In [1]:

```
#Importing all Libraries

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
import pickle
```

In [2]:

```
# Load the dataset
dataset = pd.read_csv(r"C:\Users\Jan Saida\OneDrive\Documents\Salary_Data.csv")
```

In [3]:

```
dataset
```

Out[3]:

	YearsExperience	Salary
0	1.1	39343
1	1.3	46205
2	1.5	37731
3	2.0	43525
4	2.2	39891
5	2.9	56642
6	3.0	60150
7	3.2	54445
8	3.2	64445
9	3.7	57189
10	3.9	63218
11	4.0	55794
12	4.0	56957
13	4.1	57081
14	4.5	61111
15	4.9	67938
16	5.1	66029
17	5.3	83088
18	5.9	81363
19	6.0	93940
20	6.8	91738
21	7.1	98273
22	7.9	101302
23	8.2	113812
24	8.7	109431
25	9.0	105582
26	9.5	116969
27	9.6	112635
28	10.3	122391
29	10.5	121872

In [4]:

```
# Split the data into independent and dependent variables
X = dataset.iloc[:, :-1].values
y = dataset.iloc[:, 1].values
```

In [5]:

```
X
```

Out[5]:

```
array([[ 1.1],
       [ 1.3],
       [ 1.5],
       [ 2. ],
       [ 2.2],
       [ 2.9],
       [ 3. ],
       [ 3.2],
       [ 3.2],
       [ 3.7],
       [ 3.9],
       [ 4. ],
       [ 4. ],
       [ 4.1],
       [ 4.5],
       [ 4.9],
       [ 5.1],
       [ 5.3],
       [ 5.9],
       [ 6. ],
       [ 6.8],
       [ 7.1],
       [ 7.9],
       [ 8.2],
       [ 8.7],
       [ 9. ],
       [ 9.5],
       [ 9.6],
       [10.3],
       [10.5]])
```

In [6]:

```
y
```

Out[6]:

```
array([ 39343,  46205,  37731,  43525,  39891,  56642,  60150,  54445,
        64445,  57189,  63218,  55794,  56957,  57081,  61111,  67938,
        66029,  83088,  81363,  93940,  91738,  98273, 101302, 113812,
       109431, 105582, 116969, 112635, 122391, 121872], dtype=int64)
```

In [7]:

```
# Split the dataset into training and testing sets (80-20%)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20, random_state=0)
```

In [8]:

```
X_train
```

Out[8]:

```
array([[ 9.6],
       [ 4. ],
       [ 5.3],
       [ 7.9],
       [ 2.9],
       [ 5.1],
       [ 3.2],
       [ 4.5],
       [ 8.2],
       [ 6.8],
       [ 1.3],
       [10.5],
       [ 3. ],
       [ 2.2],
       [ 5.9],
       [ 6. ],
       [ 3.7],
       [ 3.2],
       [ 9. ],
       [ 2. ],
       [ 1.1],
       [ 7.1],
       [ 4.9],
       [ 4. ]])
```

In [9]:

```
X_test
```

Out[9]:

```
array([[ 1.5],
       [10.3],
       [ 4.1],
       [ 3.9],
       [ 9.5],
       [ 8.7]])
```

In [10]:

```
y_train
```

Out[10]:

```
array([112635,  55794,  83088, 101302,  56642,  66029,  64445,  61111,
       113812,  91738,  46205, 121872,  60150,  39891,  81363,  93940,
        57189,  54445, 105582,  43525,  39343,  98273,  67938,  56957],
      dtype=int64)
```

In [11]:
```
y_test
```

Out[11]:
```
array([ 37731, 122391,  57081,  63218, 116969, 109431], dtype=int64)
```

In [12]:
```
# Train the model
regressor = LinearRegression()
regressor.fit(X_train, y_train)
```

