Sukant Thombare 21102A0037 CMPN A

In [2]:
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [5]:
```python
data=pd.read_csv('housing.csv')
data
```

Out[5]:

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	households	median_
0	-122.23	37.88	41.0	880.0	129.0	322.0	126.0	
1	-122.22	37.86	21.0	7099.0	1106.0	2401.0	1138.0	
2	-122.24	37.85	52.0	1467.0	190.0	496.0	177.0	
3	-122.25	37.85	52.0	1274.0	235.0	558.0	219.0	
4	-122.25	37.85	52.0	1627.0	280.0	565.0	259.0	
...	...	...	...	...	...	...	...	
20635	-121.09	39.48	25.0	1665.0	374.0	845.0	330.0	
20636	-121.21	39.49	18.0	697.0	150.0	356.0	114.0	
20637	-121.22	39.43	17.0	2254.0	485.0	1007.0	433.0	
20638	-121.32	39.43	18.0	1860.0	409.0	741.0	349.0	
20639	-121.24	39.37	16.0	2785.0	616.0	1387.0	530.0	

20640 rows × 10 columns

In [6]:
```python
data.shape
```

Out[6]:
```
(20640, 10)
```

In [7]:
```python
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20640 entries, 0 to 20639
Data columns (total 10 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   longitude           20640 non-null  float64
 1   latitude            20640 non-null  float64
 2   housing_median_age  20640 non-null  float64
 3   total_rooms         20640 non-null  float64
 4   total_bedrooms      20433 non-null  float64
 5   population          20640 non-null  float64
 6   households          20640 non-null  float64
 7   median_income       20640 non-null  float64
 8   median_house_value  20640 non-null  float64
 9   ocean_proximity     20640 non-null  object 
dtypes: float64(9), object(1)
memory usage: 1.6+ MB
```

In [8]:
```python
data.isnull().sum()
```

Out[8]:
```
longitude               0
latitude                0
housing_median_age      0
total_rooms             0
total_bedrooms        207
population              0
households              0
median_income           0
median_house_value      0
ocean_proximity         0
dtype: int64
```

In [9]:
```python
data.dropna(inplace=True)
```

In [10]:
```python
data.isnull().sum()
```

Out[10]:
```
longitude             0
latitude              0
housing_median_age    0
total_rooms           0
total_bedrooms        0
population            0
households            0
median_income         0
median_house_value    0
ocean_proximity       0
dtype: int64
```

In [11]:
```python
data.reset_index(inplace=True,drop=True)
```

In [12]:
```python
data['ocean_proximity'].value_counts()
```

Out[12]:
```
<1H OCEAN     9034
INLAND        6496
NEAR OCEAN    2628
NEAR BAY      2270
ISLAND           5
Name: ocean_proximity, dtype: int64
```

In [13]:
```python
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
data['ocean_proximity']=le.fit_transform(data['ocean_proximity'])
```

In [14]:
```python
data["rooms_per_household"] = data["total_rooms"]/data["households"]
data["bedrooms_per_room"] = data["total_bedrooms"]/data["total_rooms"]
data["population_per_household"]=data["population"]/data["households"]
```

In [15]:
```python
data
```

Out[15]:

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	households	median_
0	-122.23	37.88	41.0	880.0	129.0	322.0	126.0	
1	-122.22	37.86	21.0	7099.0	1106.0	2401.0	1138.0	
2	-122.24	37.85	52.0	1467.0	190.0	496.0	177.0	
3	-122.25	37.85	52.0	1274.0	235.0	558.0	219.0	
4	-122.25	37.85	52.0	1627.0	280.0	565.0	259.0	
...	...	...	...	...	...	...	...	
20428	-121.09	39.48	25.0	1665.0	374.0	845.0	330.0	
20429	-121.21	39.49	18.0	697.0	150.0	356.0	114.0	
20430	-121.22	39.43	17.0	2254.0	485.0	1007.0	433.0	
20431	-121.32	39.43	18.0	1860.0	409.0	741.0	349.0	
20432	-121.24	39.37	16.0	2785.0	616.0	1387.0	530.0	

20433 rows × 13 columns

In [16]:

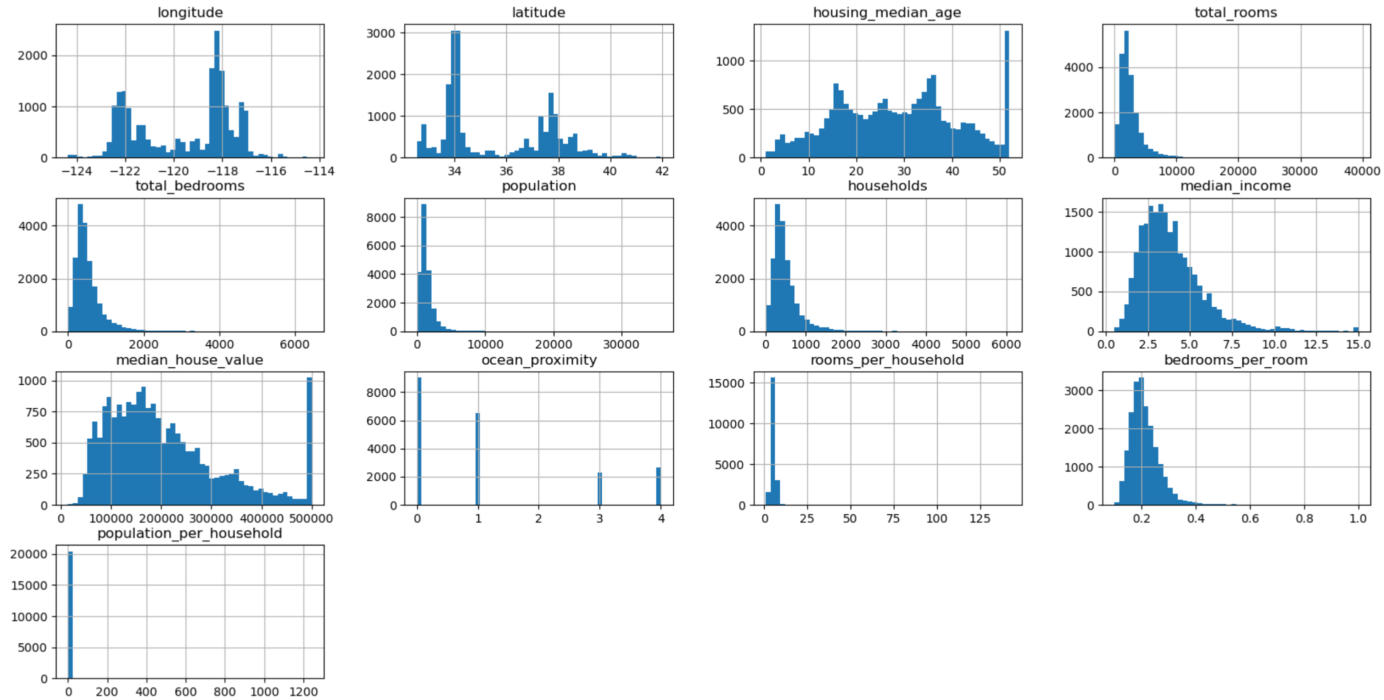
```
data.corr()
```

Out[16]:

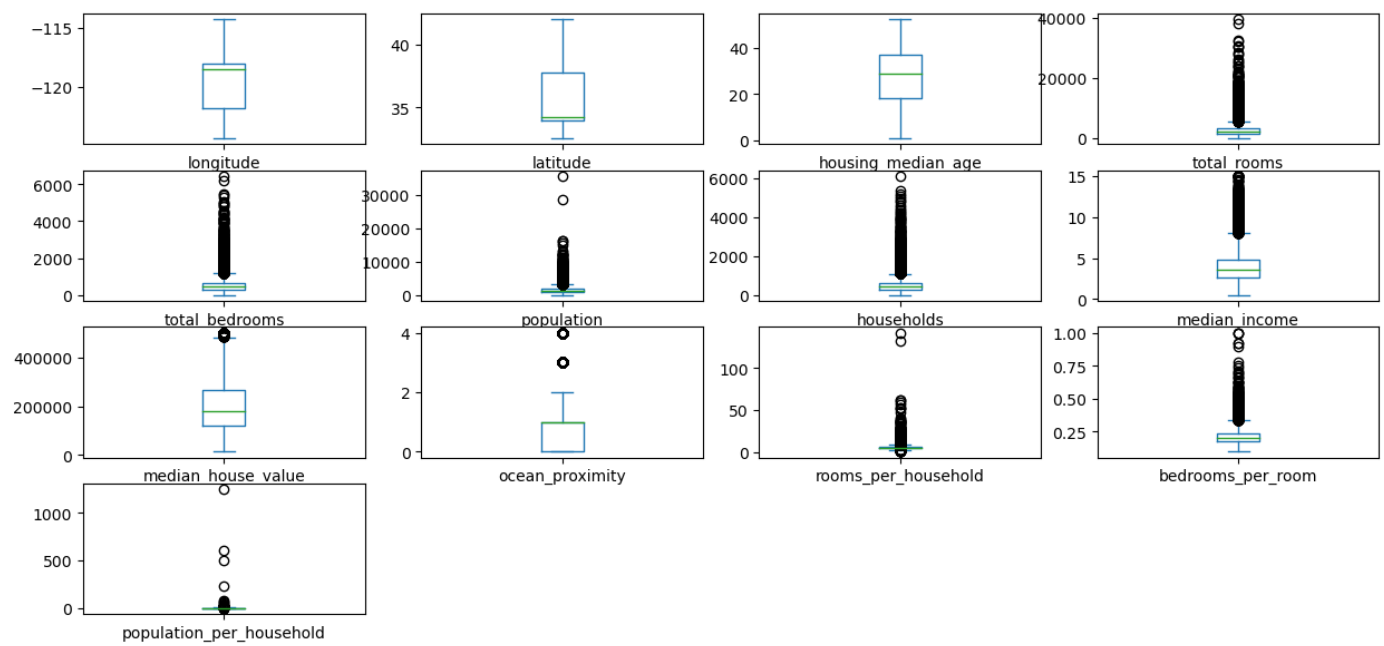
	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population
longitude	1.000000	-0.924616	-0.109357	0.045480	0.069608	0.100270
latitude	-0.924616	1.000000	0.011899	-0.036667	-0.066983	-0.108997
housing_median_age	-0.109357	0.011899	1.000000	-0.360628	-0.320451	-0.295787
total_rooms	0.045480	-0.036667	-0.360628	1.000000	0.930380	0.857281
total_bedrooms	0.069608	-0.066983	-0.320451	0.930380	1.000000	0.877747
population	0.100270	-0.108997	-0.295787	0.857281	0.877747	1.000000
households	0.056513	-0.071774	-0.302768	0.918992	0.979728	0.907186
median_income	-0.015550	-0.079626	-0.118278	0.197882	-0.007723	0.005087
median_house_value	-0.045398	-0.144638	0.106432	0.133294	0.049686	-0.025300
ocean_proximity	-0.289530	0.200801	0.112330	-0.015363	-0.014768	-0.069630
rooms_per_household	-0.027307	0.106423	-0.153031	0.133482	0.001538	-0.071898
bedrooms_per_room	0.092657	-0.113815	0.136089	-0.187900	0.084238	0.035319
population_per_household	0.002304	0.002522	0.013258	-0.024596	-0.028355	0.070062

In [17]:

```
data.hist(bins=50,figsize=(20,10));
```

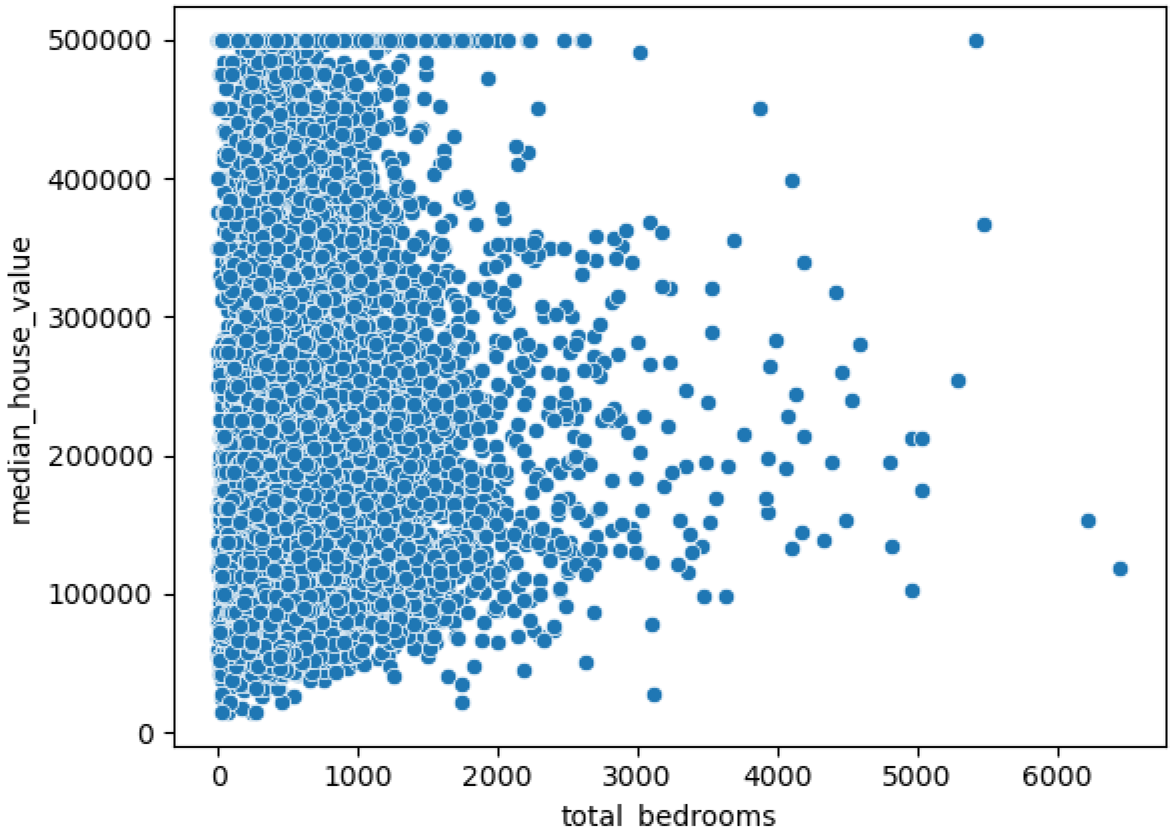


In [18]:
```
data.plot(kind='box', subplots=True, layout=(4,4), figsize=(15,7))
plt.show()
```

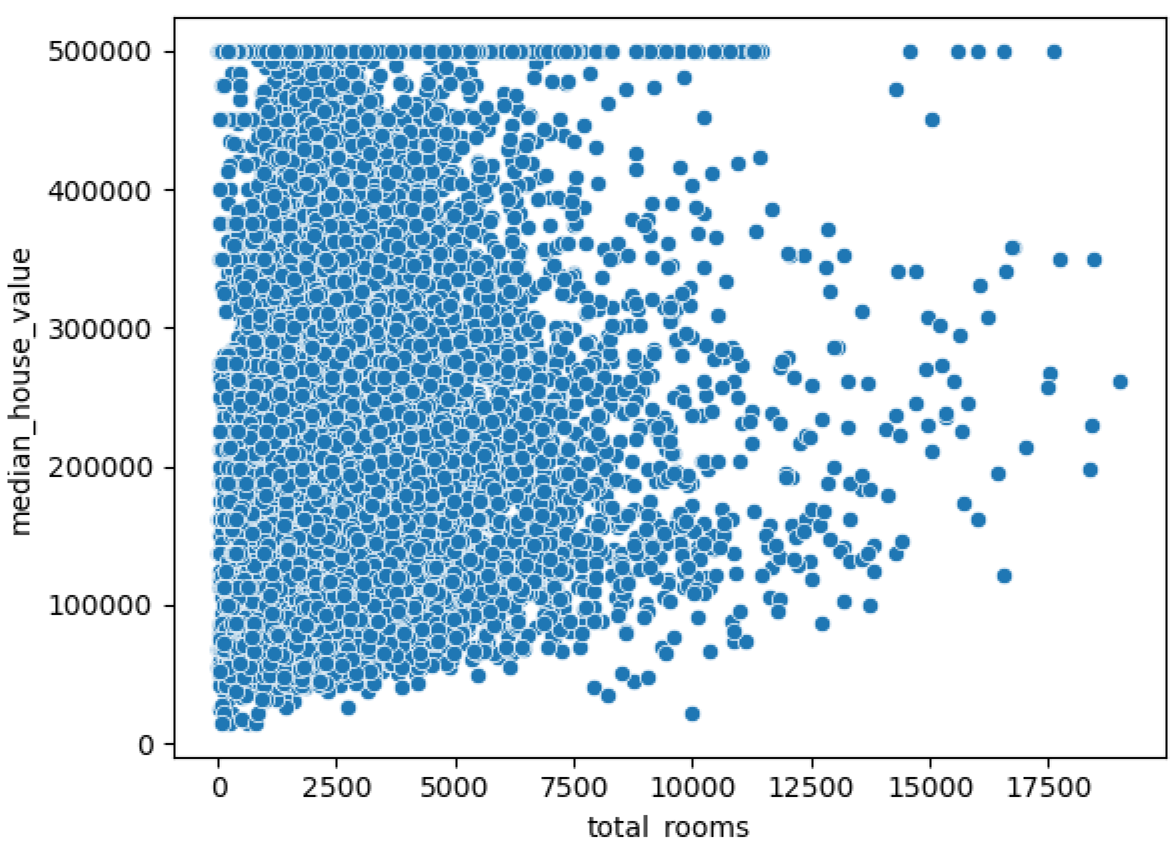


In [19]:
```
x=data.copy()
```

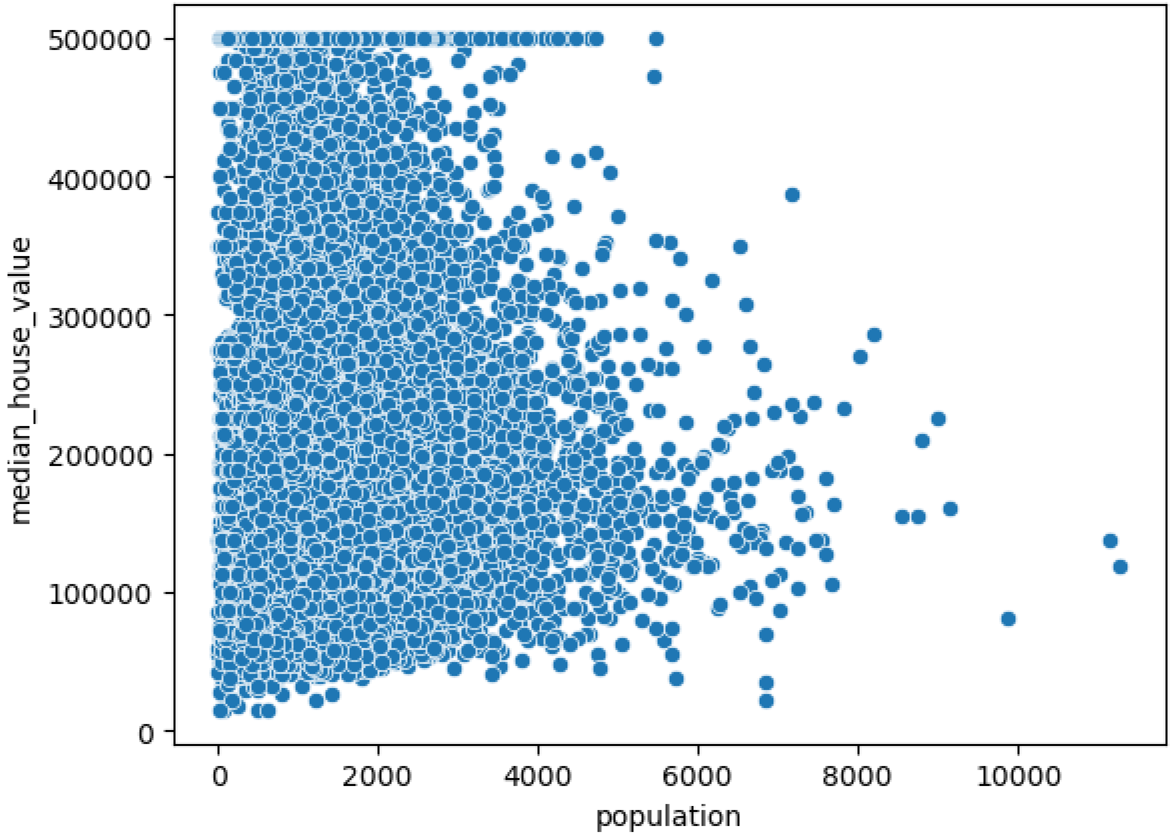
In [20]:
```
sns.scatterplot(x=x['total_bedrooms'],y=x['median_house_value'])
x=x[x['total_bedrooms']<2800]
```



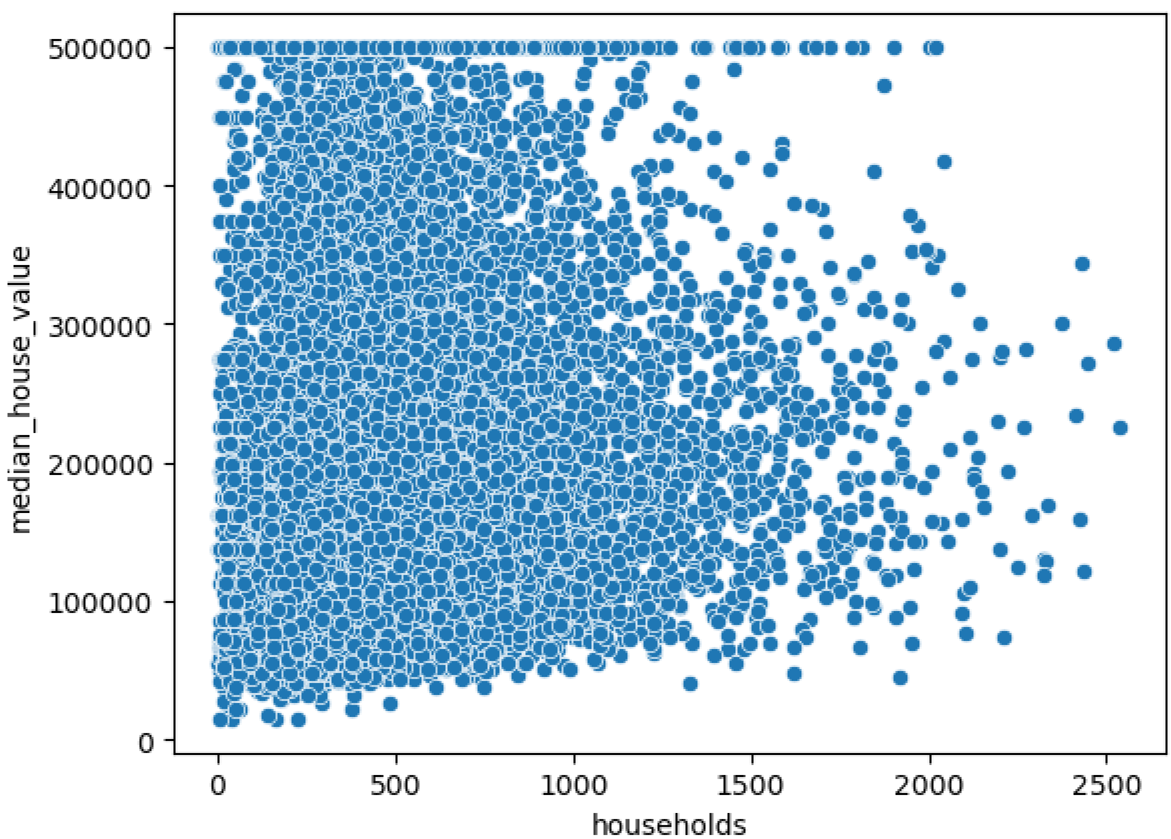
In [21]:
```
sns.scatterplot(x=x['total_rooms'],y=x['median_house_value'])
x=x[x['total_rooms']<15000]
```



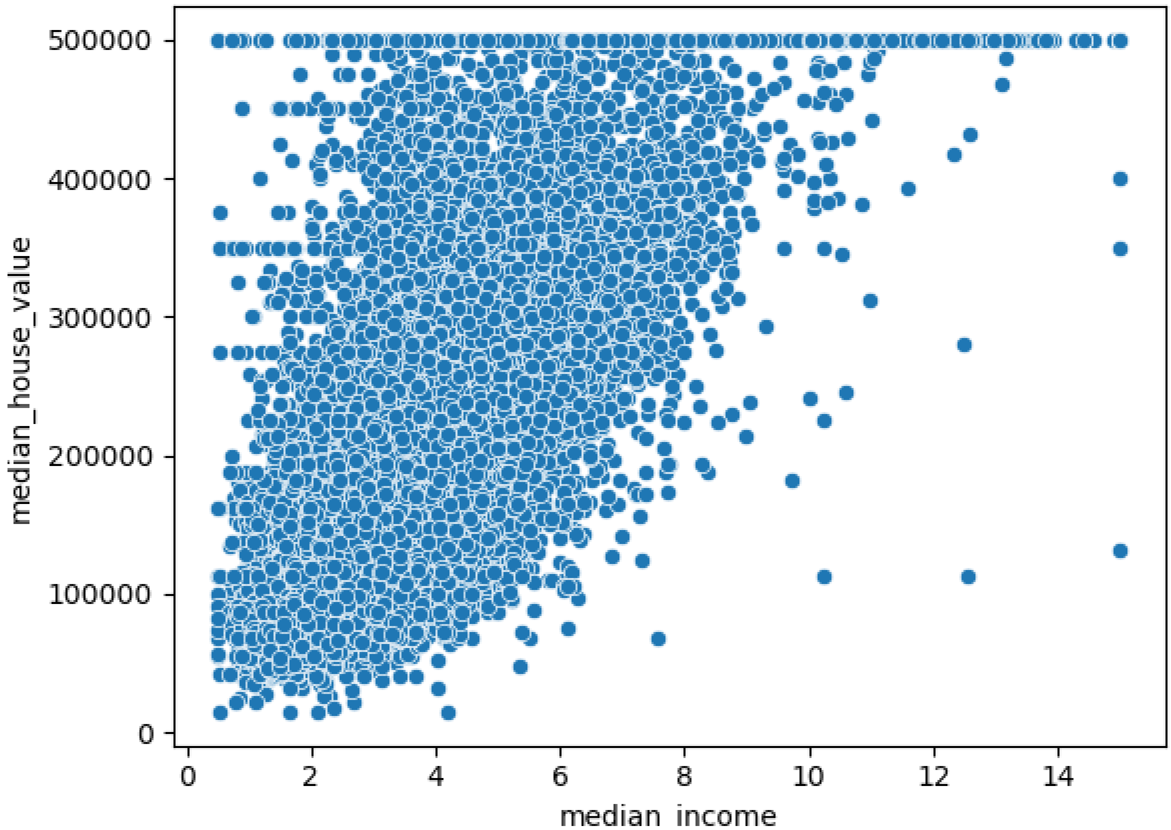
In [22]:
```
sns.scatterplot(x=x['population'],y=x['median_house_value'])
x=x[x['population']<6500]
```



In [23]:
```
sns.scatterplot(x=x['households'],y=x['median_house_value'])
x=x[x['households']<2000]
```



In [24]:
```
sns.scatterplot(x=x['median_income'],y=x['median_house_value'])
x=x[x['median_income']<9]
```

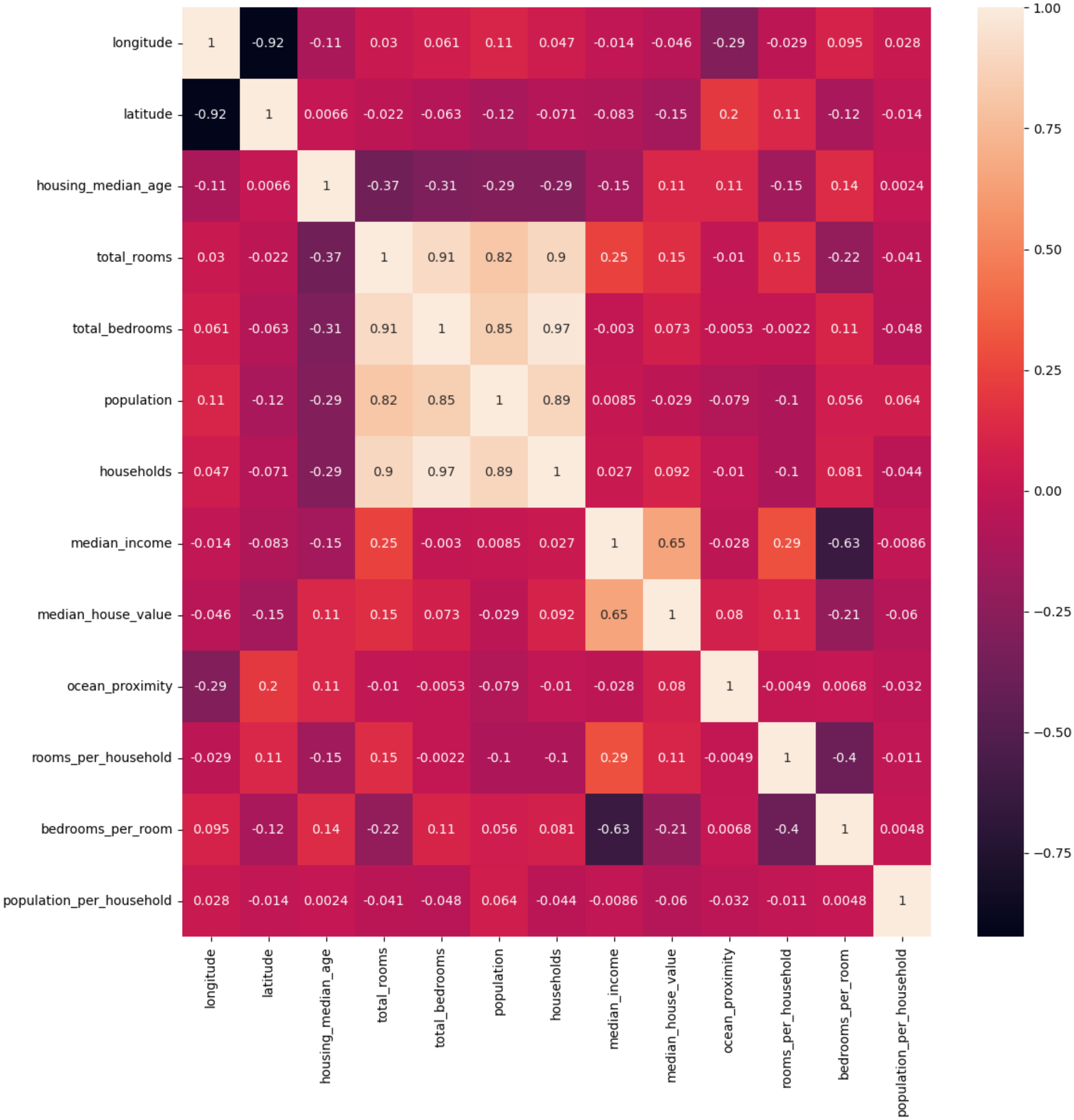


In [25]:
```
x.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 19805 entries, 0 to 20432
Data columns (total 13 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   longitude                 19805 non-null  float64
 1   latitude                  19805 non-null  float64
 2   housing_median_age        19805 non-null  float64
 3   total_rooms               19805 non-null  float64
 4   total_bedrooms            19805 non-null  float64
 5   population                19805 non-null  float64
 6   households                19805 non-null  float64
 7   median_income             19805 non-null  float64
 8   median_house_value        19805 non-null  float64
 9   ocean_proximity           19805 non-null  int32  
 10  rooms_per_household       19805 non-null  float64
 11  bedrooms_per_room         19805 non-null  float64
 12  population_per_household  19805 non-null  float64
dtypes: float64(12), int32(1)
memory usage: 2.0 MB
```

In [26]:
```
plt.figure(figsize=(13,13))
sns.heatmap(x.corr(),annot=True)
```

Out[26]:
```
<AxesSubplot:>
```


longitude 1 -0.92 -0.11 0.03 0.061 0.11 0.047 -0.014 -0.046 -0.29 -0.029 0.095 0.028
latitude -0.92 1 0.0066 -0.022 -0.063 -0.12 -0.071 -0.083 -0.15 0.2 0.11 -0.12 -0.014
housing_median_age -0.11 0.0066 1 -0.37 -0.31 -0.29 -0.29 -0.15 0.11 0.11 -0.15 0.14 0.0024
total_rooms 0.03 -0.022 -0.37 1 0.91 0.82 0.9 0.25 0.15 -0.01 0.15 -0.22 -0.041
total_bedrooms 0.061 -0.063 -0.31 0.91 1 0.85 0.97 -0.003 0.073 -0.0053 -0.0022 0.11 -0.048
population 0.11 -0.12 -0.29 0.82 0.85 1 0.89 0.0085 -0.029 -0.079 -0.1 0.056 0.064
households 0.047 -0.071 -0.29 0.9 0.97 0.89 1 0.027 0.092 -0.01 -0.1 0.081 -0.044
median_income -0.014 -0.083 -0.15 0.25 -0.003 0.0085 0.027 1 0.65 -0.028 0.29 -0.63 -0.0086
median_house_value -0.046 -0.15 0.11 0.15 0.073 -0.029 0.092 0.65 1 0.08 0.11 -0.21 -0.06
ocean_proximity -0.29 0.2 0.11 -0.01 -0.0053 -0.079 -0.01 -0.028 0.08 1 -0.0049 0.0068 -0.032
rooms_per_household -0.029 0.11 -0.15 0.15 -0.0022 -0.1 -0.1 0.29 0.11 -0.0049 1 -0.4 -0.011
bedrooms_per_room 0.095 -0.12 0.14 -0.22 0.11 0.056 0.081 -0.63 -0.21 0.0068 -0.4 1 0.0048
population_per_household 0.028 -0.014 0.0024 -0.041 -0.048 0.064 -0.044 -0.0086 -0.06 -0.032 -0.011 0.0048 1
longitude
latitude
housing_median_age
total_rooms
total_bedrooms
population
households
median_income
median_house_value
ocean_proximity
rooms_per_household
bedrooms_per_room
population_per_household
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75

In [27]:

```
x
```

Out [27]:

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	households	median_
0	-122.23	37.88	41.0	880.0	129.0	322.0	126.0	
1	-122.22	37.86	21.0	7099.0	1106.0	2401.0	1138.0	
2	-122.24	37.85	52.0	1467.0	190.0	496.0	177.0	
3	-122.25	37.85	52.0	1274.0	235.0	558.0	219.0	
4	-122.25	37.85	52.0	1627.0	280.0	565.0	259.0	
...	...	...	...	...	...	...	...	
20428	-121.09	39.48	25.0	1665.0	374.0	845.0	330.0	
20429	-121.21	39.49	18.0	697.0	150.0	356.0	114.0	
20430	-121.22	39.43	17.0	2254.0	485.0	1007.0	433.0	
20431	-121.32	39.43	18.0	1860.0	409.0	741.0	349.0	
20432	-121.24	39.37	16.0	2785.0	616.0	1387.0	530.0	

19805 rows × 13 columns

In [28]:
```
x_data = x.drop(["median_house_value" ,"rooms_per_household","bedrooms_per_room","popula
y_data = x["median_house_value"].values
```

In [29]:
```
from sklearn.preprocessing import PolynomialFeatures
feature = PolynomialFeatures(degree=3, include_bias=True, interaction_only=False)
x_data = feature.fit_transform(x_data)
```

In [30]:
```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler(copy=True, with_mean=True, with_std=True)
x_data = scaler.fit_transform(x_data)
```

In [31]:
```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train , y_test = train_test_split(x_data,y_data, test_size=0.25 , ran
```

In [32]:
```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
```

In [33]:
```
lr.fit(x_train,y_train)
```

Out [33]:
```
LinearRegression()
```

In [34]:
```
lr.score(x_train , y_train)
```

Out [34]:
```
0.7392356114861209
```

In [35]:
```
lr.score(x_test , y_test)
```

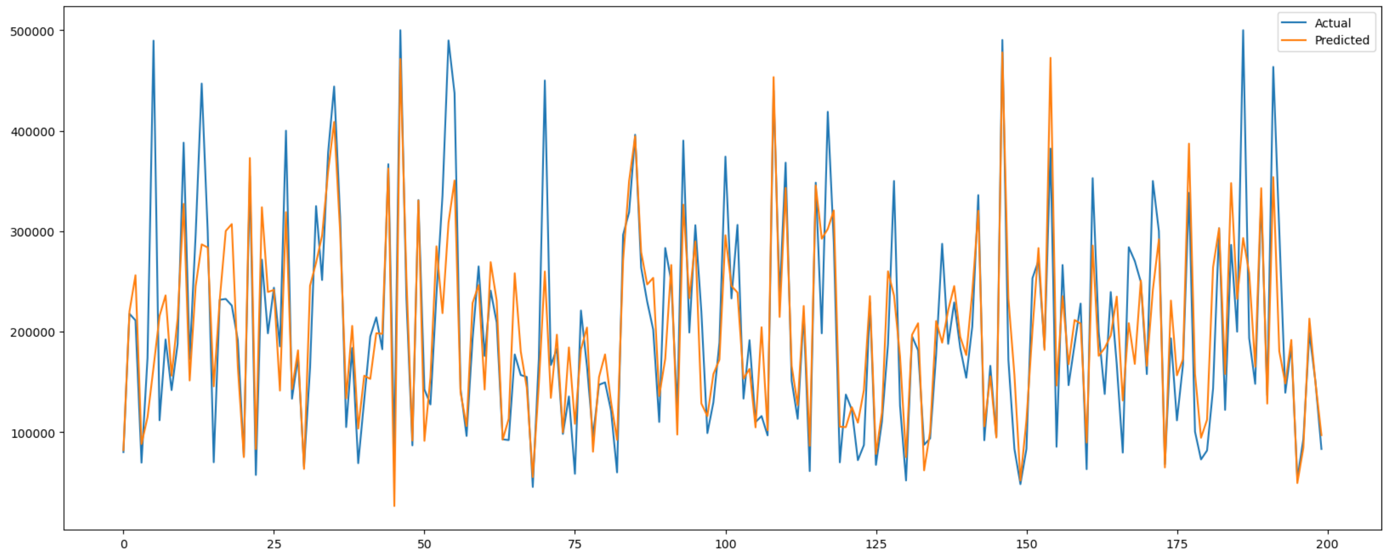
Out [35]:
```
0.7199506541756351
```

In [36]:
```
y_pred = lr.predict(x_test)
```

In [37]:
```
df = pd.DataFrame({"Y_test": y_test , "Y_pred" : y_pred})
```

In [38]:
```
plt.figure(figsize=(20,8))
plt.plot(df[:200])
plt.legend(["Actual" , "Predicted"])
```

Out [38]: <matplotlib.legend.Legend at 0x23780aeb970>



In [39]:
```
from sklearn.metrics import mean_squared_error, r2_score
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f'Mean Squared Error: {mse}')
print(f'R^2 Score: {r2}')
```

```
Mean Squared Error: 3397195426.647483
R^2 Score: 0.7199506541756351
```

In [43]:
```
input_features = pd.DataFrame({
    'longitude': [-121.09],
    'latitude': [39.48],
    'housing_median_age': [25.0],
    'total_rooms': [1665.0],
    'total_bedrooms': [374.0],
    'population': [845.0],
    'households': [330.0],
    'median_income': [1.5603],
    'ocean_proximity': [1]
})


input_data_poly = feature.transform(input_features)

input_data_scaled = scaler.transform(input_data_poly)

predicted_value = lr.predict(input_data_scaled)
```

```
C:\Users\saura\anaconda3\lib\site-packages\sklearn\base.py:443: UserWarning: X has featu
re names, but PolynomialFeatures was fitted without feature names
  warnings.warn(
```

In [42]:
```
print("Predicted Median House Value:", predicted_value[0])
```

```
Predicted Median House Value: 72727.9371930804
```

In [ ]: