In [ ]:
```
Task 2: Data Cleaning & Missing Value Handling

Dataset example: House Prices
Tool: Python (Pandas, NumPy)
```

In [1]:
```
#1.Import Required Libraries
```

In [3]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [9]:
```
#2. Load the Dataset
df = pd.read_csv("Housing.csv")
df.head()
```

Out[9]:

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheating	airconditioning	parking	prefarea
0	13300000	7420	4	2	3	yes	no	no	no	yes	2	yes
1	12250000	8960	4	4	4	yes	no	no	no	yes	3	no
2	12250000	9960	3	2	2	yes	no	yes	no	no	2	yes
3	12215000	7500	4	2	2	yes	no	yes	no	yes	3	yes
4	11410000	7420	4	1	2	yes	yes	yes	no	yes	2	no

In [15]:
```
#3. Identify Missing Values
df.isnull().sum()
```

Out[15]:
```
price               0
area                0
bedrooms            0
bathrooms           0
stories             0
mainroad            0
guestroom           0
basement            0
hotwaterheating     0
airconditioning     0
parking             0
prefarea            0
furnishingstatus    0
dtype: int64
```

In [21]:
```
#4.Visualize Missing Data
missing = df.isnull().sum()
missing = missing[missing > 0]

if missing.empty:
    print("No missing values found in the dataset.")
else:
    plt.figure(figsize=(10,5))
    missing.plot(kind='bar')
    plt.title("Missing Values per Column")
    plt.ylabel("Count")
    plt.xlabel("Columns")
    plt.show()
```

```
No missing values found in the dataset.
```

In [23]:
```
#5. Separate Numerical & Categorical Columns
num_cols = df.select_dtypes(include=[np.number]).columns
cat_cols = df.select_dtypes(include=['object']).columns
```

In [27]:
```
#For numerical columns
for col in num_cols:
    df[col] = df[col].fillna(df[col].median())

#For categorical columns
```

```
for col in cat_cols:
    df[col] = df[col].fillna(df[col].mode()[0])
```

In [ ]:
```
Observation:
Missing values were handled by assigning imputed values directly to the dataframe columns to avoid chained assignment issues a
```

In [31]:
```
#8. Drop Columns with Extremely High Missing Values
threshold = 0.6
df = df.loc[:, df.isnull().mean() < threshold]
```

In [ ]:
```
Observation:
Columns with more than 60% missing values are removed, as imputing such columns can introduce noise and reduce data quality.
```

In [33]:
```
#9.Validate Dataset After Cleaning
df.isnull().sum()

df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 545 entries, 0 to 544
Data columns (total 13 columns):
 #   Column            Non-Null Count  Dtype 
---  ------            --------------  ----- 
 0   price             545 non-null    int64 
 1   area              545 non-null    int64 
 2   bedrooms          545 non-null    int64 
 3   bathrooms         545 non-null    int64 
 4   stories           545 non-null    int64 
 5   mainroad          545 non-null    object
 6   guestroom         545 non-null    object
 7   basement          545 non-null    object
 8   hotwaterheating   545 non-null    object
 9   airconditioning   545 non-null    object
 10  parking           545 non-null    int64 
 11  prefarea          545 non-null    object
 12  furnishingstatus  545 non-null    object
dtypes: int64(6), object(7)
memory usage: 55.5+ KB
```

In [35]:
```
#10. Compare Dataset Before vs After Cleaning
print("Dataset shape after cleaning:", df.shape)
```

```
Dataset shape after cleaning: (545, 13)
```

In [37]:
```
#11. Save Cleaned Dataset
df.to_csv("cleaned_dataset.csv", index=False)
```

In [ ]:
```
Complete description:
  🔹 Step 1: Dataset Loading

The dataset was loaded into a Pandas DataFrame to examine its structure,
including rows, columns, and data types.


 🔹 Step 2: Identifying Missing Values

Missing values were identified using built-in Pandas functions to determine which columns
contained null values and how frequently they occurred.


 🔹 Step 3: Visualizing Missing Data

The distribution of missing values across columns was visualized using a bar chart.


If no missing values were present, visualization was skipped as the dataset was already complete.

 🔹 Step 4: Separating Numerical and Categorical Features

The dataset was divided into numerical and categorical columns.


 🔹 Step 5: Handling Missing Values in Numerical Columns

Missing values in numerical columns were handled using median imputation.
Median was preferred over mean to reduce the impact of outliers and skewed data distributions.

 🔹 Step 6: Handling Missing Values in Categorical Columns
```

```
Missing values in categorical columns were filled using mode imputation,
replacing missing entries with the most frequently occurring category to preserve data consistency.

 🔹 Step 7: Removing Columns with High Missing Values

Columns with extremely high proportions of missing data were removed from the dataset.
Imputing such columns could introduce noise and negatively affect data quality.

 🔹 Step 8: Dataset Validation After Cleaning

After cleaning, the dataset was validated to ensure all missing values were handled appropriately.
Data types and dataset structure were reviewed to confirm readiness for further analysis or modeling.

 🔹 Step 9: Dataset Comparison (Before vs After Cleaning)

The dataset dimensions before and after cleaning were compared
to understand the impact of preprocessing on data size and overall quality.

 🔹 Step 10: Saving the Cleaned Dataset

The cleaned dataset was saved for further analysis and modeling.
This ensures reproducibility and allows the cleaned data
to be reused in future machine learning tasks.

 Final Outcome

This task provided hands-on experience in data preprocessing,
including missing value identification,
visualization, imputation, and validation.
The cleaned dataset is now reliable and suitable for downstream analytics and machine learning workflows.
```