# DS1_C6_Hackathon 1

## Level 0

In [2]:
```
#Importing necessary modules
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from tabulate import tabulate
```

In [6]:
```
df=pd.read_csv(r"D:\Data Science\Course 6\DS1_C6_S3_BazilHousing_Data_Hackat


```

In [7]:
```
df.head()
```

Out[7]:

| | city | area | rooms | bathroom | parking spaces | floor | animal | furniture | hoa (R$) | rent amount (R$) | property tax (R$) | insur |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | São Paulo | 70 | 2 | 1 | 1 | 7 | acept | furnished | 2065 | 3300 | 211 | |
| **1** | São Paulo | 320 | 4 | 4 | 0 | 20 | acept | not furnished | 1200 | 4960 | 1750 | |
| **2** | Porto Alegre | 80 | 1 | 1 | 1 | 6 | acept | not furnished | 1000 | 2800 | 0 | |
| **3** | Porto Alegre | 51 | 2 | 1 | 0 | 2 | acept | not furnished | 270 | 1112 | 22 | |
| **4** | São Paulo | 25 | 1 | 1 | 0 | 1 | not acept | not furnished | 0 | 800 | 25 | |

In [8]:
```
1 df.tail()
```

Out[8]:

| | city | area | rooms | bathroom | parking spaces | floor | animal | furniture | hoa (R$) | rent amount (R$) | property tax (R$) |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **10687** | Porto Alegre | 63 | 2 | 1 | 1 | 5 | not acept | furnished | 402 | 1478 | 24 |
| **10688** | São Paulo | 285 | 4 | 4 | 4 | 17 | acept | not furnished | 3100 | 15000 | 973 |
| **10689** | Rio de Janeiro | 70 | 3 | 3 | 0 | 8 | not acept | furnished | 980 | 6000 | 332 |
| **10690** | Rio de Janeiro | 120 | 2 | 2 | 2 | 8 | acept | furnished | 1585 | 12000 | 279 |
| **10691** | São Paulo | 80 | 2 | 1 | 0 | 0 | acept | not furnished | 0 | 1400 | 165 |

In [9]:
```
1 print(df.shape)
```

```
(10692, 13)
```

In [10]:
```
1 print(df.columns)
```

```
Index(['city', 'area', 'rooms', 'bathroom', 'parking spaces', 'floor',
       'animal', 'furniture', 'hoa (R$)', 'rent amount (R$)',
       'property tax (R$)', 'fire insurance (R$)', 'total (R$)'],
      dtype='object')
```

In [11]:
```
1 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10692 entries, 0 to 10691
Data columns (total 13 columns):
 #   Column               Non-Null Count  Dtype 
---  ------               --------------  ----- 
 0   city                 10692 non-null  object
 1   area                 10692 non-null  int64 
 2   rooms                10692 non-null  int64 
 3   bathroom             10692 non-null  int64 
 4   parking spaces       10692 non-null  int64 
 5   floor                10692 non-null  int64 
 6   animal               10692 non-null  object
 7   furniture            10692 non-null  object
 8   hoa (R$)             10692 non-null  int64 
 9   rent amount (R$)     10692 non-null  int64 
 10  property tax (R$)    10692 non-null  int64 
 11  fire insurance (R$)  10692 non-null  int64 
 12  total (R$)           10692 non-null  int64 
dtypes: int64(10), object(3)
memory usage: 1.1+ MB
```

In [12]:
```
1 df.isnull().sum()
```

Out[12]:
```
city                 0
area                 0
rooms                0
bathroom             0
parking spaces       0
floor                0
animal               0
furniture            0
hoa (R$)             0
rent amount (R$)     0
property tax (R$)    0
fire insurance (R$)  0
total (R$)           0
dtype: int64
```

# Level 1 :Analysis

In [14]:
```
def seprate_data_types(df):
    categorical = []
    continuous = []
    for column in df.columns:
        if df[column].nunique() < 100:

            categorical.append(column)
        else:
            continuous.append(column)

    return categorical, continuous
categorical, continuous = seprate_data_types(df)
from tabulate import tabulate
table = [categorical, continuous]
print(tabulate({"Categorical":categorical,
                "Continuous": continuous}, headers = ["categorical", "contin
```

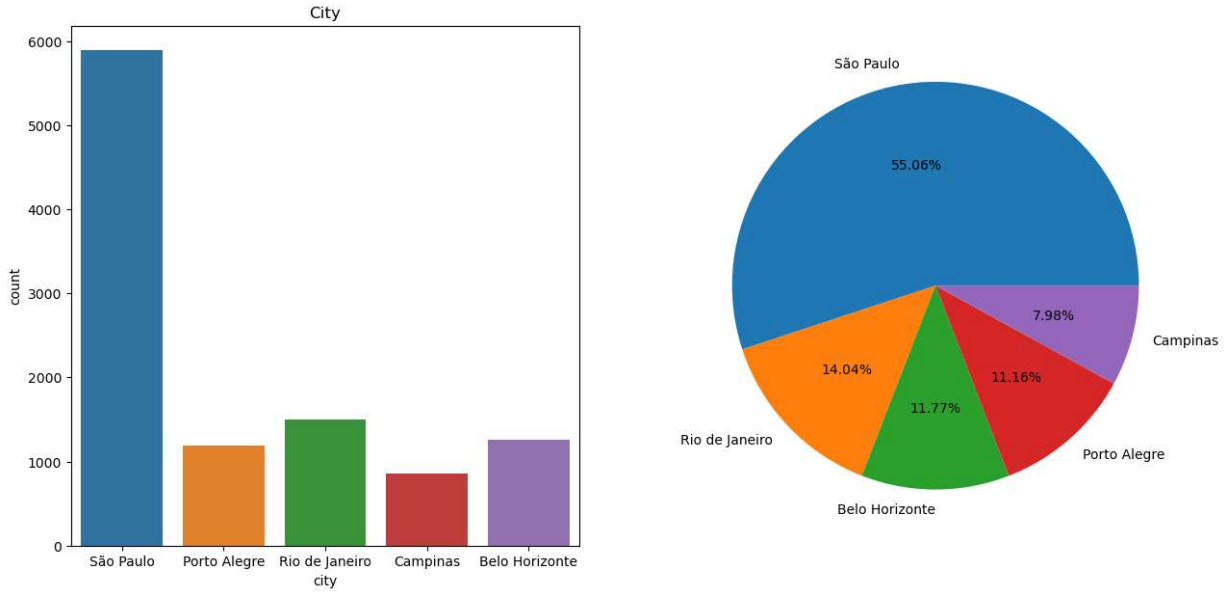
```
categorical     continuous
--------------  ------------------
city            area
rooms           hoa (R$)
bathroom        rent amount (R$)
parking spaces  property tax (R$)
floor           fire insurance (R$)
animal          total (R$)
furniture
```

In [15]:
```
def info_of_cat(col):
    print(f"Unique values in {col} are: {df[col].unique()}")
    print(f"Mode of {col} is {df[col].mode()[0]}")
    print(f"Number of missing values in {col} is {df[col].isnull().sum()}")
    if df[col].isnull().sum() > 0:
        print(f"\nThere are null values in the {col} column")

```

In [17]:
```
info_of_cat("city")
```

```
Unique values in city are: ['São Paulo' 'Porto Alegre' 'Rio de Janeiro' 'Campin
as' 'Belo Horizonte']
Mode of city is São Paulo
Number of missing values in city is 0
```

In [21]:
```
fig, ax = plt.subplots(1,2 ,figsize=(15,7))
ax[0].set_title("City")
percentage = df["city"].value_counts()
labels = list(df["city"].value_counts().index)

sns.countplot(x = df["city"], ax=ax[0])
plt.pie(percentage, labels = labels, autopct= "%0.2f%%" )
plt.show()
```



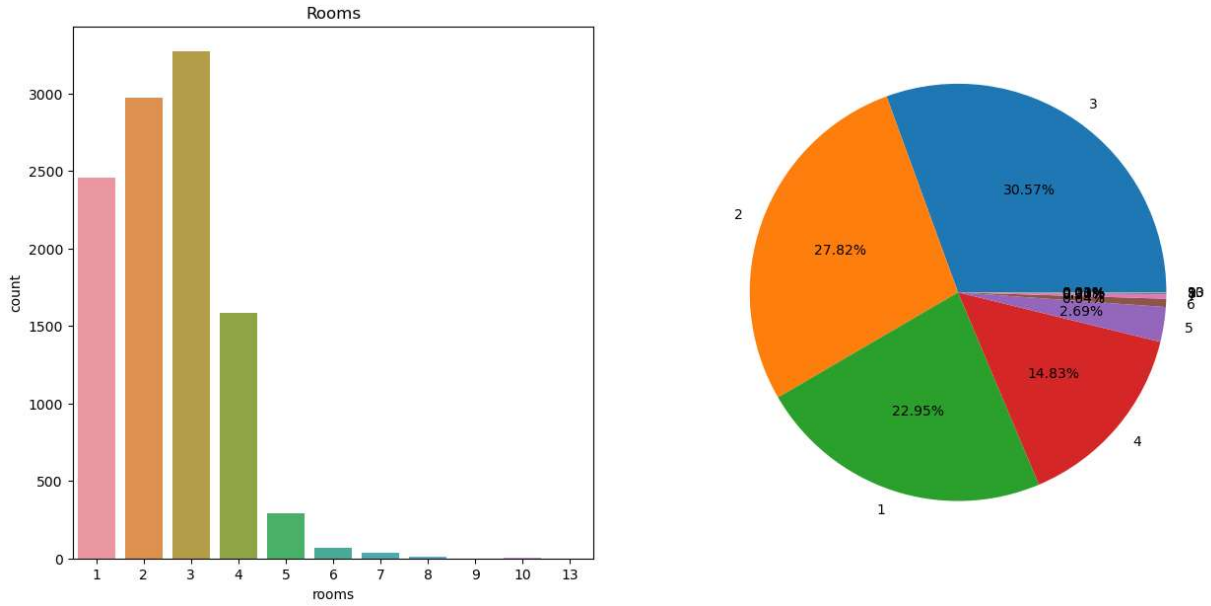

# Interpretation :

## From above graphs we can see that majority of employees are from Sao Paulo

In [23]:
```
info_of_cat("rooms")
```

```
Unique values in rooms are: [ 2  4  1  3  7  5  8  6 10 13  9]
Mode of rooms is 3
Number of missing values in rooms is 0
```

In [31]:
```
fig, ax = plt.subplots(1,2 ,figsize=(15,7))
ax[0].set_title("Rooms")
percentage = df["rooms"].value_counts()
labels = list(df["rooms"].value_counts().index)

sns.countplot(x = df["rooms"], ax=ax[0])
plt.pie(percentage, labels = labels, autopct= "%0.2f%%" )
plt.show()
```

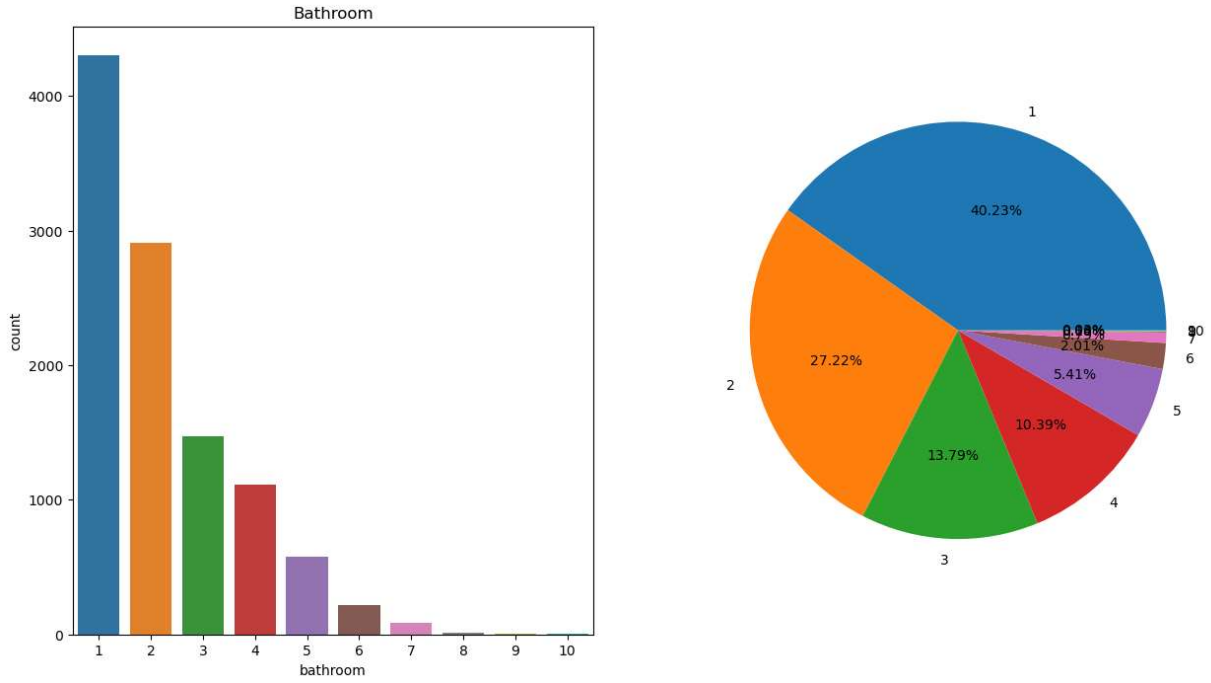



# Majority of the homes are 2 bedroom and 3 bedroom contributing to more than 58% of all composition of homes

In [32]:
```
info_of_cat("bathroom")
```

```
Unique values in bathroom are: [ 1  4  3  2  6  5  7  9  8 10]
Mode of bathroom is 1
Number of missing values in bathroom is 0
```

In [51]:
```
fig, ax = plt.subplots(1,2, figsize=(15,8))
ax[0].set_title("Bathroom")
percentage=df["bathroom"].value_counts()
labels = list(df["bathroom"].value_counts().index)

sns.countplot(x= df["bathroom"], ax = ax[0])
plt.pie(percentage,labels= labels, autopct="%0.2f%%")
plt.show()
```



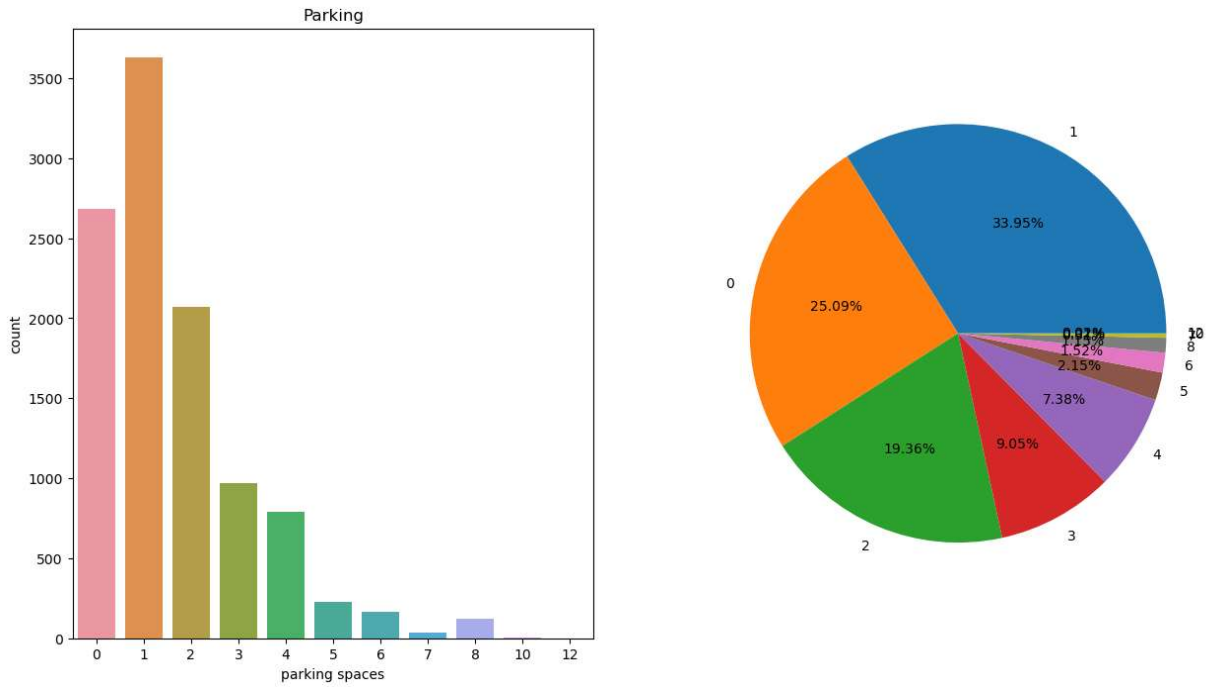

# Interpretation :

## Majority of homes have 1 and 2 bathrooms contributing 70% of the homes

In [37]:
```
info_of_cat("parking spaces")
```

```
Unique values in parking spaces are: [ 1  0  7  4  2  6  3  8  5 10 12]
Mode of parking spaces is 1
Number of missing values in parking spaces is 0
```

In [40]:
```
fig, ax=plt.subplots(1,2, figsize=(15,8))
ax[0].set_title("Parking")
percentage= df["parking spaces"].value_counts()
labels= list(df["parking spaces"].value_counts().index)

sns.countplot(x=df["parking spaces"], ax = ax[0])
plt.pie(percentage,labels=labels, autopct="%0.2f%%")
plt.show()
```




# Interpretation :

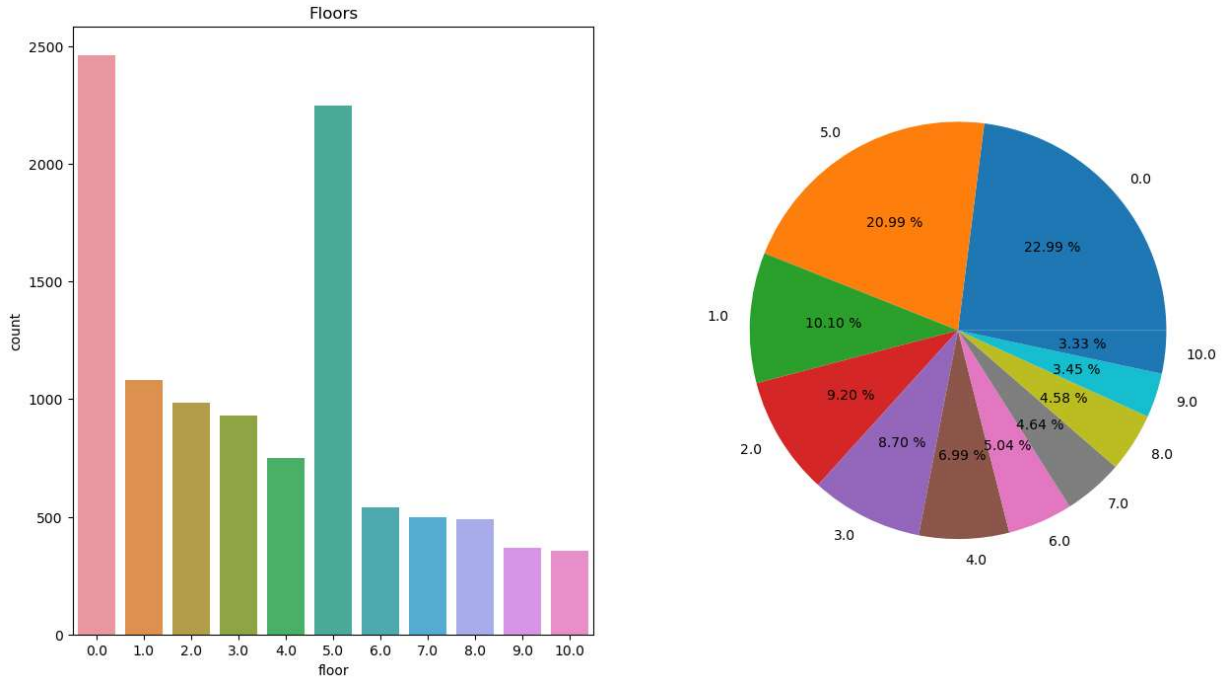
### Most of homes have only 1 parking space contributing to 35% and next to it nearly 25% of homes having no parking space at all

In [41]:
```
info_of_cat("floor")
```

```
Unique values in floor are: [  7  20   6   2   1   0   4   3  10  11  24   9
8  17  18   5  13  15
  16  14  26  12  21  19  22  27  23  35  25  46  28  29 301  51  32]
Mode of floor is 0
Number of missing values in floor is 0
```

In [61]:
```
mean = int(df["floor"].mean())
x = df[df["floor"] > 10].index
for index in x:
    df.loc[index, "floor"] = mean
```

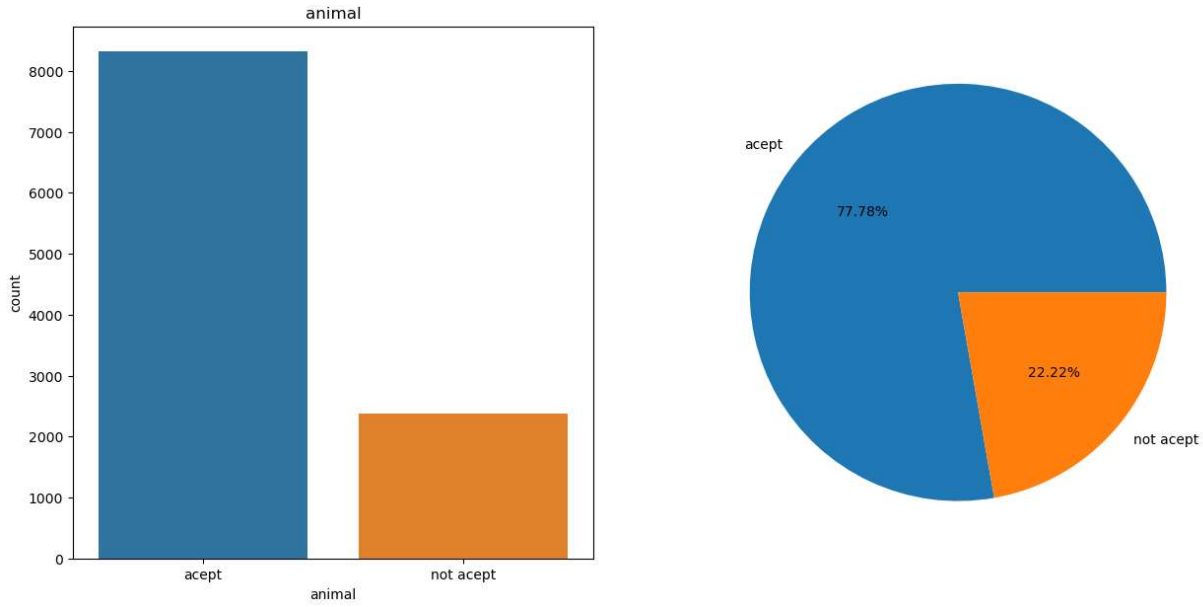
In [62]:
```
fig, ax = plt.subplots(1,2, figsize=(15,8))
ax[0].set_title("Floors")
percentage = df["floor"].value_counts()
labels = df["floor"].value_counts().index

sns.countplot(x= df["floor"], ax= ax[0])
plt.pie(percentage, labels = labels , autopct="%0.2f %%")
plt.show()
```




# Interpretation:

### In many homes there are no extra floors they all are villa .... most homes have 5 floors

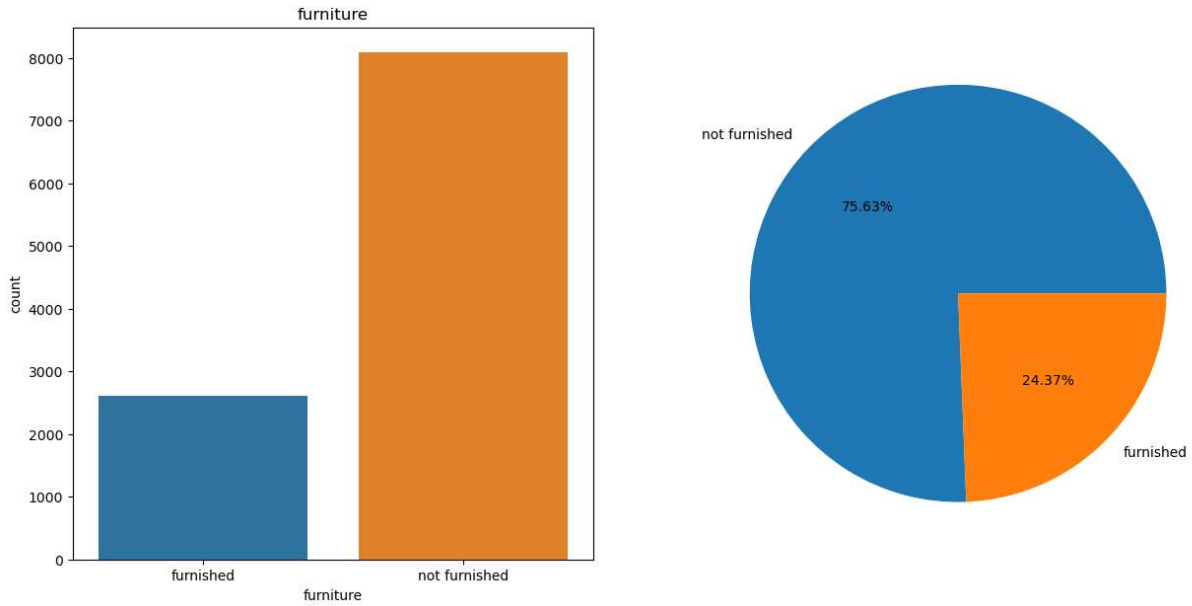
In [52]:
```
fig, ax = plt.subplots(1,2 ,figsize=(15,7))
ax[0].set_title("animal")
percentage = df["animal"].value_counts()
labels = list(df["animal"].value_counts().index)

sns.countplot(x = df["animal"], ax=ax[0])
plt.pie(percentage, labels = labels, autopct= "%0.2f%%" )
plt.show()
```




# Interpretation

## In almost all homes nearly 77% accept pets

In [53]:
```
fig, ax = plt.subplots(1,2 ,figsize=(15,7))
ax[0].set_title("furniture")
percentage = df["furniture"].value_counts()
labels = list(df["furniture"].value_counts().index)

sns.countplot(x = df["furniture"], ax=ax[0])
plt.pie(percentage, labels = labels, autopct= "%0.2f%%" )
plt.show()
```




## Interpretation :

**Nearly 75% homes are unfurnished and 25% are furnished**

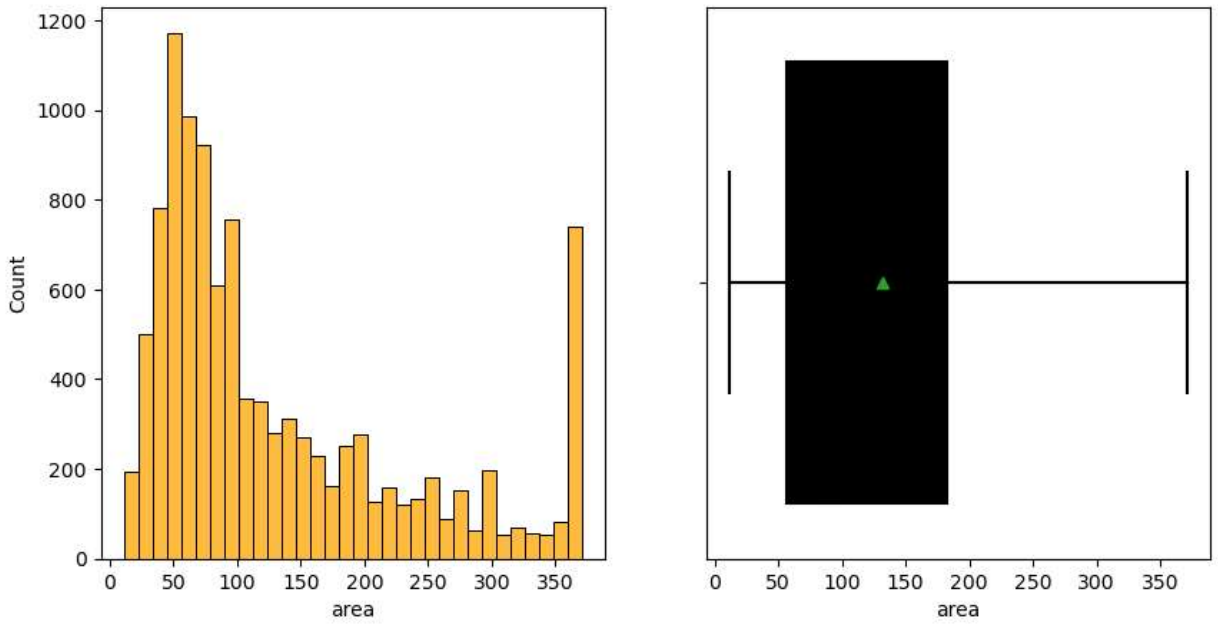
# Level 1 Analysis for numerical data

In [77]:
```
def num_level1(df,col):
    print(f"The mean of the {col} is {df[col].mean()}")
    print(f"The median of the {col} is {df[col].median()}")
    print(f"The mode of the {col} is {df[col].mode()[0]}")
    print(f"The standard deviation of the {col} is {df[col].std()}")
    print(f"Number of missing values in the {col} is {df[col].isnull().sum()
    fig, ax = plt.subplots(1, 2, figsize= (10,5))
    sns.histplot(x = df[col], ax =ax[0], color = "orange")
    sns.boxplot(x = df[col], ax = ax[1], color = "black",showmeans=True)
    plt.show()

def outlier_treatment(dataframe,columns):
    for item in columns:
        percentile25 = dataframe[item].quantile(0.25)
        percentile75 = dataframe[item].quantile(0.75)
        iqr=percentile75-percentile25
        upper_limit = percentile75 + 1.5 * iqr
        lower_limit = percentile25 - 1.5 * iqr
        dataframe[item] = np.where(dataframe[item] > upper_limit,upper_limit
        np.where(dataframe[item] < lower_limit,lower_limit,dataframe[item]))
    return dataframe


```

In [65]:
```
df=outlier_treatment(df,continuous)
```

In [78]:
```
num_level1(df,continuous[0])
```

```
The mean of the area is 132.0876356154134
The median of the area is 90.0
The mode of the area is 371.0
The standard deviation of the area is 101.33092381207521
Number of missing values in the area is 13
```

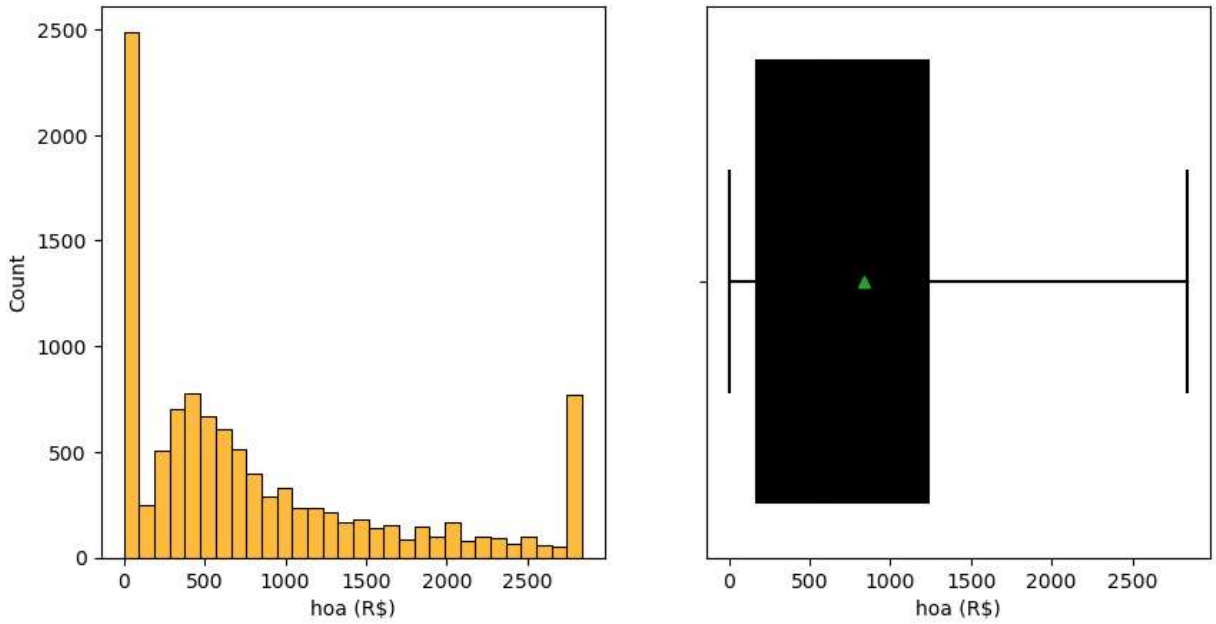


# Interpretation:

## We can see the boxplot looks clean of outliers as well as majority of data lie in 0 to 150sqft

In [79]:
```
1 num_level1(df,continuous[1])
```

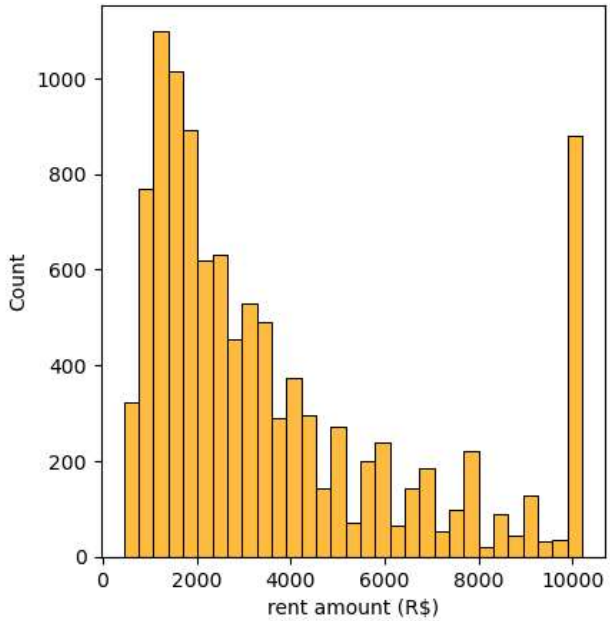
```
The mean of the hoa (R$) is 836.9882856341189
The median of the hoa (R$) is 560.0
The mode of the hoa (R$) is 0.0
The standard deviation of the hoa (R$) is 856.598027516404
Number of missing values in the hoa (R$) is 13
```

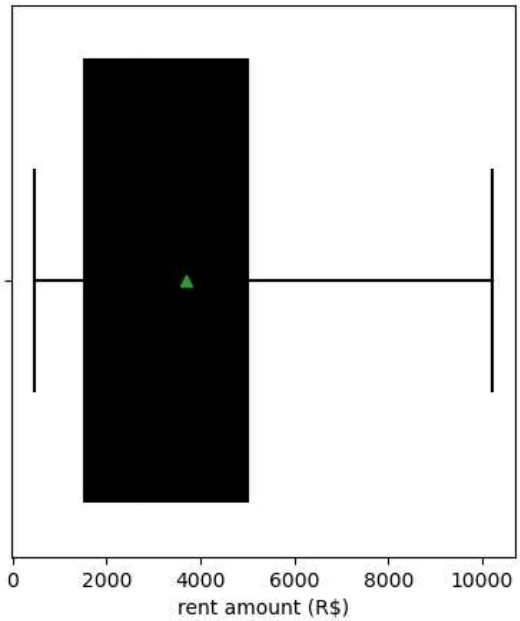



In [ ]:
```
1
```

In [80]:
```
1 num_level1(df,continuous[2])
```

```
The mean of the rent amount (R$) is 3688.2547699214365
The median of the rent amount (R$) is 2661.0
The mode of the rent amount (R$) is 10205.0
The standard deviation of the rent amount (R$) is 2821.8628993304974
Number of missing values in the rent amount (R$) is 13
```



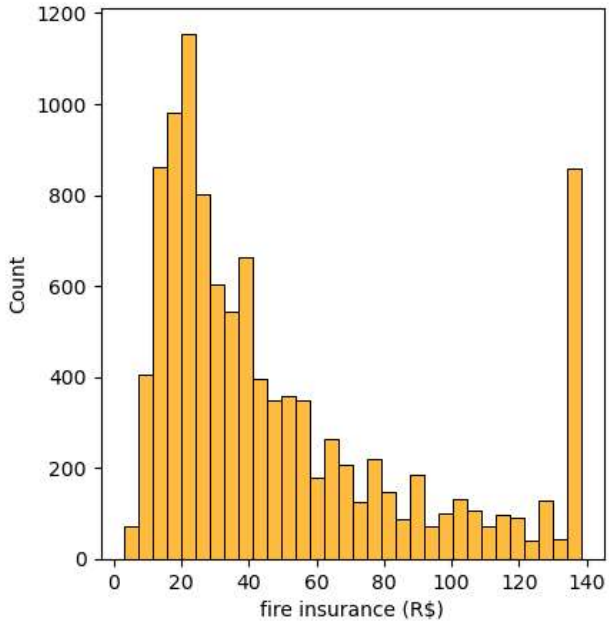



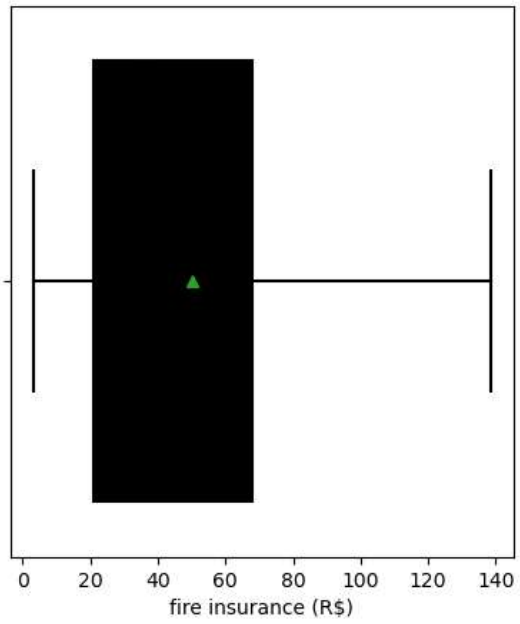

# Interpretation :

### From the above charts its clear that all the Hoa rates lie in 100 to 1250 it has normal distribution

In [82]:
```
1 num_level1(df,continuous[4])
```

```
The mean of the fire insurance (R$) is 50.107510288065846
The median of the fire insurance (R$) is 36.0
The mode of the fire insurance (R$) is 138.5
The standard deviation of the fire insurance (R$) is 38.614564862056085
Number of missing values in the fire insurance (R$) is 13
```



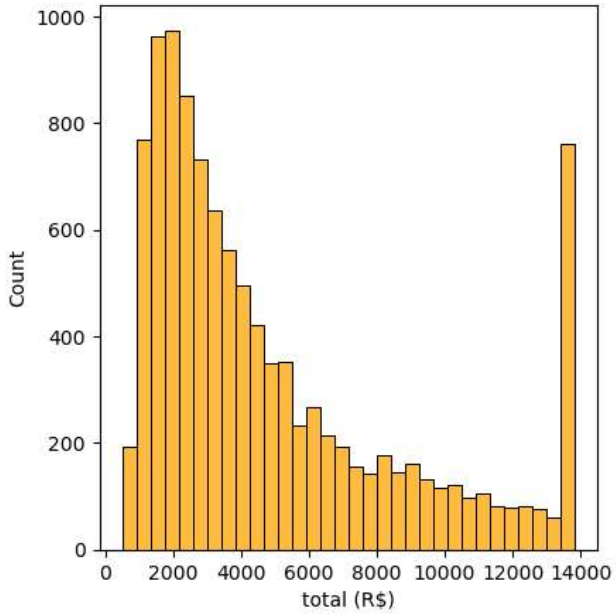



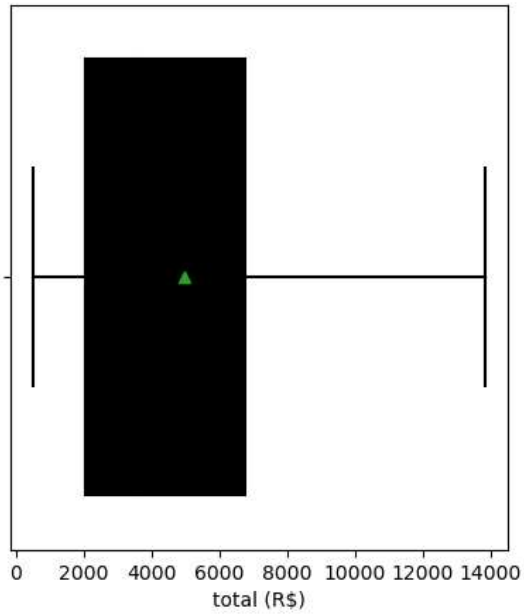

# Interpretation:

**we can see that the highest fire insurance ever claimed is 140 which is claimed also by considerably large people and many claims lie in range 10 to 60 dollars**

In [83]:
```
num_level1(df,continuous[5])
```

```
The mean of the total (R$) is 4966.518308080808
The median of the total (R$) is 3581.5
The mode of the total (R$) is 13827.375
The standard deviation of the total (R$) is 3794.8994208776344
Number of missing values in the total (R$) is 13
```



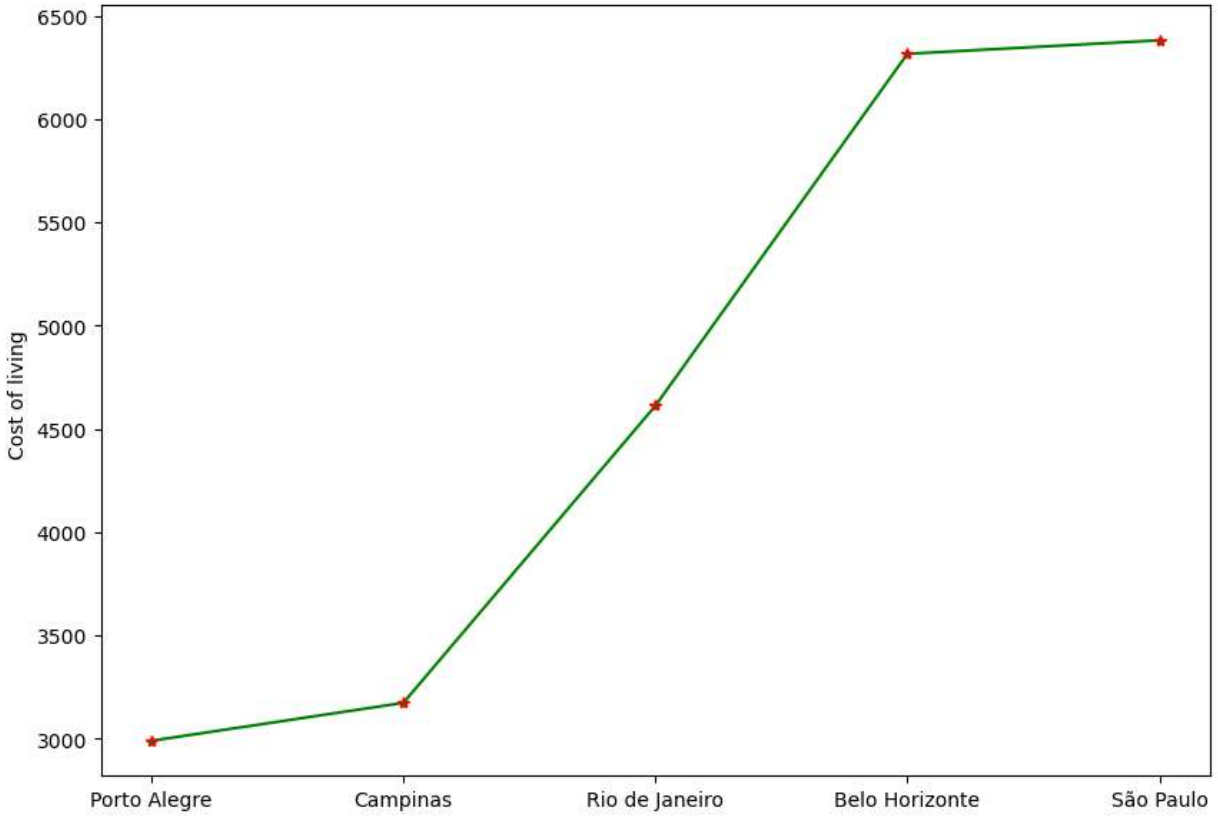

## Interpretation:

**The highest cost ever is 14000 dollars for a home while most homes cost around 2500 to 5000 dollars**

# Level 2: Bivariate Analysis (Getting closer to the BIG QUESTION

# Total cost of living vs City

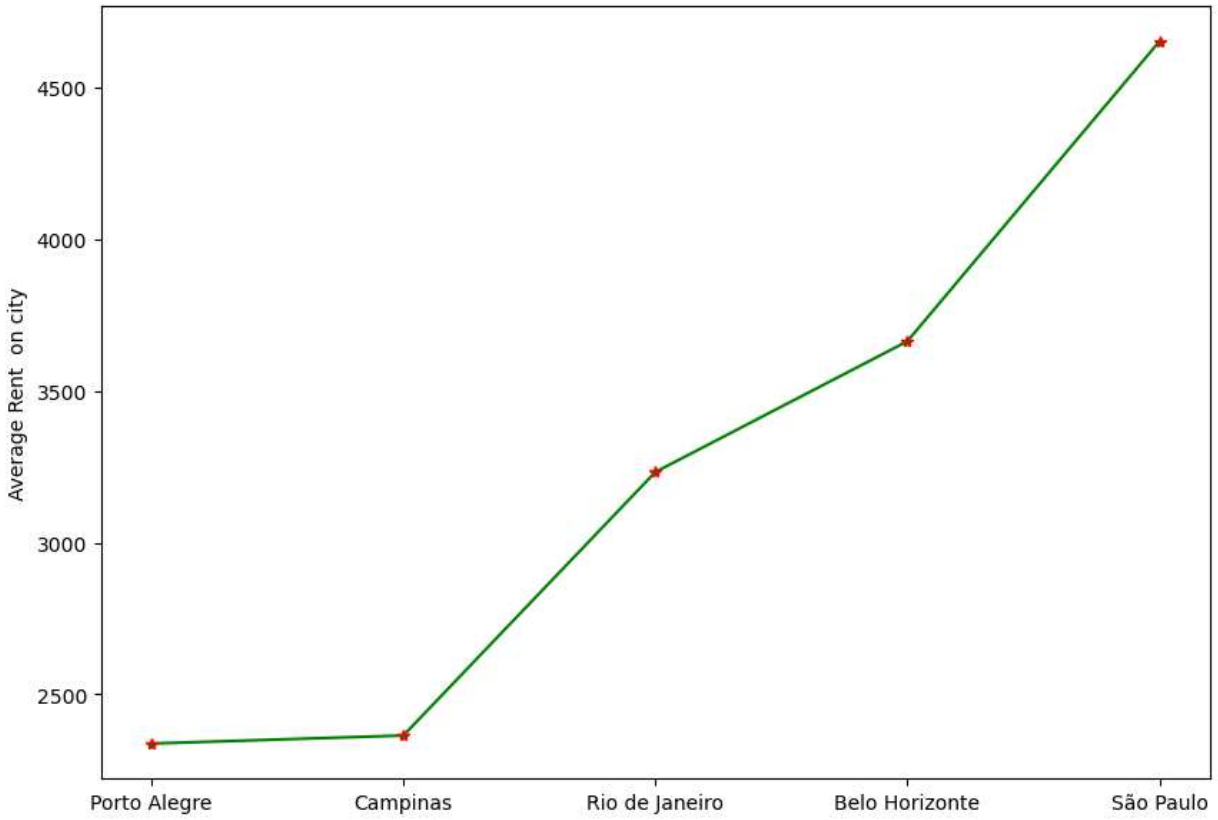
In [87]:
```
fig, ax = plt.subplots(figsize = (10, 7))
city_col=empdf.groupby("city").mean()["total (R$)"].sort_values()
plt.plot(city_col, data = empdf,color="green",marker="*", markeredgecolor="r
plt.ylabel("Cost of living ")
plt.show()
```




In [ ]:
```
Interpretation:
From the above analysis we can see the cost of living is the highest in the
```

In [ ]:
```
Total rent amount vs city
```

In [89]:
```
fig, ax = plt.subplots(figsize = (10, 7))
city_col=empdf.groupby("city").mean()["rent amount (R$)"].sort_values()
plt.plot(city_col, data = empdf,color="green",marker="*", markeredgecolor="r
plt.ylabel("Average Rent  on city")
plt.show()
```

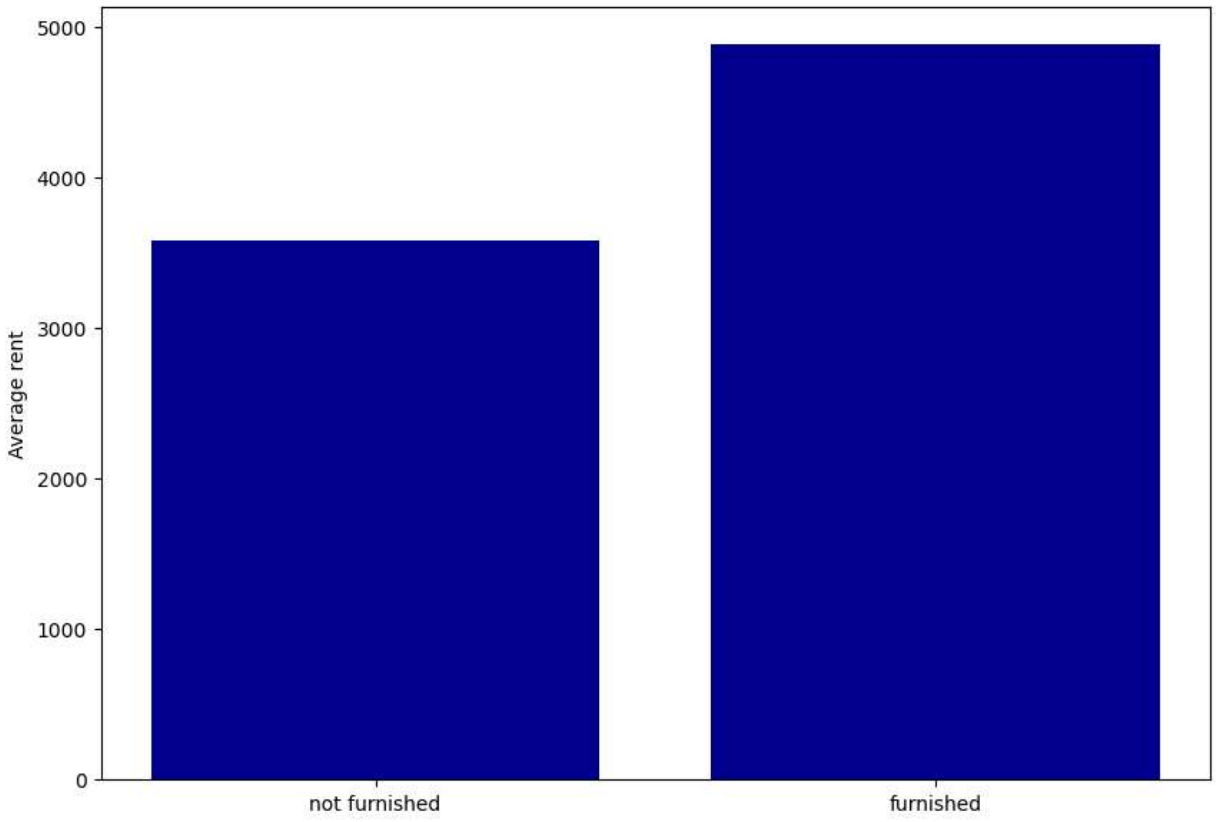



# Interpretation:

### We can see that again Sai Paulo is the city with highest cost of living

# Furnishment vs average rent

In [91]:
```
fig, ax = plt.subplots(figsize = (10, 7))
fur_col=empdf.groupby("furniture").mean()["rent amount (R$)"].sort_values()
plt.bar(fur_col.index,fur_col,color="darkblue")
plt.ylabel("Average rent")
plt.show()
```




## Interpretation:

**We can see that amonst all homes furnished homes have higher rent than unfurnished homes**

# Level 3 - analysis

**One could consider analyzing all the above columns for the customers who have left and having 2 or 3 dependents. However it could be a meaningless visualization, hence it is better to consult the domain expert to choose the appropriate columns for further analysis.**

1) rental amount 2) property tax 3) rooms

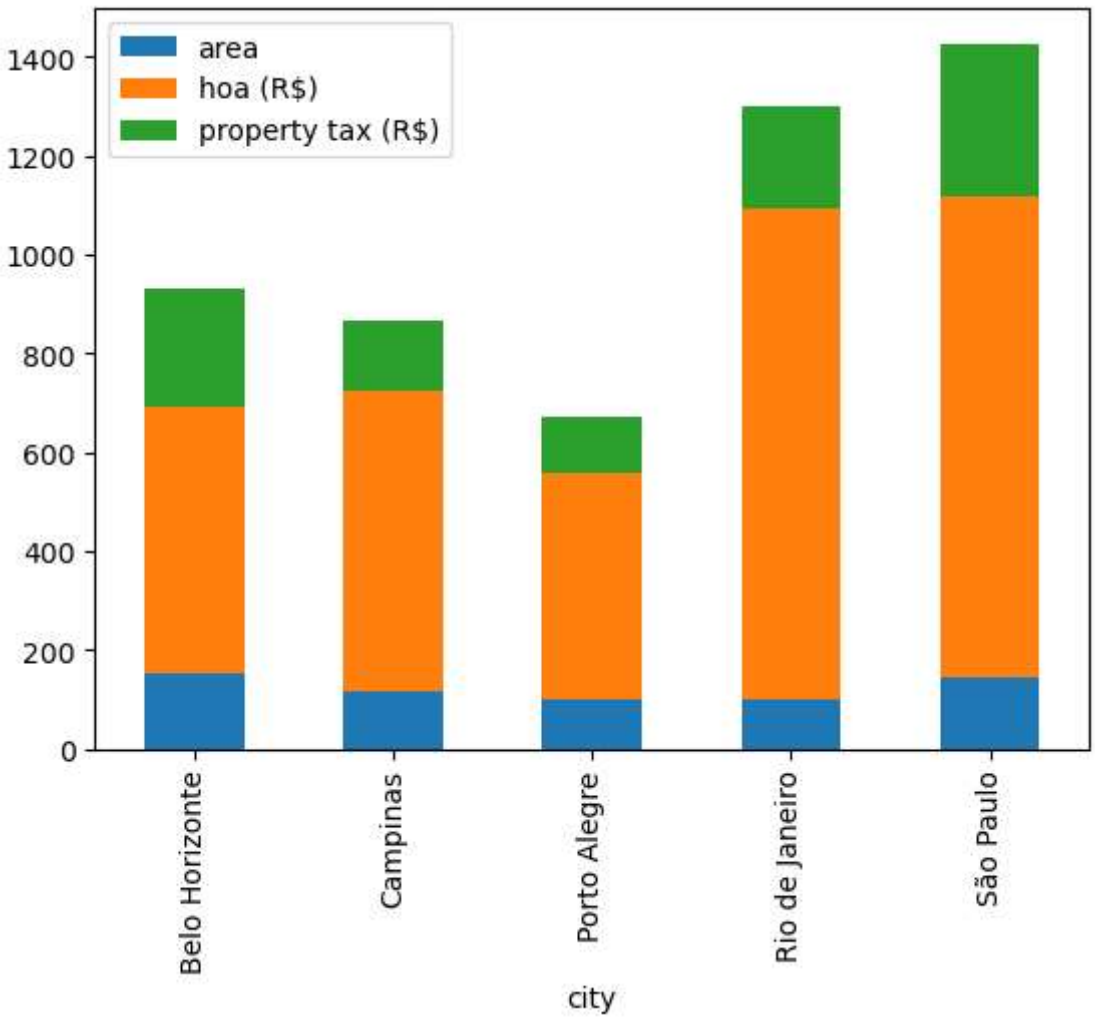
In [28]:
```
print(tabulate({"Categorical":categorical,"continuous": numerical},headers =
```

```
categorical     numerical
--------------  ------------------
city            area
rooms           hoa (R$)
bathroom        rent amount (R$)
parking spaces  property tax (R$)
floor           fire insurance (R$)
animal          total (R$)
furniture
```

In [39]:
```
homes=empdf.groupby(["city"]).mean().loc[:,["area","hoa (R$)","property tax
```

In [40]:
```
homes.plot(kind='bar', stacked=True)
```

Out[40]: <AxesSubplot:xlabel='city'>

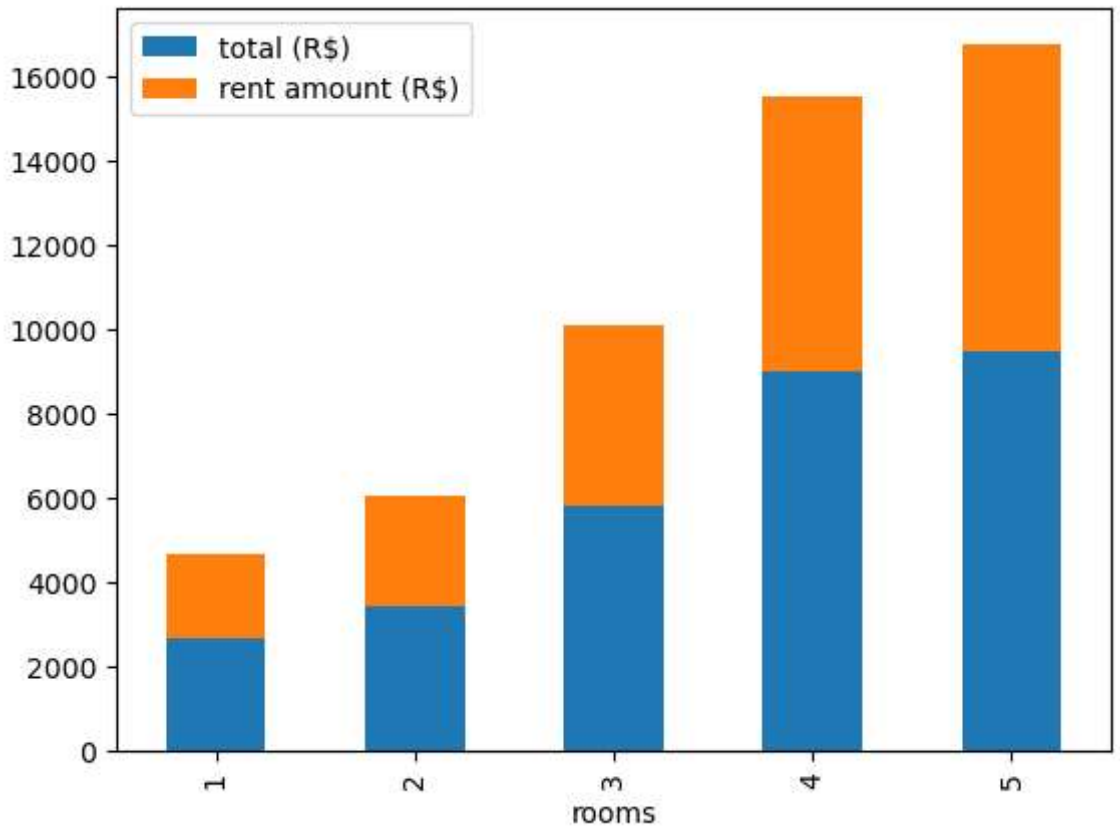



# Interpretation :

**We can see that on overall analysis of area and hoa and property tax Sao paulo and Rio de Janerio have the highest expense factor**

## Rooms vs total cost

In [38]:
```
empdf.groupby("rooms").mean().loc[:,["total (R$)","rent amount (R$)"]].plot(
plt.show()
```




## Interpretations:

**5&4 rooms have really high total cost of living compared to all other room levels**