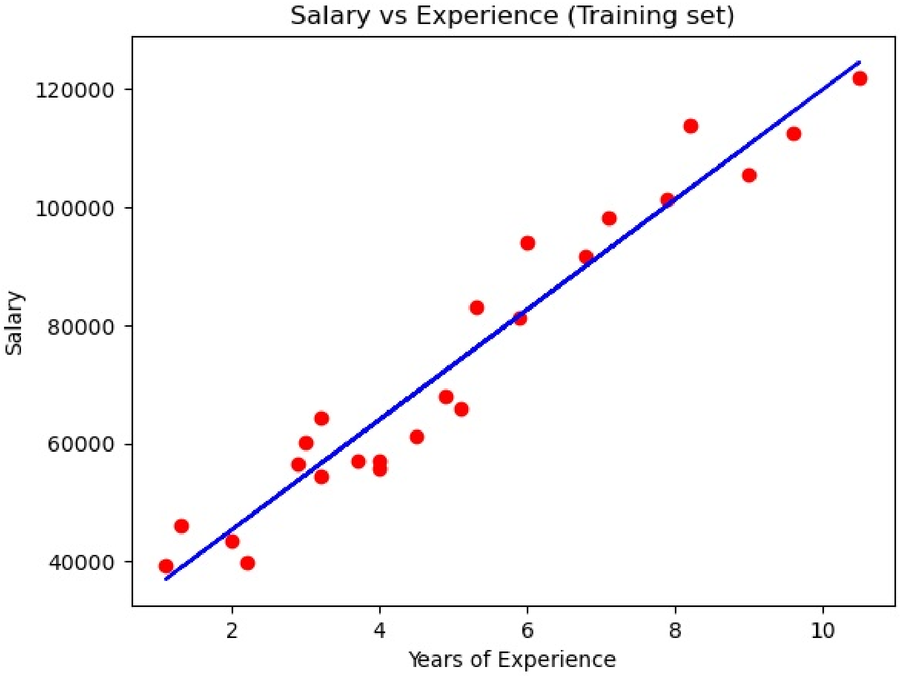
Out[12]:
```
▾ LinearRegression
LinearRegression()
```

In [13]:
```
# Predict the test set
y_pred = regressor.predict(X_test)
```

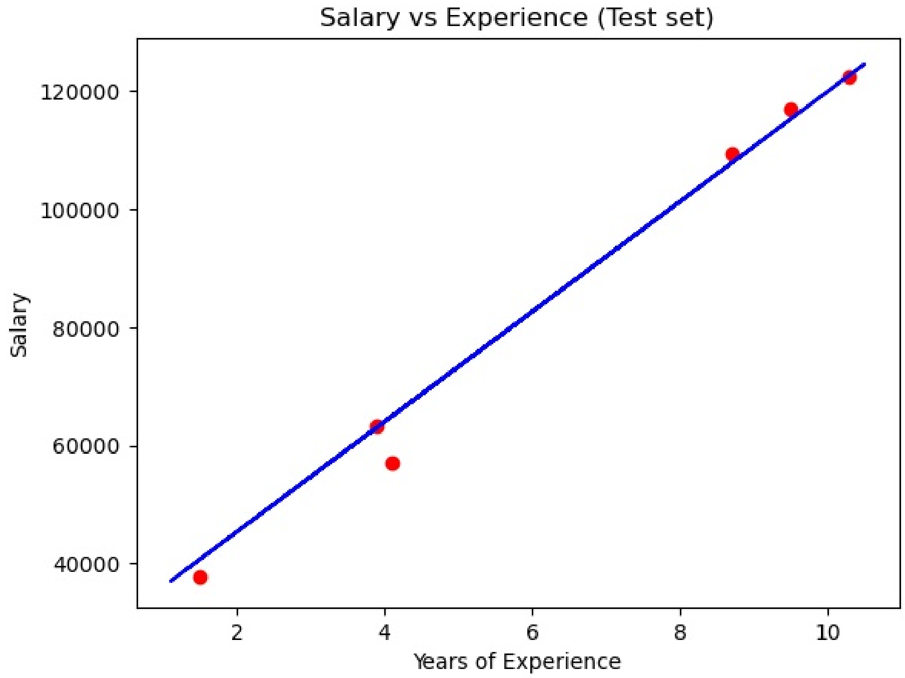
In [14]:
```
y_pred
```

Out[14]:
```
array([ 40748.96184072, 122699.62295594,  64961.65717022,  63099.14214487,
       115249.56285456, 107799.50275317])
```

In [15]:
```
# Visualize the training set
plt.scatter(X_train, y_train, color='red')
plt.plot(X_train, regressor.predict(X_train), color='blue')
plt.title('Salary vs Experience (Training set)')
plt.xlabel('Years of Experience')
plt.ylabel('Salary')
plt.show()
```



In [16]:
```
# Visualize the test set
plt.scatter(X_test, y_test, color='red')
plt.plot(X_train, regressor.predict(X_train), color='blue')
plt.title('Salary vs Experience (Test set)')
plt.xlabel('Years of Experience')
plt.ylabel('Salary')
plt.show()
```



```
In [17]: # Predict salary for 12 and 20 years of experience using the trained model
         y_12 = regressor.predict([[12]])
         y_20 = regressor.predict([[20]])
         print(f"Predicted salary for 12 years of experience: ${y_12[0]:,.2f}")
         print(f"Predicted salary for 20 years of experience: ${y_20[0]:,.2f}")
```

```
Predicted salary for 12 years of experience: $138,531.00
Predicted salary for 20 years of experience: $213,031.60
```

```
In [18]: # Check model performance
         bias = regressor.score(X_train, y_train)
         variance = regressor.score(X_test, y_test)
         train_mse = mean_squared_error(y_train, regressor.predict(X_train))
         test_mse = mean_squared_error(y_test, y_pred)
```

```
In [19]: bias
```

```
Out[19]:  0.9411949620562126
```

```
In [20]: variance
```

```
Out[20]:  0.988169515729126
```

```
In [21]: train_mse
```

```
Out[21]:  36149670.11816131
```

```
In [22]: test_mse
```

```
Out[22]:  12823412.298126549
```

```
In [23]: print(f"Training Score (R^2): {bias:.2f}")
         print(f"Testing Score (R^2): {variance:.2f}")
         print(f"Training MSE: {train_mse:.2f}")
         print(f"Test MSE: {test_mse:.2f}")
```

```
Training Score (R^2): 0.94
Testing Score (R^2): 0.99
Training MSE: 36149670.12
Test MSE: 12823412.30
```

