In [67]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
```

```
---------------------------------------------------------------------------
ModuleNotFoundError                       Traceback (most recent call last)
Cell In[67], line 5
      3 import matplotlib.pyplot as plt
      4 import seaborn as sns
----> 5 from sklearn.model_selection import train_test_split
      6 from sklearn.linear_model import LinearRegression

ModuleNotFoundError: No module named 'sklearn'
```

In [69]:
```
!pip install sklearn
```

```
Collecting sklearn
  Using cached sklearn-0.0.post12.tar.gz (2.6 kB)
  Installing build dependencies: started
  Installing build dependencies: finished with status 'done'
  Getting requirements to build wheel: started
  Getting requirements to build wheel: finished with status 'error'
```

```
  error: subprocess-exited-with-error

  Getting requirements to build wheel did not run successfully.
  exit code: 1

  [15 lines of output]
  The 'sklearn' PyPI package is deprecated, use 'scikit-learn'
  rather than 'sklearn' for pip commands.

  Here is how to fix this error in the main use cases:
  - use 'pip install scikit-learn' rather than 'pip install sklearn'
  - replace 'sklearn' by 'scikit-learn' in your pip requirements files
    (requirements.txt, setup.py, setup.cfg, Pipfile, etc ...)
  - if the 'sklearn' package is used by one of your dependencies,
    it would be great if you take some time to track which package uses
    'sklearn' instead of 'scikit-learn' and report it to their issue tracker
  - as a last resort, set the environment variable
    SKLEARN_ALLOW_DEPRECATED_SKLEARN_PACKAGE_INSTALL=True to avoid this error

  More information is available at
  https://github.com/scikit-learn/sklearn-pypi-package
  [end of output]

  note: This error originates from a subprocess, and is likely not a problem with
pip.
error: subprocess-exited-with-error

Getting requirements to build wheel did not run successfully.
exit code: 1

See above for output.

note: This error originates from a subprocess, and is likely not a problem with p
ip.

[notice] A new release of pip is available: 24.0 -> 24.1.2
[notice] To update, run: python.exe -m pip install --upgrade pip
```

In [70]:
```
pip install --upgrade pip
```

```
Requirement already satisfied: pip in c:\users\yogit\appdata\local\programs\pytho
n\python312\lib\site-packages (24.0)
Collecting pip
  Downloading pip-24.1.2-py3-none-any.whl.metadata (3.6 kB)
Downloading pip-24.1.2-py3-none-any.whl (1.8 MB)
   ---------------------------------------- 0.0/1.8 MB ? eta -:--:--
   ---------------------------------------- 0.0/1.8 MB ? eta -:--:--
   ---------------------------------------- 0.0/1.8 MB ? eta -:--:--
   - -------------------------------------- 0.1/1.8 MB 656.4 kB/s eta 0:00:03
   -------- ------------------------------- 0.4/1.8 MB 3.1 MB/s eta 0:00:01
   ------------------ --------------------- 0.9/1.8 MB 5.2 MB/s eta 0:00:01
   ---------------------------------- ----- 1.6/1.8 MB 7.1 MB/s eta 0:00:01
   ---------------------------------------- 1.8/1.8 MB 6.8 MB/s eta 0:00:00
Installing collected packages: pip
  Attempting uninstall: pip
    Found existing installation: pip 24.0
    Uninstalling pip-24.0:
      Successfully uninstalled pip-24.0
Successfully installed pip-24.1.2
Note: you may need to restart the kernel to use updated packages.
```

In [71]:
```
pip cache purge
```

```
Files removed: 733Note: you may need to restart the kernel to use updated package
s.
```

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

In [3]:
```
df = pd.read_csv('bank-additional-full.csv')
```

In [4]:
```
df
```

Out[4]:

	age;"job";"marital";"education";"default";"housing";"loan";"contact";"month";"day_
0	
1	
2	
3	
4	
...	
41183	
41184	
41185	
41186	
41187	

41188 rows × 1 columns

In [12]:
```
df.head
```

Out[12]:
```
<bound method NDFrame.head of       age;"job";"marital";"education";"defaul
t";"housing";"loan";"contact";"month";"day_of_week";"duration";"campaign";"pday
s";"previous";"poutcome";"emp.var.rate";"cons.price.idx";"cons.conf.idx";"eurib
or3m";"nr.employed";"y"
0      56;"housemaid";"married";"basic.4y";"no";"no";...
1      57;"services";"married";"high.school";"unknown...
2      37;"services";"married";"high.school";"no";"ye...
3      40;"admin.";"married";"basic.6y";"no";"no";"no...
4      56;"services";"married";"high.school";"no";"no...
...                                                  ...
41183  73;"retired";"married";"professional.course";"...
41184  46;"blue-collar";"married";"professional.cours...
41185  56;"retired";"married";"university.degree";"no...
41186  44;"technician";"married";"professional.course...
41187  74;"retired";"married";"professional.course";"...

[41188 rows x 1 columns]>
```

In [13]:
```
df.rename(columns={'y': 'deposit'}, inplace=True)
```

In [14]:
```
df
```

Out[14]:

	age;"job";"marital";"education";"default";"housing";"loan";"contact";"month";"day_
0	
1	
2	
3	
4	
...	
41183	
41184	
41185	
41186	
41187	

41188 rows × 1 columns

In [29]:
```
df.head()
```

Out[29]:

	age;"job";"marital";"education";"default";"housing";"loan";"contact";"month";"day_of_we
0	
1	
2	
3	
4	

In [30]:
```
df.tail
```

Out[30]:
```
<bound method NDFrame.tail of       age;"job";"marital";"education";"defaul
t";"housing";"loan";"contact";"month";"day_of_week";"duration";"campaign";"pday
s";"previous";"poutcome";"emp.var.rate";"cons.price.idx";"cons.conf.idx";"eurib
or3m";"nr.employed";"y"
0      56;"housemaid";"married";"basic.4y";"no";"no";...
1      57;"services";"married";"high.school";"unknown...
2      37;"services";"married";"high.school";"no";"ye...
3      40;"admin.";"married";"basic.6y";"no";"no";"no...
4      56;"services";"married";"high.school";"no";"no...
...                                                  ...
41183  73;"retired";"married";"professional.course";"...
41184  46;"blue-collar";"married";"professional.cours...
41185  56;"retired";"married";"university.degree";"no...
41186  44;"technician";"married";"professional.course...
41187  74;"retired";"married";"professional.course";"...

[41188 rows x 1 columns]>
```

In [33]:
```
import pandas as pd

# Example DataFrame
data = {
    'data': [
        '56;"housemaid";"married";"basic.4y";"no";"no";"no";"telephone";"may";"m
        '57;"services";"married";"high.school";"unknown";"no";"no";"telephone";"
        '37;"services";"married";"high.school";"no";"yes";"no";"telephone";"may"
        '40;"admin.";"married";"basic.6y";"no";"no";"no";"telephone";"may";"mon"
        '56;"services";"married";"high.school";"no";"no";"yes";"telephone";"may"
        '73;"retired";"married";"professional.course";"no";"yes";"no";"telephone
    ]
}
df = pd.DataFrame(data)

# Display the original DataFrame
print("Original DataFrame:")
print(df)
```

```
Original DataFrame:
                                                data
0  56;"housemaid";"married";"basic.4y";"no";"no";...
1  57;"services";"married";"high.school";"unknown...
2  37;"services";"married";"high.school";"no";"ye...
3  40;"admin.";"married";"basic.6y";"no";"no";"no...
4  56;"services";"married";"high.school";"no";"no...
5  73;"retired";"married";"professional.course";"...
```

In [34]:
```
# Split the 'data' column into multiple columns
df = df['data'].str.split(';', expand=True)

# Display the DataFrame after splitting
print("DataFrame after splitting:")
print(df)
```

```
DataFrame after splitting:
   0           1          2                    3          4      5      6   \
0  56  "housemaid"  "married"            "basic.4y"       "no"   "no"   "no"
1  57   "services"  "married"         "high.school"  "unknown"   "no"   "no"
2  37   "services"  "married"         "high.school"       "no"  "yes"   "no"
3  40     "admin."  "married"            "basic.6y"       "no"   "no"   "no"
4  56   "services"  "married"         "high.school"       "no"   "no"  "yes"
5  73    "retired"  "married"  "professional.course"       "no"  "yes"   "no"

