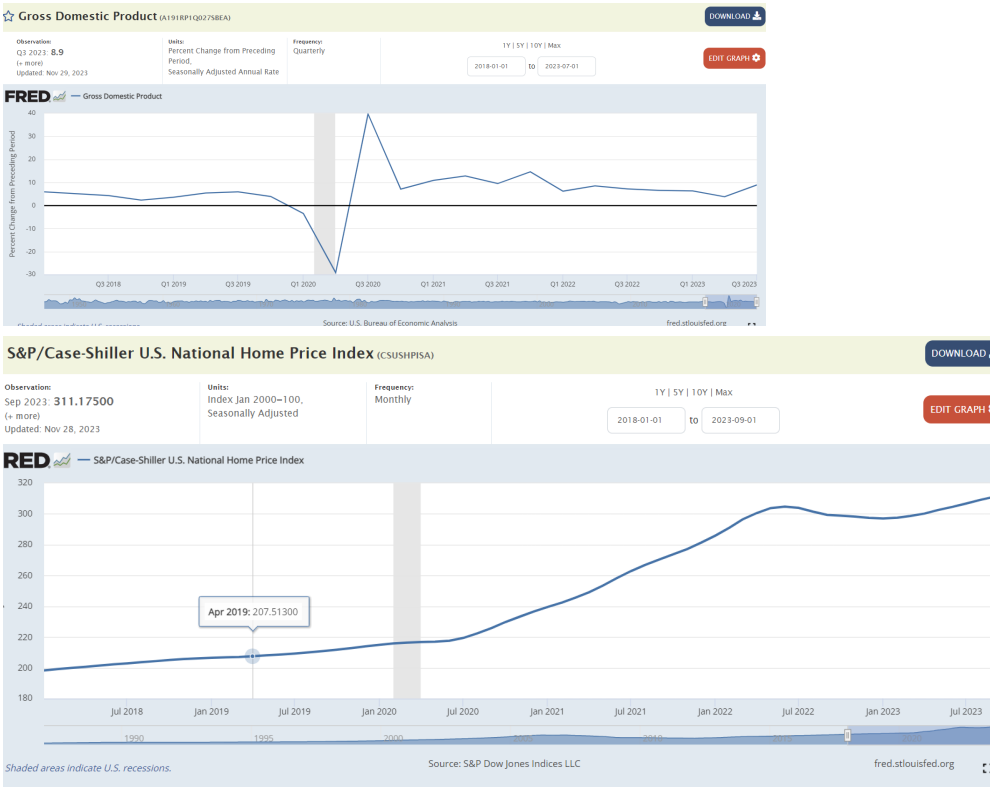
PROBLEM STATEMENT - Find publicly available data for key factors that influence US home prices nationally. Then,build a data science model that explains how these factors impacted home prices over the last 20 years. Use the S&P Case-Schiller Home Price Index as a proxy for home prices

The prices of homes is influenced by a number of demand and supply side factors which are discussed below:

1. **Income Level:** High-income levels correlate with high home prices. Gross Domestic Product (GDP) is taken as an indicator of the overall income level of the economy. Real Median Household income **is** taken as an indicator of average household income.
2. **Interest Rates:** The interest rates, particularly mortgage rates significantly influence home affordability and, consequently, demand and prices. 30-Year Mortgage Rate has been chosen as the relevant predictor.
3. **Housing Supply:** National trends in housing supply can affect overall home prices. A shortage of housing relative to demand tends to drive prices up, and vice versa. Housing inventory has been used as a relevant independent variable.
4. **Inflation:** Inflationary pressures can contribute to rising home prices as the cost of materials and labor increases, leading to higher construction costs. For incorporating inflation pressures in our model, we use Producer Price Index:Construction Materials.
5. **Consumer Confidence:** The overall confidence of consumers in the national economy and income levels can affect their willingness to make significant investments like purchasing a home. GDP Growth rate has been taken as a proxy for consumer confidence.. [A noticeable expectation in this regard can be noted in 2020, when because of the COVID19 pandemics, growth rates dipped sharply, however, the prices of homes continued to rise due to supply side constraints and increased demand owing to increased at-home work options.]



Variables/Datasets that were considered but not included in the model:

1. "University of Michigan: Consumer sentiment" Index was considered as an indicator of consumer confidence, but since we already have accounted for the absolute income levels, interest rates and inflation in our model, therefore the remaining major determinant of consumer confidence, i.e. growth in income has been represented through the GDP Growth Rate.
2. 'Unemployment' was considered as a predictor of demand, but since unemployment has a negative correlation with inflation and Gross Domestic product Growth, which have already been accounted for in our model, therefore no new independent variable was introduced.
3. Federal Funds Effective Rate (FEDFUNDS) was considered as a measure of interest rate, however, as discussed above, 30 Year Mortage Rate is a more relevant factor for affecting consumer demand.

The steps undertaken in this projects are discussed as below:

1. **Data Collection** - The requisite datasets were downloaded from https://fred.stlouisfed.org/ .

   Real Median Household Income in the United States (MEHOINUSA672N)-[Yearly] - Median national income of US over span of 20 years.

   Gross Domestic Product (GDP) [Quarterly]- Gross domestic product (GDP), the featured measure of U.S. output, is the market value of the goods and services produced by labor and property located in the United States.

   Year-on-year Growth Rate of Gross Domestic Product (A191RP1Q027SBEA) [Quarterly] - Percent Change from Preceding Period, Seasonally Adjusted Annual Rate

   Producer Price Index by Commodity: Special Indexes: Construction Materials (WPUSI012011) - [Monthly]

   30-Year Fixed Rate Mortgage Average in the United States (MORTGAGE30US) - [Weekly]

   Housing Inventory Estimate: Vacant Housing Units in the United States (EVACANTUSQ176N) - [Quarterly]

2. **Data Analysis** - After collecting and merging data, Correlation between factors and correlation between HomePrice index and Factors was evaluated

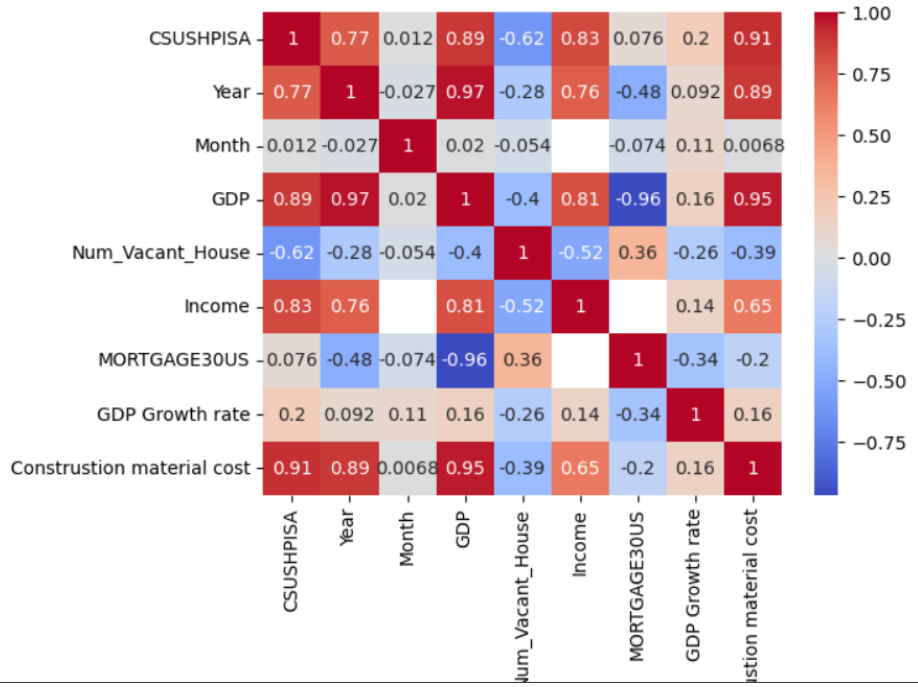
Checking for Null Values

```
In [175]: 1 data_final.isna().sum()
```

```
Out[175]: CSUSHPISA                  0
          Year                       0
          Month                      0
          GDP                      166
          Num_Vacant_House         166
          Income                   229
          MORTGAGE30US             216
          GDP Growth rate          167
          Construstion material cost   0
          dtype: int64
```

Checking for Correlation betweeen factors before pre processing

```
[176]: 1 corr = data_final.corr()
       2 sns.heatmap(corr,annot=True,cmap='coolwarm')
       3 plt.show()
```



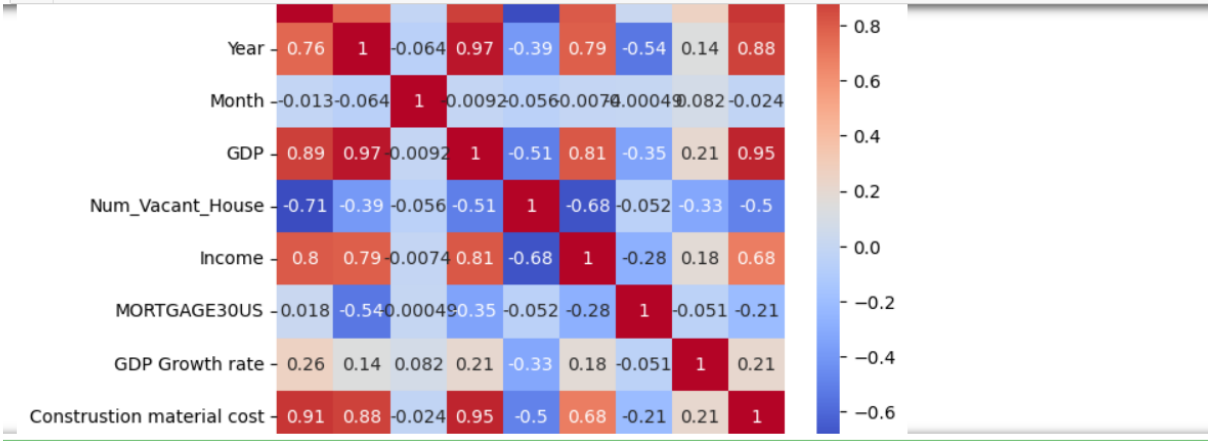
Following correlation values are notable: GDP-CSUSHPISA-> 0.89 , Construction material cost -CSUSHPISA-> 0.91 , Income-CSUSHPISA-> 0.83, Mortage30US-GDP-> -0.96, Income - GDP = 0.81

3. Handling Null values - using interpolation method- Null values were found in the Dataset due to different frequencies of reporting . So before proceeding further, the missing data was interpolated using the linear Interpolation method .

```
In [186]: data_final.isna().sum()

Out[186]: CSUSHPISA                    0
          Year                         0
          Month                        0
          GDP                          0
          Num_Vacant_House             0
          Income                       0
          MORTGAGE30US                 7
          GDP Growth rate              3
          Construstion material cost   0
          dtype: int64

In [187]: data_final.dropna(inplace = True)

In [188]: data_final.isna().sum()

Out[188]: CSUSHPISA                    0
          Year                         0
          Month                        0
          GDP                          0
          Num_Vacant_House             0
          Income                       0
          MORTGAGE30US                 0
          GDP Growth rate              0
          Construstion material cost   0
          dtype: int64

In [189]: data_final.shape

Out[189]: (242, 9)
```

After handling null values, correlation was evaluated again.

```
sns.heatmap(corr,annot=True,cmap='coolwarm')
plt.show()
```



4. Exploratory Data Analysis(EDA) on the dataset.

5. Basic Linear regression model- model-fitting was done in simple basic mode.

In [69]:
```
lr1 = LinearRegression()
lr1.fit(x_train,y_train)
```

Out[69]:
```
LinearRegression
LinearRegression()
```

In [70]:
```
mscore(lr1)
```

```
Train Score 0.9753405778100711
Test Score 0.9782742485186939
```

In [71]:
```
#print regression coefficients
print(lr1.intercept_)
```

```
-140.6806991544864
```

In [72]:
```
print(lr1.coef_)
```

```
[-5.86304327e-04 -4.84011100e-03  3.16580178e-03  1.02956329e+01
  2.32354112e-01  6.86388916e-01]
```

In [73]:
```
ypred_lr1 =  lr1.predict(x_test)
eval_model(y_test,ypred_lr1)
```

```
MAE 5.493365059444946
MSE 50.52884772634824
RMSE 7.108364630936446
R2 Score 0.9782742485186939
```

Resultant R2 Score - 0.9782

train score - 0.9753

test score - 0.9782

6. Optimizing model by scaling data and applying lasso

TUNNING BASIC MODEL

In [218]:
```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
x_train_sc = scaler.fit_transform(x_train)
x_test_sc = scaler.transform(x_test)
```

In [219]:
```
lasso_cv_model = LassoCV(alphas=np.logspace(-4, 4, 100),cv=5)
lasso_cv_model.fit(x_train_sc, y_train)
lasso_cv_predictions = lasso_cv_model.predict(x_test_sc)
```

In [220]:
```
optimal_alpha = lasso_cv_model.alpha_

mse = mean_squared_error(y_test, lasso_cv_predictions)

rmse = np.sqrt(mse)

r2 = r2_score(y_test, lasso_cv_predictions)

print(f"Optimal Alpha: {optimal_alpha:.4f}")
print(f"Mean Squared Error (MSE): {mse:.2f}")
print(f"RMSE: {rmse:2f}")
print(f"R-squared (R2): {r2:.2f}")
```

```
Optimal Alpha: 0.2477
Mean Squared Error (MSE): 49.47
RMSE: 7.033229
R-squared (R2): 0.98
```

7. Checking robustness of model by CV

Used to check robustness of the model

In [221]:
```
from sklearn.model_selection import cross_val_score
```

In [222]:
```
model_lr  = LinearRegression()
cv_res = cross_val_score(model_lr,x,y,scoring='r2',cv=5)  # cv = number of splits
print(cv_res)  # we will get 5 different R2 scores for 5 differnet splits
print(cv_res.mean())
```

```
[ 0.41090803  0.77275415  0.71639188 -1.58183654 -0.70036164]
-0.07642882237431592
```

In [223]:
```
cross_val_scores = cross_val_score(lasso_cv_model, x_train_sc, y_train, cv=5, scoring='r2')
print("Cross-Validation R-squared Scores:")
print(cross_val_scores)
```

```
Cross-Validation R-squared Scores:
[0.98175733 0.96764296 0.95411014 0.9786769  0.97684151]
```

In [224]:
```

mean_r2 = cross_val_scores.mean()
std_r2 = cross_val_scores.std()

print(f"Mean R-squared: {mean_r2:.4f}")
print(f"Standard Deviation of R-squared: {std_r2:.4f}")

```

```
Mean R-squared: 0.9718
Standard Deviation of R-squared: 0.0100
```

In [225]:
```
print(0.1 * rmse)
print(0.1 * df_final['CSUSHPISA'].mean()) # rmse should be less than the 10% of mean of targ
```

```
0.7033229030569772
18.664628512396696
```

7a. Saving the Predicted values and actual test values and plot of excel graph for basic linear regression model.

01-07-2016	180	176.4688
01-11-2012	144	145.693
01-08-2021	266	260.4644
01-09-2005	176	163.9637
01-08-2008	161	170.5303
01-02-2005	163	166.6175
01-02-2014	162	157.7412
01-08-2011	139	143.9831
01-11-2019	212	209.1059
01-01-2015	168	171.5386
01-07-2023	306	307.4194
01-11-2009	148	144.9897
01-05-2013	152	155.0453
01-12-2012	145	147.3809
01-12-2004	159	165.4669
01-10-2019	211	211.1508
01-03-2009	148	152.7162
01-07-2017	191	186.3578
01-10-2006	183	178.9256
01-08-2023	309	307.0103

8. Attempting other Regression model and saving result of best model

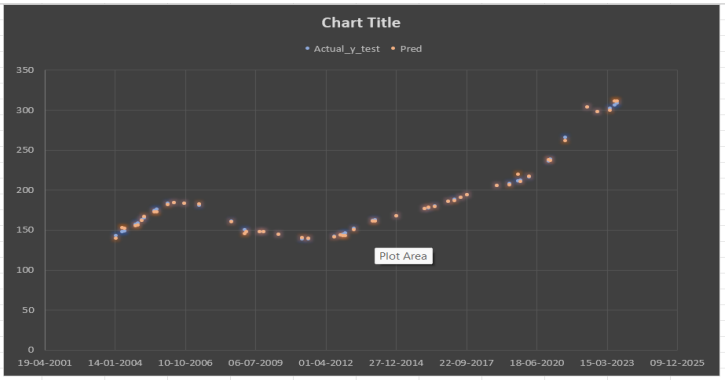
In [119]:
```
all_res = pd.concat([lr2_df,dt1_df,knn1_df,rf1_df,ada1_df,xgb1_df])
all_res
```

