Importing scraper and necessary libraries

In [ ]:
```
from scraper import listing_df
import pandas as pd
import matplotlib.pyplot as plt
from datetime import date
import seaborn as sns
import webbrowser as wb
```

Initializing Dataframe

In [ ]:
```
df = listing_df
```

Checking and dropping NaN values if any

In [ ]:
```
print(df.isna().sum())
```

```
prop_ID       0
rent_pw       0
suburb        0
avail_date    0
bills_inc     0
prop_pop      0
beds          0
baths         0
dtype: int64
```

Describing dataset

In [ ]:
```
df.describe()
```

Out[ ]:

	prop_ID	rent_pw	suburb	avail_date	bills_inc	prop_pop	beds	baths
count	252	252	252	252	252	252	252	252
unique	252	72	144	39	2	8	6	4
top	1-bed-sydney-epping-2121-P1270615	350	Randwick	Available Now	Yes	2	3	2
freq	1	22	7	159	184	56	66	115

Checking for duplicates

In [ ]:
```
try:
    print(pd.concat(x for _, x in df.groupby("prop_ID") if len(x) > 1))
except ValueError as err:
    print("No duplicates found!")
```

```
No duplicates found!
```

Removing duplicates if any

In [ ]:
```
df.drop_duplicates(inplace=True)
```

Formatting columns to relevant types

In [ ]:
```
 # Removing commas from the rent column
df["rent_pw"] = df["rent_pw"].replace(",","",regex=True)

# Converting beds, baths, rent_pw, prop_pop columns to int
df[["prop_pop","beds","baths","rent_pw"]] = df[["prop_pop","beds","baths","re

# Removing "Available " from avail_date column
df["avail_date"] = df["avail_date"].replace("Available ","",regex=True)

# Replacing "Now" with today's date
df.loc[(df["avail_date"]=="Now"), "avail_date"] = date.today()

# Converting avail_date column to datetime data type
df["avail_date"] = pd.to_datetime(df["avail_date"])
```

Getting property type from the `prop_ID` column

In [ ]:
```
# Initializing a property type column from the prop_ID column
prop_type = df["prop_ID"].str.split("-sydney", n=1, expand=True)

# Setting prop_type column first array element (first split)
df["prop_type"] = prop_type[0]

# Replacing "-" with spaces to clean up column
df["prop_type"] = df["prop_type"].replace("-"," ",regex=True)

# Capitalizing column values
df["prop_type"] = df["prop_type"].str.capitalize()
```

Describing dataset after converting columns to relevant data types

In [ ]:
```
df.describe()
```

Out[ ]:

	rent_pw	prop_pop	beds	baths
count	252.000000	252.000000	252.000000	252.000000
mean	392.765873	2.730159	3.297619	2.083333
std	527.776140	1.784898	1.429176	0.881980
min	70.000000	0.000000	1.000000	1.000000
25%	260.000000	1.000000	2.000000	1.000000
50%	320.000000	3.000000	3.000000	2.000000
75%	426.250000	4.000000	4.000000	3.000000
max	8000.000000	7.000000	6.000000	4.000000

In [ ]:
```
def url_creator(id):
    return f'https://flatmates.com.au/{id}'

df
```

Out[ ]:

	prop_ID	rent_pw	suburb	avail_date	bills_inc	prop_pop	beds	baths	prop_type
0	share-house-sydney-lavender-bay-2060-P1240929	540	Lavender Bay	2023-01-02	Yes	2	2	2	Share house
1	share-house-sydney-gymea-2227-P961141	275	Gymea	2022-12-22	No	1	3	1	Share house
2	studio-sydney-concord-west-2138-P1278107	425	Concord West	2022-12-22	Yes	0	1	1	Studio
3	whole-property-sydney-north-ryde-2113-P1092185	400	North Ryde	2022-12-22	Yes	0	3	3	Whole property
4	granny-flat-sydney-winston-hills-2153-P1263162	380	Winston Hills	2022-12-22	No	0	1	1	Granny flat
...	...	...	...	...	...	...	...	...	...
247	share-house-sydney-sydney-olympic-park-2127-P9...	350	Sydney Olympic Park	2023-01-06	No	1	2	2	Share house
248	share-house-sydney-maroubra-2035-P688098	240	Maroubra	2022-12-22	Yes	4	4	2	Share house
249	share-house-sydney-kareela-2232-P620370	300	Kareela	2022-12-22	No	2	5	4	Share house
250	share-house-sydney-coogee-2034-P1279352	430	Coogee	2022-12-22	Yes	2	2	1	Share house

	prop_ID	rent_pw	suburb	avail_date	bills_inc	prop_pop	beds	baths	prop_type
251	share-house-sydney-coogee-2034-P1275670	430	Coogee	2022-12-22	Yes	2	2	1	Share house

