# A real estate agent want help to predict the house price for regions in Usa.he gave us the dataset to work on to use linear Regression model.Create a model that helps him to estimate ¶

## Data Collection

In [ ]:
```
#import libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [ ]:
```
#import the dataset
data=pd.read_csv(r"C:\Users\user\Desktop\Vicky\13_placement.csv")
```

In [ ]:
```
#to display top 10 rows
data.head()
```

In [ ]:
```
#to display null values
data.info()
```

In [ ]:
```
data.shape
```

In [ ]:
```
#to display summary of statistics
data.describe()
```

In [ ]:
```
#to display columns name
data.columns
```

In [ ]:
```
data1=data[['cgpa', 'placement_exam_marks', 'placed']]
```

In [ ]:
```
sns.pairplot(data1)
```

## EDA and Visualization

In [ ]:
```
#sns.distplot(data['Co2-Emissions'])
```

In [ ]:
```
sns.heatmap(data1.corr())
```

## To train the model

we are going to train the linear regression model ;We need to split the two variable x and y where x in independent variable (input) and y is dependent of x(output) so we could ignore address columns as it is not requires for our model

In [ ]:
```
x=data[['cgpa', 'placement_exam_marks']]
y=data1['placed']
```

In [ ]:
```
#To split test and train data
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.1)
```

In [ ]:
```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(x_train,y_train)
```

In [ ]:
```
lr.intercept_
```

In [ ]:
```
coeff = pd.DataFrame(lr.coef_,x.columns,columns=["Co-efficient"])
coeff
```

In [ ]:
```
prediction = lr.predict(x_train)
plt.scatter(y_train,prediction)
```

In [ ]:
```
lr.score(x_test,y_test)
```

In [ ]:
```
lr.score(x_train,y_train)
```

In [ ]:
```
from sklearn.linear_model import Ridge,Lasso
```

In [ ]:
```
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
rr.score(x_test,y_test)
```

In [ ]:
```
la=Lasso(alpha=10)
la.fit(x_train,y_train)
la.score(x_test,y_test)
```

In [ ]:
```
from sklearn.linear_model import ElasticNet
en= ElasticNet()
en.fit(x_train,y_train)
```

In [ ]:
```
print(en.coef_)
```

In [ ]:
```
print(en.intercept_)
```

In [ ]:
```
prediction = en.predict(x_test)
prediction
```

In [ ]:
```
print(en.score(x_test,y_train))
```

In [ ]:
```
from sklearn import metrics
```

In [ ]:
```
print("Mean Absolute error:",metrics.mean_absolute_error(y_test,prediction))
```

In [ ]:
```
print("Mean Absolute Square error:",metrics.mean_squared_error(y_test,predicti
```

In [ ]:
```
print("Root mean Square error:",np.sqrt(metrics.mean_squared_error(y_test,pred
```

In [ ]:
```

```