model.pkl','rb'))
print(model.predict([[2, 2, 2,1,2,3,6]]))
```

Name	Type	Size
dataset	DataFrame	(21613, 21)
model	linear_model._base.LinearRegression	1
regressor	linear_model._base.LinearRegression	1
X	DataFrame	(21613, 7)
y	Series	(21613,)

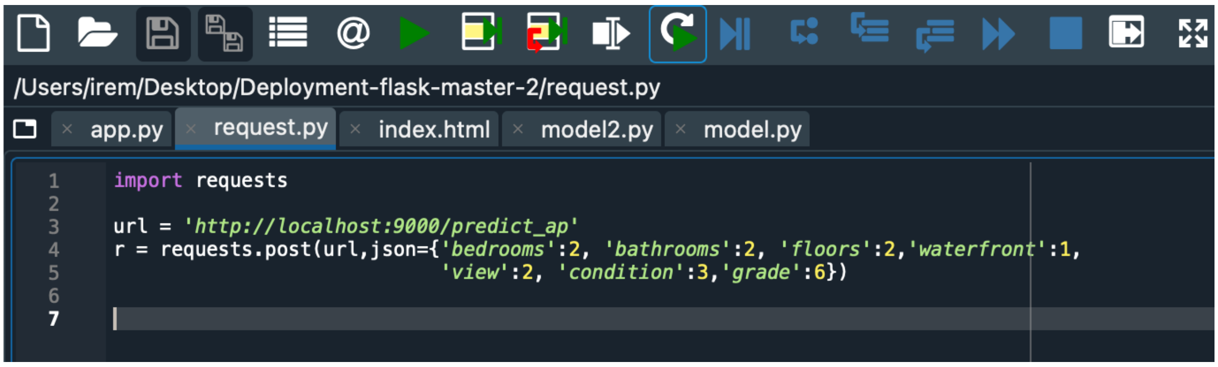
- Linear regression model had conducted to predict house price, taking some characteristics of the houses as predictors.



/Users/irem/Desktop/Deployment-flask-master-2/templates/index.html
app.py
request.py
index.html
model2.py
model.py

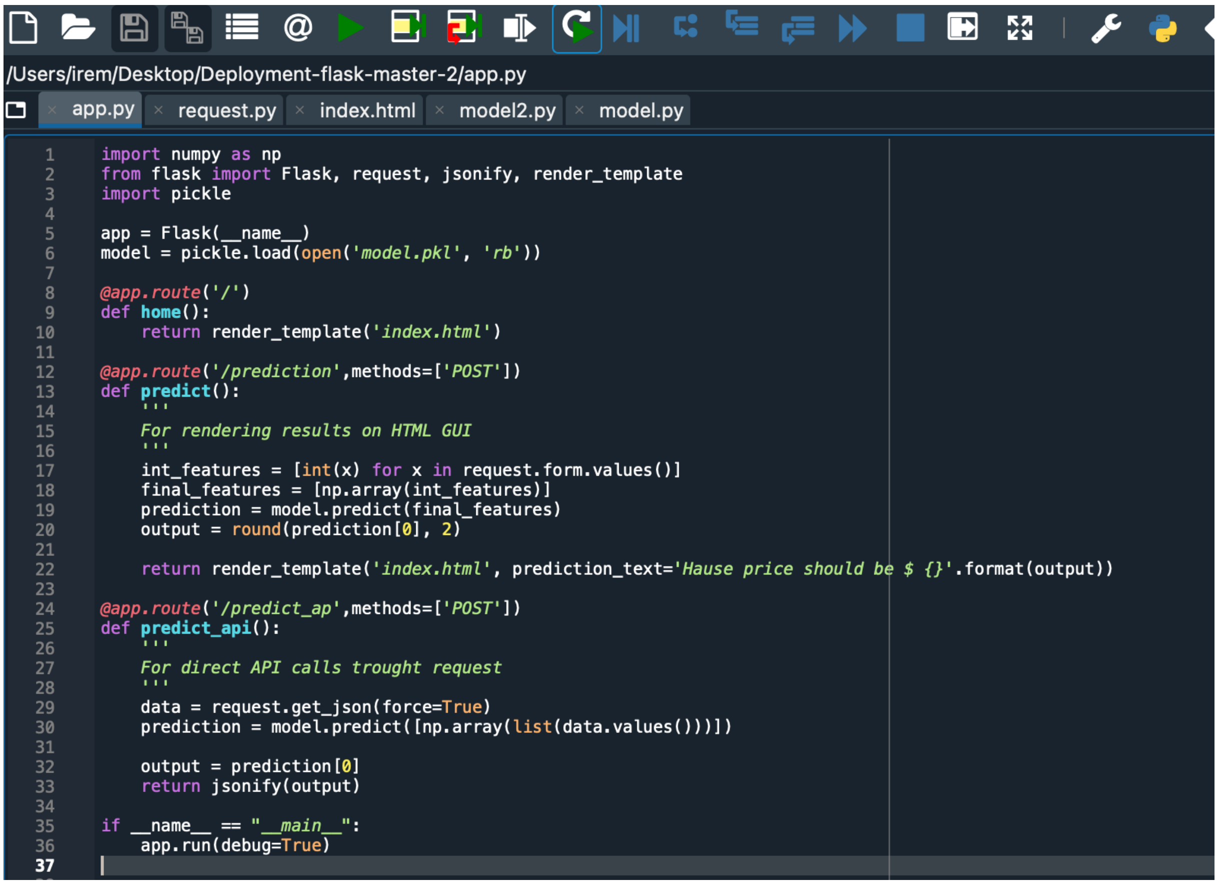
```html
<!DOCTYPE html>
<html >
<!--From https://codepen.io/frytyler/pen/EGdtg-->
<head>
  <meta charset="UTF-8">
  <title>ML API</title>
  <link href='https://fonts.googleapis.com/css?family=Pacifico' rel='stylesheet' type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Arimo' rel='stylesheet' type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Hind:300' rel='stylesheet' type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Open+Sans+Condensed:300' rel='stylesheet' type='text/css'>
<link rel="stylesheet" href="{{ url_for('static', filename='css/style.css') }}">