252 rows × 9 columns

Checking distribution of rent values

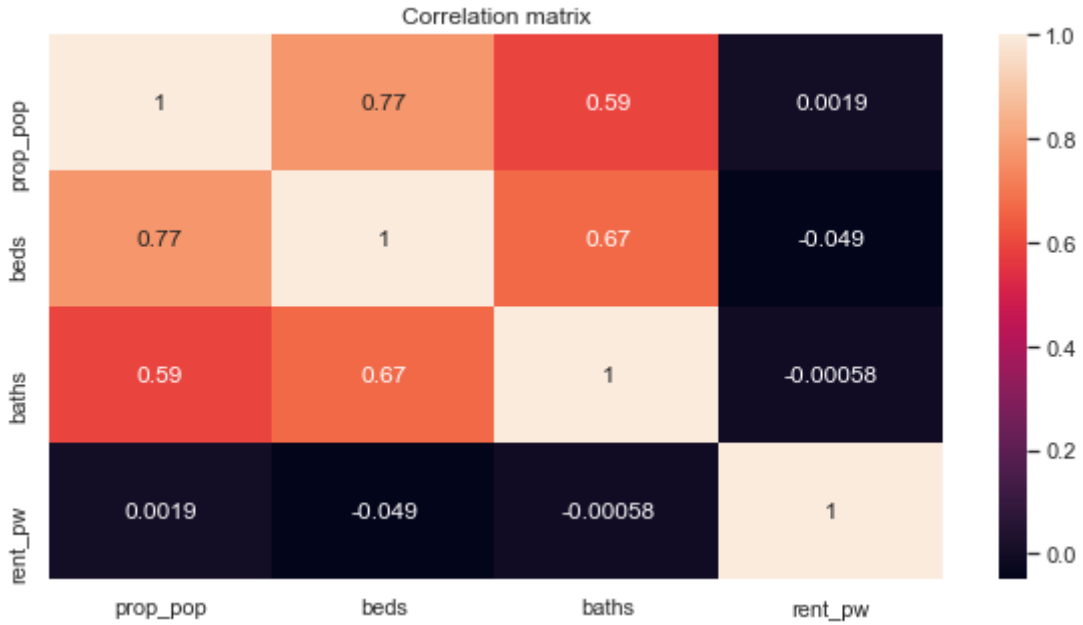
In [ ]:

```
# Plotting frequency histogram to check distribution of rent values
sns.set(rc={"figure.figsize":(10,5)})
plt.hist(df["rent_pw"],color="salmon",edgecolor="blue")

plt.title("Distribution of rent values")
plt.xlabel("Rent")
plt.ylabel("Frequency")
plt.show()
```

Heatmap to look for patterns

In [ ]:

```
# Creating a view with only numerical values
numerical_data = df[["prop_pop","beds","baths","rent_pw"]]

# creating a heatmap of correlation amongst values
sns.set(rc={"figure.figsize":(10,5)})
sns.heatmap(numerical_data.corr(),annot=True)

plt.title("Correlation matrix")
plt.show()
```



Negative correlation amongst rent, beds and bathrooms indicates that rent does not necessarily increase with respect to number of beds and baths.

Median rent by property type

In [ ]:

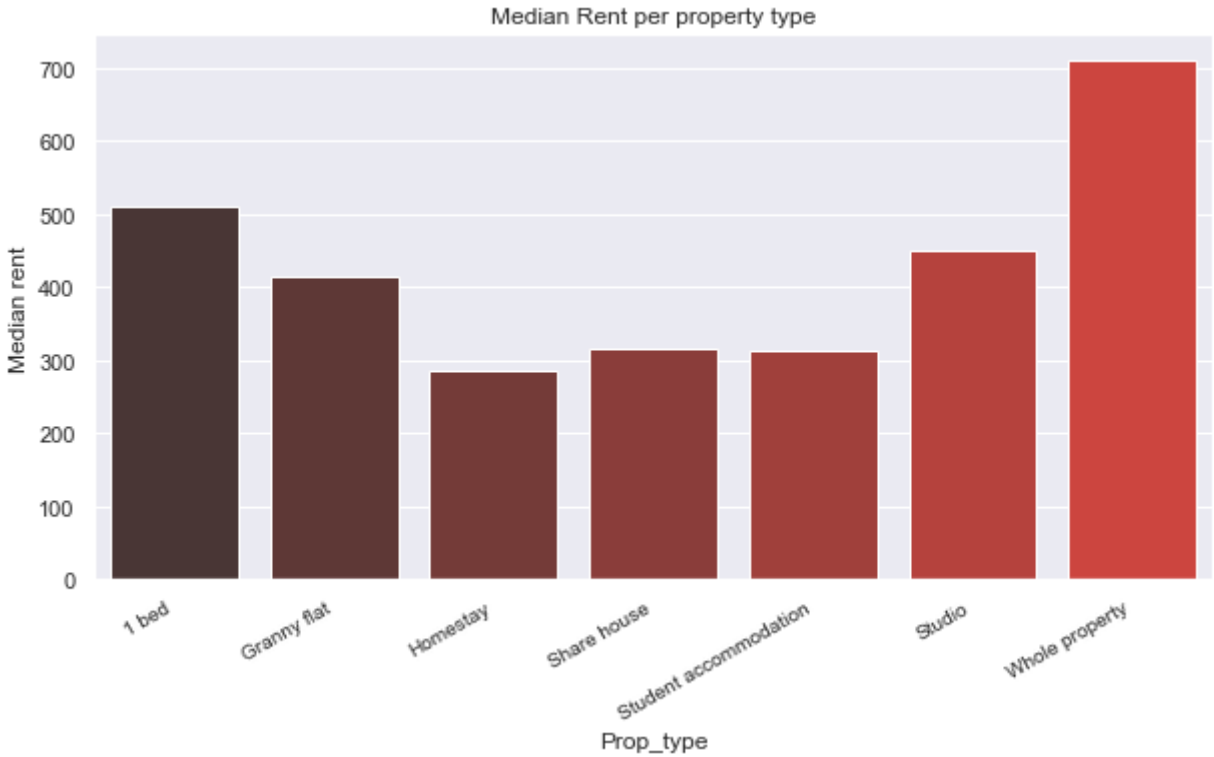
```
med_rent_per_prop = df.groupby(["prop_type"], as_index=False).median()
med_rent_per_prop = med_rent_per_prop[["prop_type","rent_pw"]]
med_rent_per_prop.columns = ["Prop_type","Median Rent"]
med_rent_per_prop
```

Out[ ]:

	Prop_type	Median Rent
0	1 bed	510.0
1	Granny flat	415.0
2	Homestay	285.0
3	Share house	315.0
4	Student accommodation	313.5
5	Studio	450.0
6	Whole property	710.0

In [ ]:

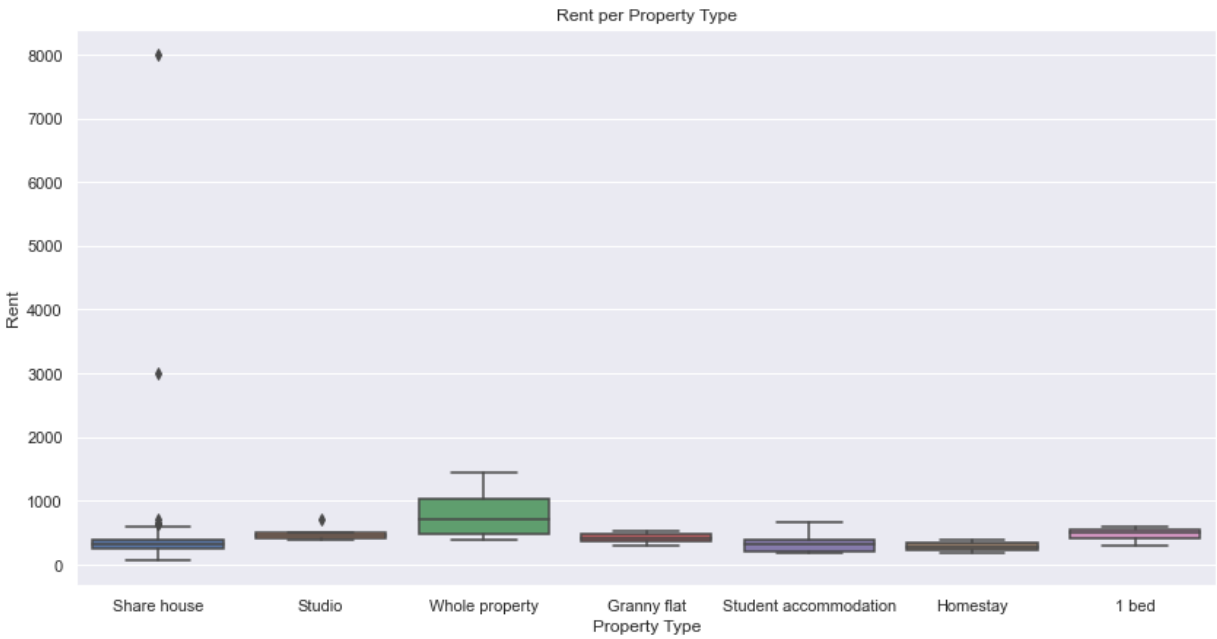
```
sns.set(rc={"figure.figsize":(10,5)})
palette = sns.color_palette("Reds_d", len(med_rent_per_prop)+6)
fig = sns.barplot(x=med_rent_per_prop['Prop_type'],y=med_rent_per_prop["Media
fig.set_xticklabels(fig.get_xticklabels(),fontsize=10, rotation =30, ha="righ
plt.title("Median Rent per property type")

plt.ylabel("Median rent")
plt.show()
```



Checking rent values

In [ ]:

```
# Checking for more outliers by creating boxplots
sns.set(rc={"figure.figsize":(14,7)})
sns.boxplot(df['prop_type'], df["rent_pw"])
plt.xlabel("Property Type")
plt.ylabel("Rent")
plt.title("Rent per Property Type")
plt.show()
```



Finding the minimum rent with bills included

In [ ]:

```
# Creating views where bills are included in rent
view = df.loc[(df["bills_inc"] == "Yes")]

# Getting rows with minimum rent where bills are included
```

```
result = view.loc[(view["rent_pw"] == view["rent_pw"].min())]

# Getting url for the property
url = url_creator(result['prop_ID'].values[0])
print(f"Click URL to view the property with the lowest rent where bills are i
```

```
Click URL to view the property with the lowest rent where bills are included :
        https://flatmates.com.au/share-house-sydney-penrith-2750-P1133933
```

Finding minimum rent where share houses have 1 bathroom per 2 people

Property population was increased by 1 to simulate the number of bathrooms available per person after one more person moves into the property.

In [ ]:

```
# Creating bath-to-pop ratio column (rounded to one decimal)
df["bath_to_pop_ratio"] = (df["baths"]/(df["prop_pop"]+1)).round(1)

# Getting share houses where there are atleast 1 bathroom per 2 bedrooms
view = df.loc[(df["bath_to_pop_ratio"] >= 0.5) & (df["prop_type"] == "Share h

# getting minimum rent where there are atleast 1 bathroom per 2 bedrooms
result = view.loc[(view["rent_pw"] == view["rent_pw"].min())]

url = url_creator(result['prop_ID'].values[0])
print(f"Click URL to view the property with the lowest rent where share house
```

```
Click URL to view the property with the lowest rent where share houses have 1
bathroom per 2 people :
        https://flatmates.com.au/share-house-sydney-penrith-2750-P1133933
```

Share houses with a population between 1 and 3

In [ ]:

```
# Creating views to get share houses with a population in range 1 - 3 (inclus
view = df.loc[(df["prop_pop"] >= 1) & (df["prop_pop"] <= 3) & (df["prop_type"
result = view.loc[(view["rent_pw"] == view["rent_pw"].min())]

url = url_creator(result['prop_ID'].values[0])
print(f"Click URL to view the property with the lowest rent where share house
```

```
Click URL to view the property with the lowest rent where share houses between
1 and 3 people:
        https://flatmates.com.au/share-house-sydney-penrith-2750-P1133933
```

Properties available in the next 10 days

In [ ]:

```
# Initializing empty column
df["days_to_avail"] = ""

# Converting datetime.date into datetime64 data type
today = pd.to_datetime(date.today())

# Filling column with difference of dates
df["days_to_avail"] = (df["avail_date"] - today)

# Convertin column to string
df["days_to_avail"] = df["days_to_avail"].astype("string")

# Splitting column strings to remove " days"
days = df["days_to_avail"].str.split(" ", n=1,expand=True)
```

```
# Replacing column values with numeric part
df["days_to_avail"] = days[0]

# Converting column to numeric
df["days_to_avail"] = pd.to_numeric(df["days_to_avail"])

# Initializing the number of days
days = 10

# Selecting properties that are available in the next 10 days
view = df.loc[(df["days_to_avail"] == days)]
view = view.sort_values(by=['rent_pw'])

# Printing count of available properties
print(f'{view["prop_ID"].count()} properties available in {days} days \n')

print("Click on URLs given to below to view properties available in the next

for id in range(view['prop_ID'].count()):
    url = url_creator(view["prop_ID"].values[id])
    print(f'\t {id+1}. {url} \n')

view
```

```
10 properties available in 10 days

Click on URLs given to below to view properties available in the next 10 days

        1. https://flatmates.com.au/student-accommodation-sydney-milperra-221
4-P1079356

        2. https://flatmates.com.au/student-accommodation-sydney-richmond-275
3-P1079342

        3. https://flatmates.com.au/student-accommodation-sydney-parramatta-2
150-P1077348

        4. https://flatmates.com.au/share-house-sydney-yagoona-2199-P1279604

        5. https://flatmates.com.au/student-accommodation-sydney-marsfield-21
22-P1077300

        6. https://flatmates.com.au/student-accommodation-sydney-frenchs-fore
st-2086-P1279753

        7. https://flatmates.com.au/student-accommodation-sydney-randwick-203
1-P1077243

        8. https://flatmates.com.au/share-house-sydney-lane-cove-north-2066-P
1237034

        9. https://flatmates.com.au/student-accommodation-sydney-newtown-2042
-P453555

        10. https://flatmates.com.au/share-house-sydney-2000-P1279696
```

Out[ ]:

	prop_ID	rent_pw	suburb	avail_date	bills_inc	prop_pop	beds	baths	p
227	student-accommodation-sydney-milperra-2214-P10...	187	Milperra	2023-01-01	Yes	5	5	2	accon

	prop_ID	rent_pw	suburb	avail_date	bills_inc	prop_pop	beds	baths	p
229	student-accommodation-sydney-richmond-2753-P10...	187	Richmond	2023-01-01	Yes	5	5	2	accon
224	student-accommodation-sydney-parramatta-2150-P...	245	Parramatta	2023-01-01	Yes	6	6	2	accon
155	share-house-sydney-yagoona-2199-P1279604	295	Yagoona	2023-01-01	Yes	2	3	3	Sh
230	student-accommodation-sydney-marsfield-2122-P1...	295	Marsfield	2023-01-01	Yes	5	5	4	accon
82	student-accommodation-sydney-frenchs-forest-20...	300	Frenchs Forest	2023-01-01	Yes	2	5	2	accon
225	student-accommodation-sydney-randwick-2031-P10...	327	Randwick	2023-01-01	Yes	6	6	2	accon
53	share-house-sydney-lane-cove-north-2066-P1237034	335	Lane Cove North	2023-01-01	Yes	3	3	2	Sh
223	student-accommodation-sydney-newtown-2042-P453555	362	Newtown	2023-01-01	Yes	5	5	2	accon
135	share-house-sydney-2000-P1279696	450	Sydney	2023-01-01	No	1	2	2	Sh

In [ ]: