```
#Name :Mane Shivraj Pandurang

#class: B.E.A.I & D.S.
#Roll No:37
#Subject : Deep Learning (CL-IV)
```

In [35]:
```
## Practical No. 1
#1. Problem Statement Real estate agents want help to predict the house price fo
```

In [4]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
```

In [7]:
```
df = pd.read_csv("/content/USA_Housing.csv")
```

In [17]:
```
df.head()
```

Out[17]:

	Avg. Area Income	Avg. Area House Age	Avg. Area Number of Rooms	Avg. Area Number of Bedrooms	Area Population	Price
0	79545.458574	5.682861	7.009188	4.09	23086.800503	1.059034e+06
1	79248.642455	6.002900	6.730821	3.09	40173.072174	1.505891e+06
2	61287.067179	5.865890	8.512727	5.13	36882.159400	1.058988e+06
3	63345.240046	7.188236	5.586729	3.26	34310.242831	1.260617e+06
4	59982.197226	5.040555	7.839388	4.23	26354.109472	6.309435e+05

In [18]:
```
# Checking for missing values
print(df.isnull().sum())
```

```
Avg. Area Income                0
Avg. Area House Age             0
Avg. Area Number of Rooms       0
Avg. Area Number of Bedrooms    0
Area Population                 0
Price                           0
dtype: int64
```

In [19]:
```
print (df.head())
```

```
   Avg. Area Income  Avg. Area House Age  Avg. Area Number of Rooms  \
0      79545.458574             5.682861                   7.009188   
1      79248.642455             6.002900                   6.730821   
2      61287.067179             5.865890                   8.512727   
3      63345.240046             7.188236                   5.586729   
4      59982.197226             5.040555                   7.839388   

   Avg. Area Number of Bedrooms  Area Population         Price  
0                          4.09     23086.800503  1.059034e+06  
1                          3.09     40173.072174  1.505891e+06  
2                          5.13     36882.159400  1.058988e+06  
3                          3.26     34310.242831  1.260617e+06  
4                          4.23     26354.109472  6.309435e+05  
```

In [22]:
```
#Step 4: Define Features and Target Variable
X = df.drop(columns=['Price'])  # Features
y = df['Price']  # Target variable
```

In [23]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_
```

In [24]:
```
model = LinearRegression()
model.fit(X_train, y_train)
```

Out[24]:
```
▾ LinearRegr ssion
LinearRegression()
```

In [28]:
```
#Step 7: Make Predictions
y_pred = model.predict(X_test)
```

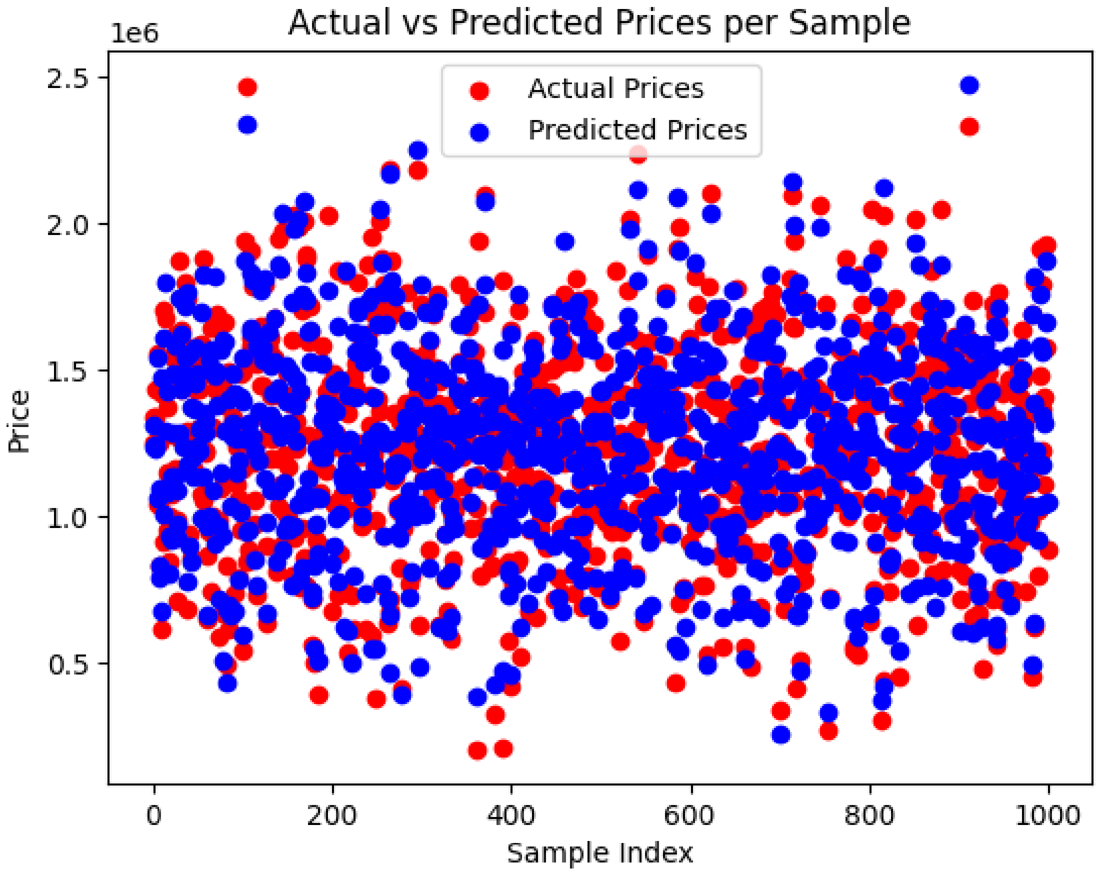
In [29]:
```
#Step 8: Evaluate the Model
print("Mean Absolute Error (MAE):", mean_absolute_error(y_test, y_pred))
print("Mean Squared Error (MSE):", mean_squared_error(y_test, y_pred))
print("Root Mean Squared Error (RMSE):", np.sqrt(mean_squared_error(y_test, y_pr
print("R-squared Score (R2):", r2_score(y_test, y_pred))
```

```
Mean Absolute Error (MAE): 80879.0972348982
Mean Squared Error (MSE): 10089009300.894518
Root Mean Squared Error (RMSE): 100444.06055558745
R-squared Score (R2): 0.9179971706834289
```

In [34]:
```
#Step 9: Visualize the Predictionsplt.figure(figsize=(10, 6))

plt.scatter(range(len(y_test)), y_test, color='red', label='Actual Prices')
plt.scatter(range(len(y_pred)), y_pred, color='blue', label='Predicted Prices')

plt.xlabel("Sample Index")
plt.ylabel("Price")
plt.title("Actual vs Predicted Prices per Sample")
plt.legend()
plt.show()
```



Actual vs Predicted Prices per Sample
1e6
2.5
2.0
1.5
1.0
0.5
Price
0
200
400
600
800
1000
Sample Index
● Actual Prices
● Predicted Prices