</head>

<body>
 <div class="login">
    <h1>Prediction of Hause Price</h1>

     <!-- Main Input For Receiving Query to our ML -->
    <form action="{{ url_for('predict')}}"method="post">
        <input type="text" name="bedrooms" placeholder="Bedrooms" required="required" />
        <input type="text" name="bathrooms" placeholder="Bathrooms" required="required" />
        <input type="text" name="floors" placeholder="Floors" required="required" />
        <input type="text" name="waterfront" placeholder="Waterfront" required="required" />
        <input type="text" name="view" placeholder="View" required="required" />
        <input type="text" name="condition" placeholder="Condition" required="required" />
        <input type="text" name="grade" placeholder="Grade" required="required" />

        <button type="submit" class="btn btn-primary btn-block btn-large">Predict</button>
    </form>

   <br>
   <br>
   {{ prediction_text }}

 </div>


</body>
</html>
```

- This is the root node where API URL should go.
- index.html file is like our home page.



```python
import requests

url = 'http://localhost:9000/predict_ap'
r = requests.post(url,json={'bedrooms':2, 'bathrooms':2, 'floors':2,'waterfront':1,
                            'view':2, 'condition':3,'grade':6})

```

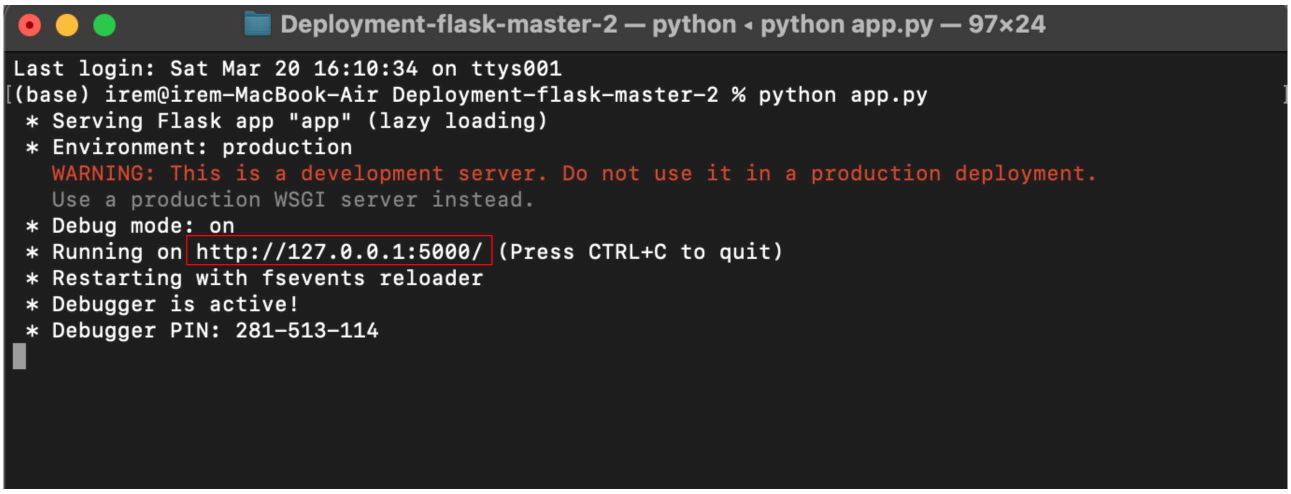
- We create request.py file to give URL. We are just saying that request or post URL and we give json values.


/Users/irem/Desktop/Deployment-flask-master-2/app.py

```python
import numpy as np
from flask import Flask, request, jsonify, render_template
import pickle

app = Flask(__name__)
model = pickle.load(open('model.pkl', 'rb'))

@app.route('/')
def home():
    return render_template('index.html')

@app.route('/prediction',methods=['POST'])
def predict():
    '''
    For rendering results on HTML GUI
    '''
    int_features = [int(x) for x in request.form.values()]
    final_features = [np.array(int_features)]
    prediction = model.predict(final_features)
    output = round(prediction[0], 2)

    return render_template('index.html', prediction_text='Hause price should be $ {}'.format(output))

@app.route('/predict_ap',methods=['POST'])
def predict_api():
    '''
    For direct API calls trought request
    '''
    data = request.get_json(force=True)
    prediction = model.predict([np.array(list(data.values()))])

    output = prediction[0]
    return jsonify(output)

if __name__ == "__main__":
    app.run(debug=True)
```

- In this app.py file, we create flask environment where we will be creating our API, and where we will read this file and then we will give the input to the file.

Then we open the terminal, and the read app.py file on the terminal.

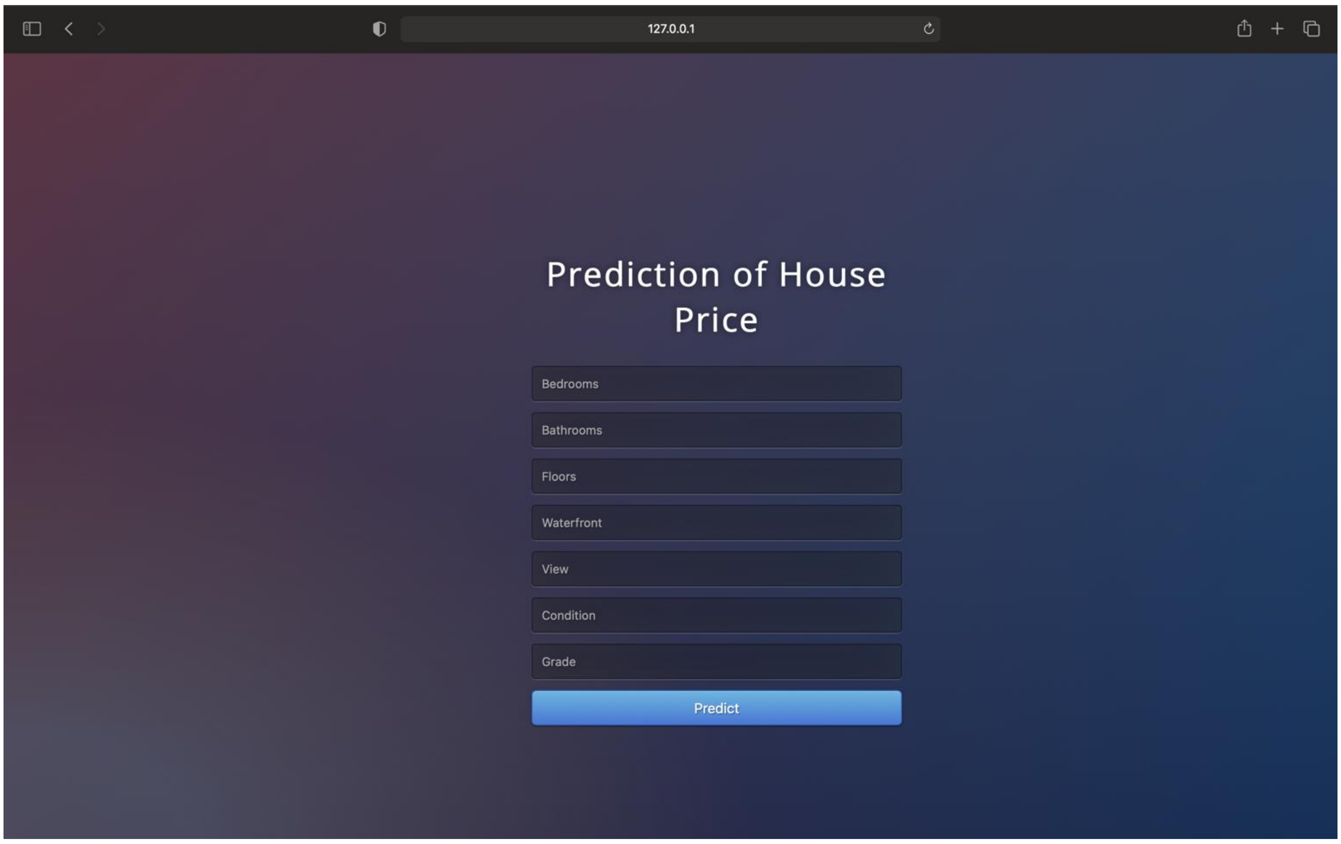

Deployment-flask-master-2 — python ◂ python app.py — 97×24

```
Last login: Sat Mar 20 16:10:34 on ttys001
(base) irem@irem-MacBook-Air Deployment-flask-master-2 % python app.py
 * Serving Flask app "app" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
 * Restarting with fsevents reloader
 * Debugger is active!
 * Debugger PIN: 281-513-114
```

We see that address of the demo is http://127.0.0.1:5000/

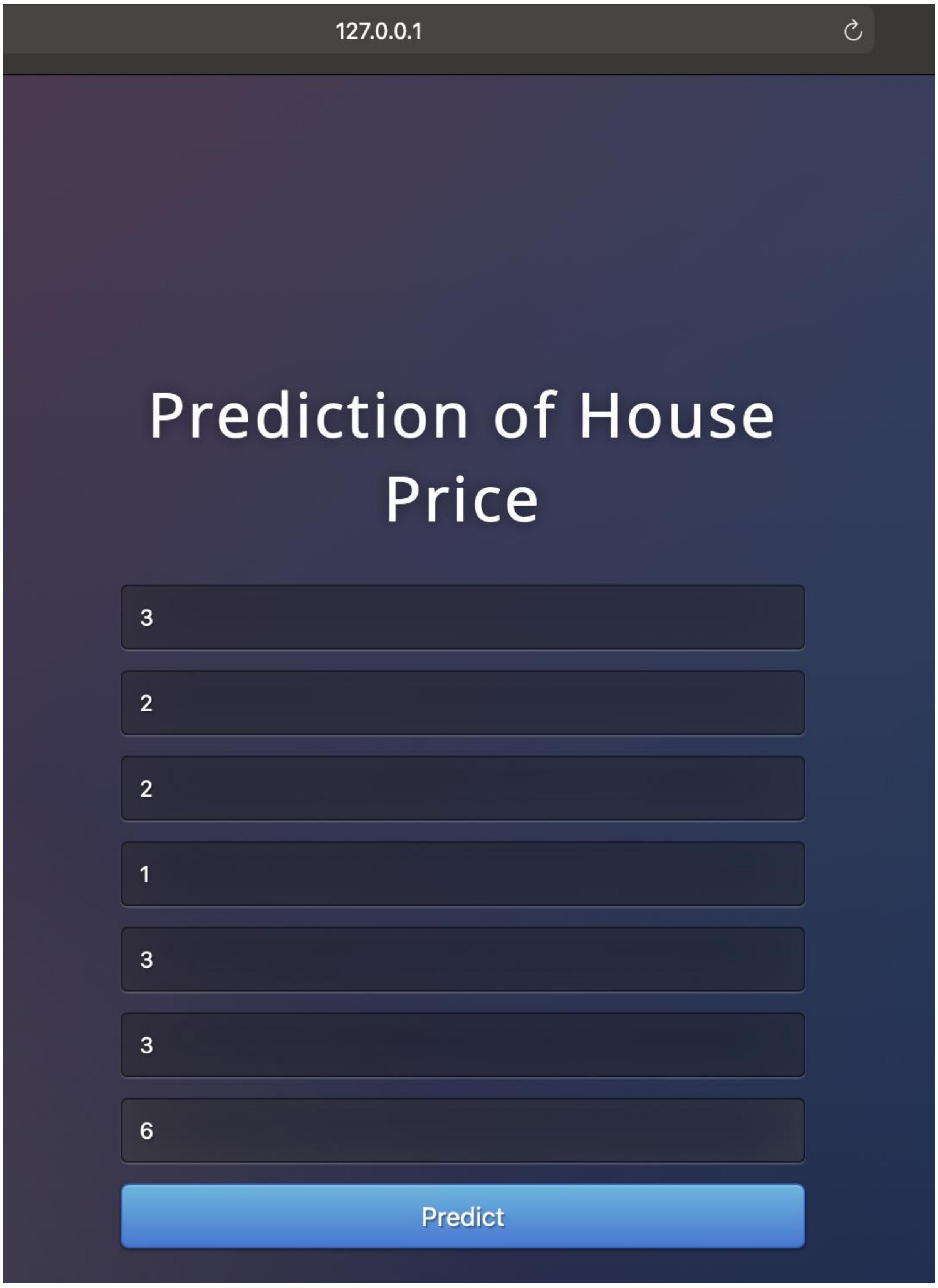
Then we go to this address.

This is our home page.



Let’s try some values to see the prediction. Let’s see what the difference between the price of the houses with waterfront is and without waterfront keeping the other covariates same.

Price of the house with waterfront:



Prediction of House Price

3

2

2

1

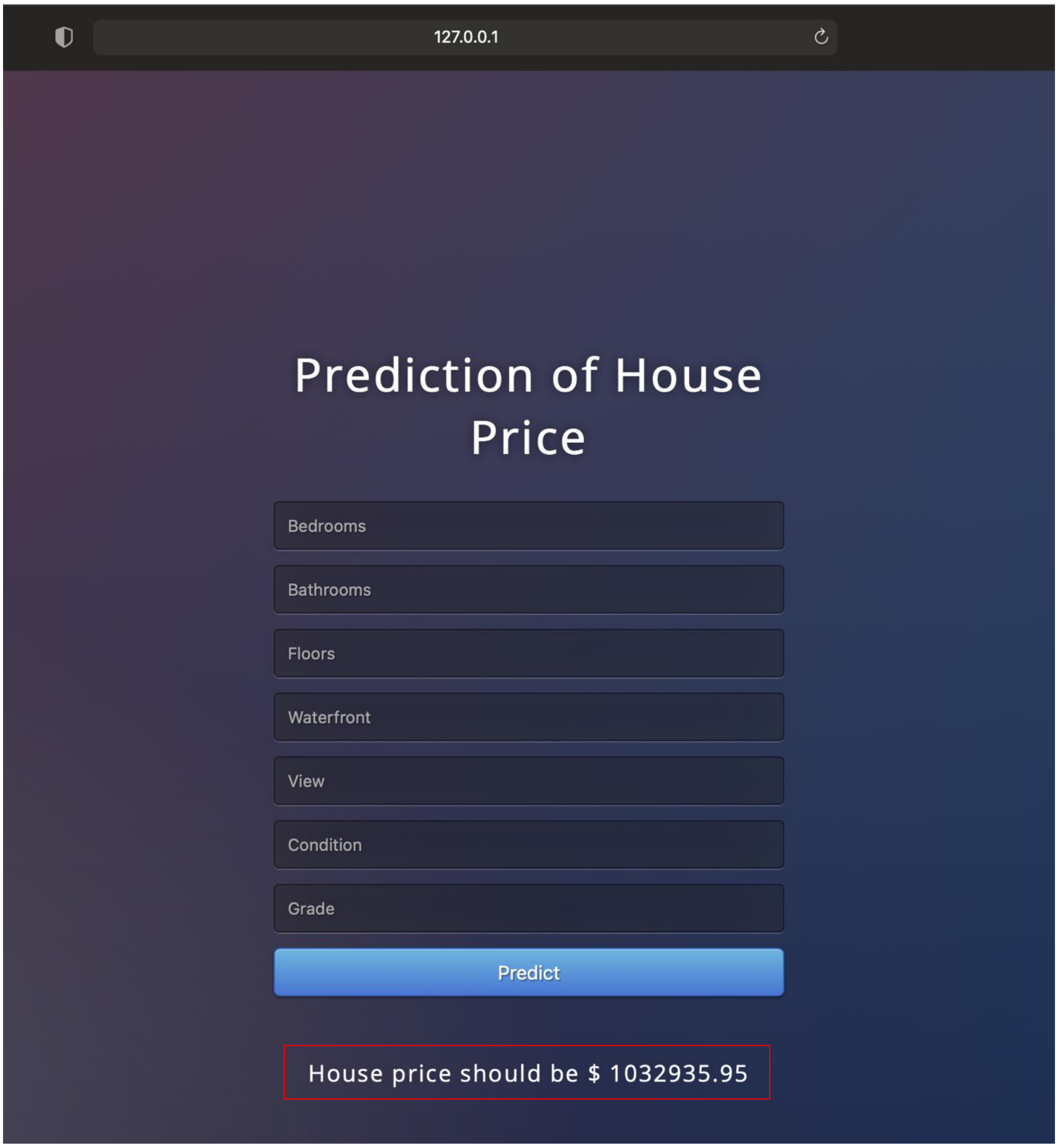
3

3

6

First, we enter the characteristics of the house.

Then we predict:


127.0.0.1

Prediction of House Price

Bedrooms

Bathrooms

Floors

Waterfront

View

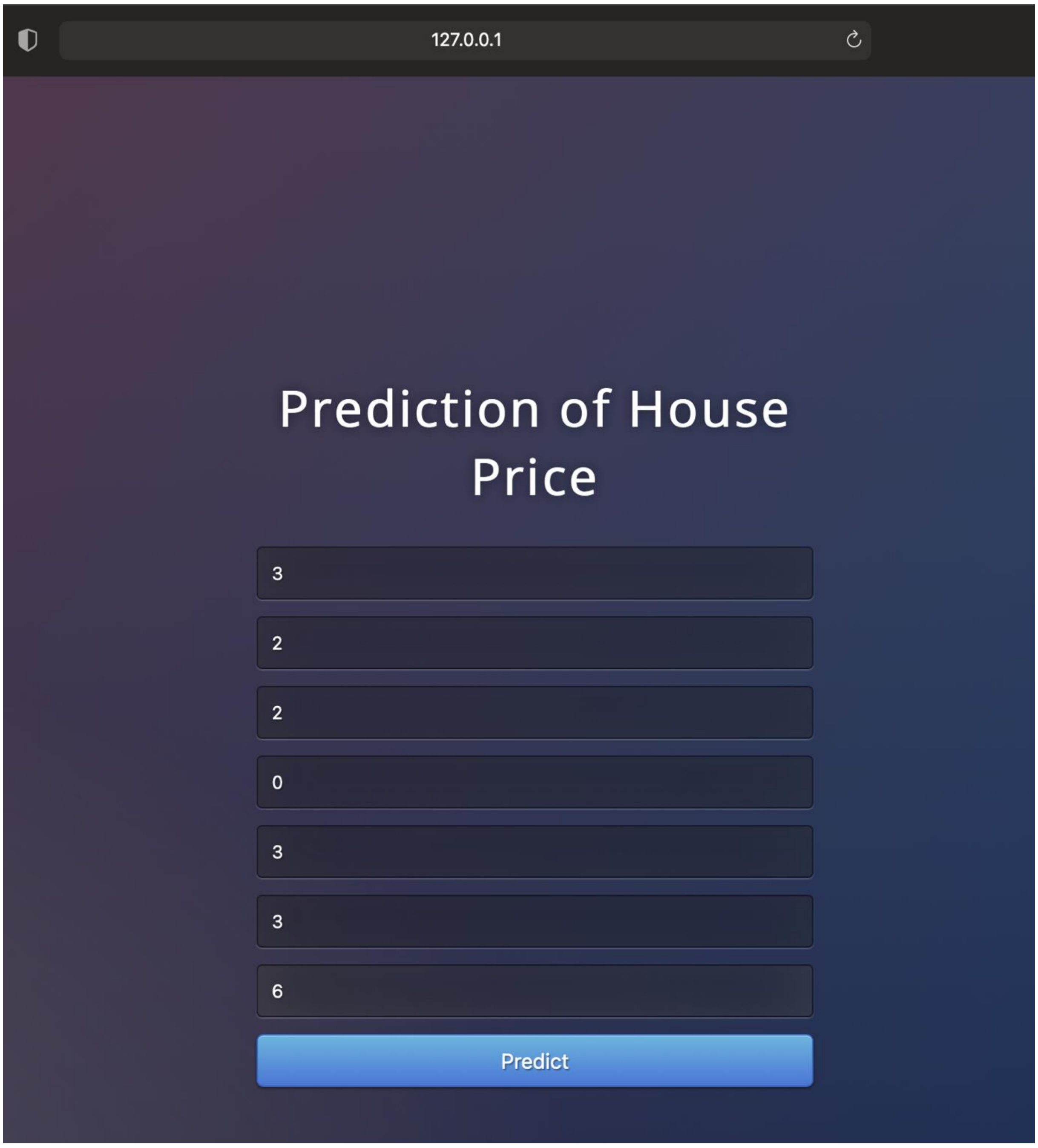
Condition

Grade

Predict

House price should be $ 1032935.95