Out[119]:

	Train_R2	Test_R2	Test_MSE	Test_RMSE	Test_MAE
LinReg	0.975341	0.978514	49.927752	7.065957	5.488418
DTree_Reg	0.985762	0.984272	36.547834	6.045480	4.558808
KNN_Reg	0.989517	0.987957	27.985347	5.290118	4.123684
RF_Reg	0.997025	0.994501	12.779144	3.574793	2.676995
AdaBoost_Reg	0.992122	0.992029	18.523197	4.303858	3.542984
XGBoost_Reg	1.000000	0.997706	5.330154	2.308713	1.617565

9. Plotly result of XGBoost model on graph

DATE	Actual_y_test	Pred
01-08-2005	174	172.59798
01-02-2004	143	139.66206
01-04-2016	178	178.51808
01-01-2021	239	237.19461
01-05-2023	302	299.6773
01-03-2020	216	217.09409
01-06-2019	208	206.851
01-02-2016	177	177.06676
01-05-2004	148	152.71289
01-01-2013	146	143.05377
01-05-2007	181	182.39417
01-12-2020	236	238.03163
01-06-2022	304	303.77737
01-10-2017	194	194.0496
01-06-2010	145	144.71509
01-01-2017	186	185.96988
01-11-2022	298	297.703
01-11-2004	157	155.5971
01-02-2009	150	145.62022
01-03-2014	163	160.94908
01-05-2006	184	183.86864
01-03-2005	165	167.07684
01-08-2012	142	141.52797
01-02-2006	183	182.0806



10. Since Median Household Income display a high correlation with Gross Domestic Product, and since the frequency of data is yearly (consequently the resultant high degree of interpolations are not amenable to accurate modelling), therefore, the basic linear regression model was re-applied after removing Median Household Income as an independent variable.

Documenting result

```
Out[143]: LinearRegression
          LinearRegression()

In [144]:
1 mscore(lr_ni)

Train Score 0.9606922761970201
Test Score 0.9668038615049208

In [145]:
1 print(lr_ni.intercept_)
2 print(lr_ni.coef_)

110.41415356710635
[ 5.42052560e-03 -8.99155184e-03  1.11472265e+01  7.34354974e-02
  3.63384970e-01]

In [146]:
1 ypred_lr_ni =  lr_ni.predict(x2_test)
2 eval_model(y2_test,ypred_lr_ni)

MAE 7.337732558653294
MSE 77.2061960003508
RMSE 8.786705639791 9
R2 Score 0.9668038615049208

In [147]:
1 res_df2 = pd.DataFrame({'Actual_y_test':y2_test,'Pred':ypred_lr_ni})
2 res_df2.head(5)

Out[147]:
```

DATE	Actual_y_test	Pred
2005-08-01	174.442	164.163372
2004-02-01	143.192	151.332915

It can be noted that there is a decrease in the r2 score from **0.9782** to **0.9668** . Although the difference is moderately significant, the new model still remains robust.

11. Attempting an OLS model and Checking for Multicollinearity

Upon inspecting the OLS model, there wasn't a significant improvement observed. Consequently, the results did not prove to be very fruitful. The VIF (Variance Inflation Factor) values were notably high, indicating the presence of multicollinearity.It is evident that historical GDP data can influence construction material costs, and vice versa, leading to multicollinearity in various factors. Typically, one would address multicollinearity by either removing factors with high VIF or employing dimension reduction techniques such as PCA. However, in this dataset, we opt to retain these factors because they are essential contributors to the model. Removing them could adversely affect the model's

performance.Therefore, despite the high VIF values, we choose not to eliminate these factors, recognizing their importance in influencing the model's performance negatively if removed.