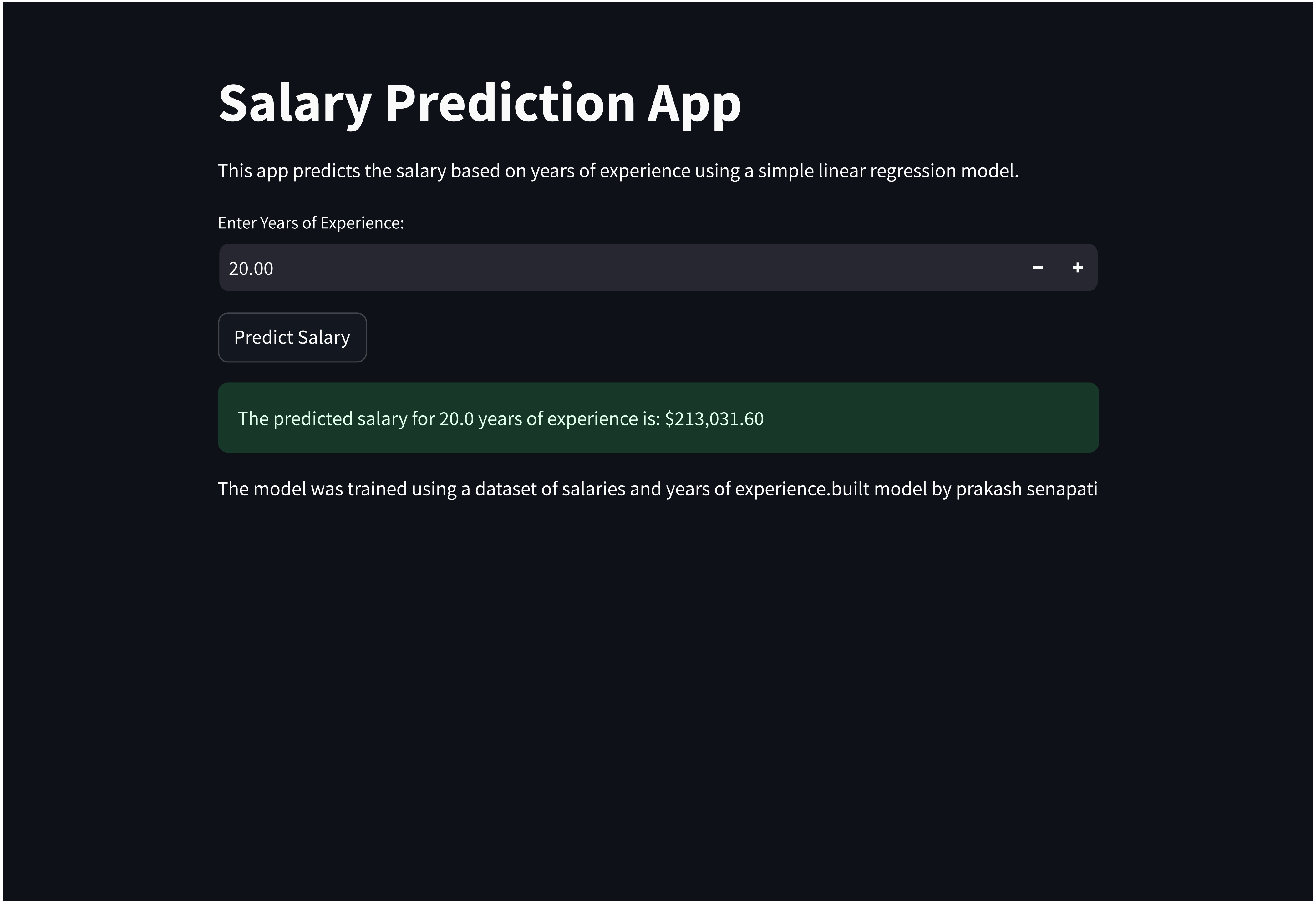
```
In [24]: # Save the trained model to disk
         filename = 'linear_regression_model.pkl'
         with open(filename, 'wb') as file:
             pickle.dump(regressor, file)
         print("Model has been pickled and saved as linear_regression_model.pkl")
```

```
Model has been pickled and saved as linear_regression_model.pkl
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js

Streamlit\Salary_Prediction_slr.py

```python
import streamlit as st
import pickle
import numpy as np

# Load the saved model

with open(r"C:\Users\Jan Saida\linear_regression_model.pkl", "rb") as file:
    model = pickle.load(file)


# Set the title of the Streamlit app
st.title("Salary Prediction App")

# Add a brief description
st.write("This app predicts the salary based on years of experience using a simple linear regression model.")

# Add input widget for user to enter years of experience
years_experience = st.number_input("Enter Years of Experience:", min_value=0.0, max_value=50.0, value=1.0, step=0.5)

# When the button is clicked, make predictions
if st.button("Predict Salary"):
    # Make a prediction using the trained model
    experience_input = np.array([[years_experience]])  # Convert the input to a 2D array for prediction
    prediction = model.predict(experience_input)

    # Display the result
    st.success(f"The predicted salary for {years_experience} years of experience is: ${prediction[0]:,.2f}")