Price of the house without waterfront:


127.0.0.1

Prediction of House Price

3

2

2

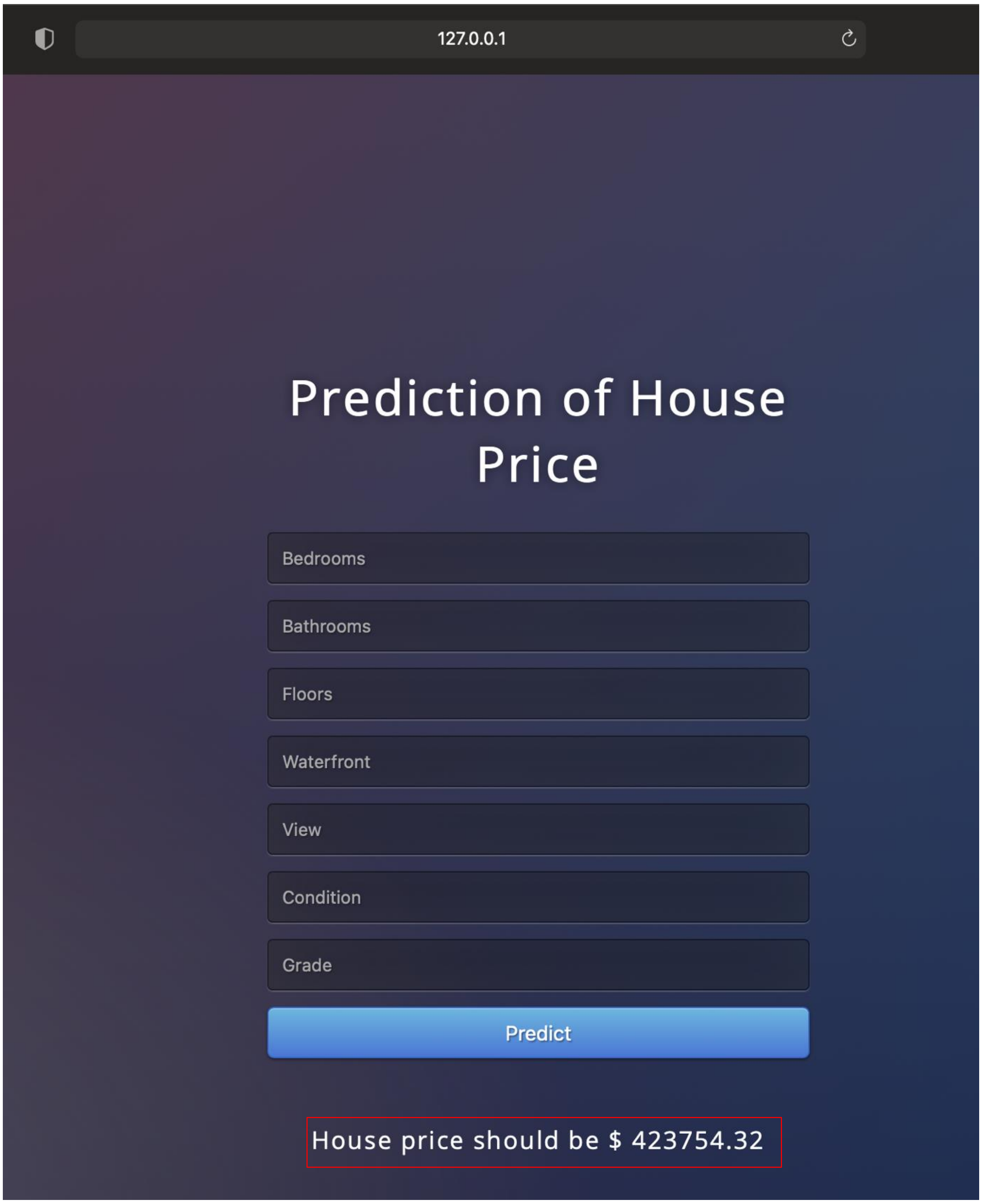
0

3

3

6

Then we predict:


127.0.0.1

Prediction of House Price

House price should be $ 423754.32

Then we see that while the house price with waterfront is $1.032.935, the house price without waterfront is $423.754.