# Display information about the model
st.write("The model was trained using a dataset of salaries and years of experience.built model by prakash senapati")
```



Salary Prediction App

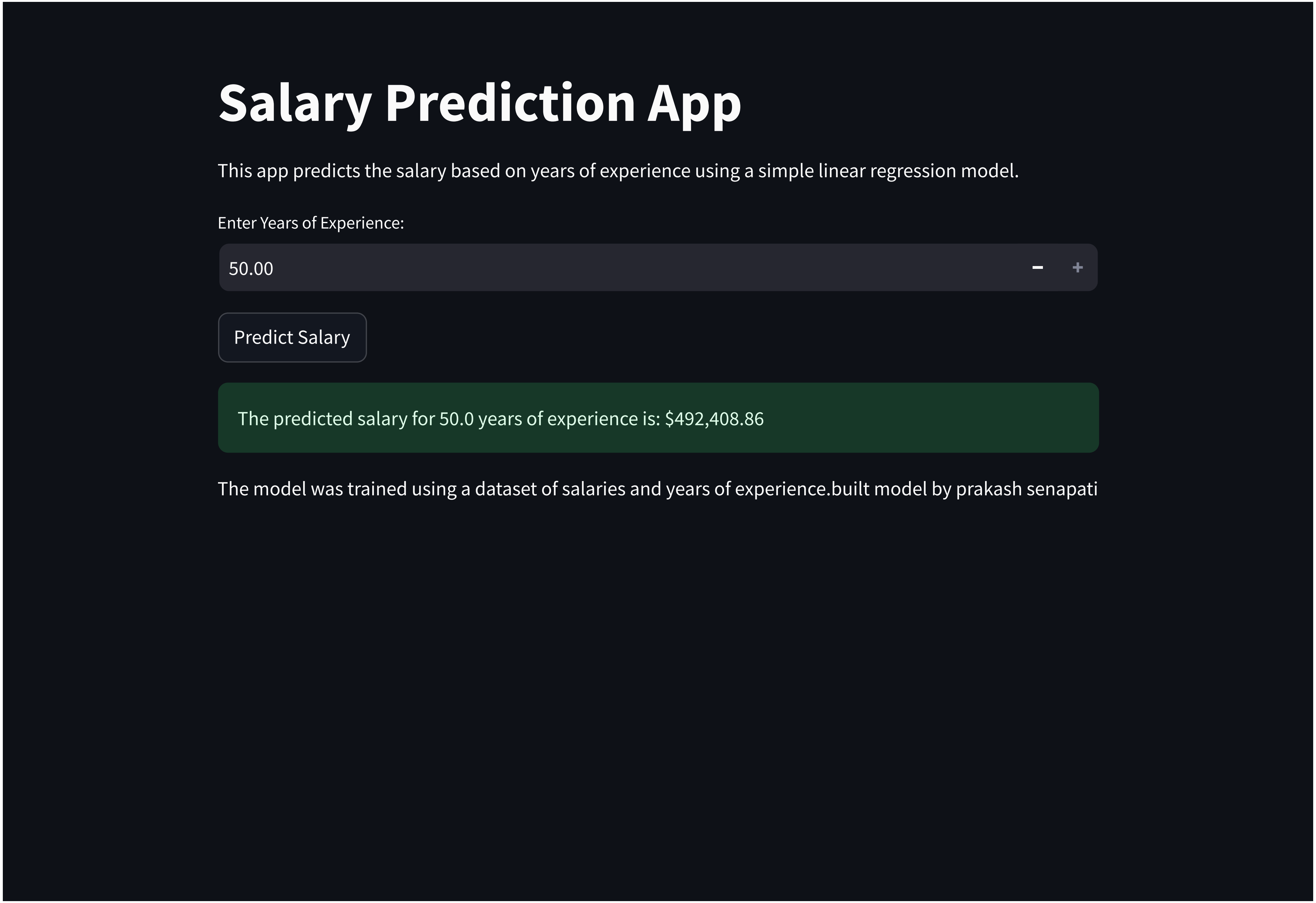
This app predicts the salary based on years of experience using a simple linear regression model.

Enter Years of Experience:

20.00

The predicted salary for 20.0 years of experience is: $213,031.60

The model was trained using a dataset of salaries and years of experience.built model by prakash senapati



Salary Prediction App

This app predicts the salary based on years of experience using a simple linear regression model.

Enter Years of Experience:

50.00

The predicted salary for 50.0 years of experience is: $492,408.86

The model was trained using a dataset of salaries and years of experience.built model by prakash senapati