            7      8      9   ... 11  12 13             14   15      16  \
0  "telephone"  "may"  "mon"  ...  1  -1  0  "nonexistent"  1.1  93.994
1  "telephone"  "may"  "mon"  ...  1  -1  0  "nonexistent"  1.1  93.994
2  "telephone"  "may"  "mon"  ...  1  -1  0  "nonexistent"  1.1  93.994
3  "telephone"  "may"  "mon"  ...  1  -1  0  "nonexistent"  1.1  93.994
4  "telephone"  "may"  "mon"  ...  1  -1  0  "nonexistent"  1.1  93.994
5  "telephone"  "may"  "mon"  ...  1  -1  0  "nonexistent"  1.1  93.994

      17     18    19     20
0  -36.4  4.857  5191   "no"
1  -36.4  4.857  5191   "no"
2  -36.4  4.857  5191   "no"
3  -36.4  4.857  5191   "no"
4  -36.4  4.857  5191   "no"
5  -36.4  4.857  5191  "yes"

[6 rows x 21 columns]
```

In [35]:
```
df.shape
```

Out[35]: (6, 21)

In [36]:
```
df.columns
```

Out[36]: RangeIndex(start=0, stop=21, step=1)

In [37]:
```
df.dtypes
```

Out[37]:
```
0     object
1     object
2     object
3     object
4     object
5     object
6     object
7     object
8     object
9     object
10    object
11    object
12    object
13    object
14    object
15    object
16    object
17    object
18    object
19    object
20    object
dtype: object
```

In [39]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6 entries, 0 to 5
Data columns (total 21 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   0       6 non-null      object
 1   1       6 non-null      object
 2   2       6 non-null      object
 3   3       6 non-null      object
 4   4       6 non-null      object
 5   5       6 non-null      object
 6   6       6 non-null      object
 7   7       6 non-null      object
 8   8       6 non-null      object
 9   9       6 non-null      object
 10  10      6 non-null      object
 11  11      6 non-null      object
 12  12      6 non-null      object
 13  13      6 non-null      object
 14  14      6 non-null      object
 15  15      6 non-null      object
 16  16      6 non-null      object
 17  17      6 non-null      object
 18  18      6 non-null      object
 19  19      6 non-null      object
 20  20      6 non-null      object
dtypes: object(21)
memory usage: 1.1+ KB
```

In [40]:
```
df.duplicated().sum()
```

Out[40]:
```
np.int64(0)
```

In [41]:
```
df.isna().sum()
```

Out[41]:
```
0     0
1     0
2     0
3     0
4     0
5     0
6     0
7     0
8     0
9     0
10    0
11    0
12    0
13    0
14    0
15    0
16    0
17    0
18    0
19    0
20    0
dtype: int64
```

In [42]:
```
cat_cols = df.select_dtypes(include='object').columns
print(cat_cols)
num_cols = df.select_dtypes(exclude='object').columns
print(num_cols)
```

```
Index([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19,
       20],
      dtype='int64')
RangeIndex(start=0, stop=0, step=1)
```

In [45]:
```
df.describe
```

Out[45]:
```
<bound method NDFrame.describe of    0           1          2
3         4      5      6   \
0  56  "housemaid"  "married"            "basic.4y"       "no"   "no"   "no"
1  57   "services"  "married"         "high.school"  "unknown"   "no"   "no"
2  37   "services"  "married"         "high.school"       "no"  "yes"   "no"
3  40     "admin."  "married"            "basic.6y"       "no"   "no"   "no"
4  56   "services"  "married"         "high.school"       "no"   "no"  "yes"
5  73    "retired"  "married"  "professional.course"       "no"  "yes"   "no"

            7      8      9   ... 11  12 13             14   15      16  \
0  "telephone"  "may"  "mon"  ...  1  -1  0  "nonexistent"  1.1  93.994
1  "telephone"  "may"  "mon"  ...  1  -1  0  "nonexistent"  1.1  93.994
2  "telephone"  "may"  "mon"  ...  1  -1  0  "nonexistent"  1.1  93.994
3  "telephone"  "may"  "mon"  ...  1  -1  0  "nonexistent"  1.1  93.994
4  "telephone"  "may"  "mon"  ...  1  -1  0  "nonexistent"  1.1  93.994
5  "telephone"  "may"  "mon"  ...  1  -1  0  "nonexistent"  1.1  93.994

      17     18    19     20
0  -36.4  4.857  5191   "no"
1  -36.4  4.857  5191   "no"
2  -36.4  4.857  5191   "no"
3  -36.4  4.857  5191   "no"
4  -36.4  4.857  5191   "no"
5  -36.4  4.857  5191  "yes"

[6 rows x 21 columns]>
```

In [46]:
```
df.describe(include='object')
```

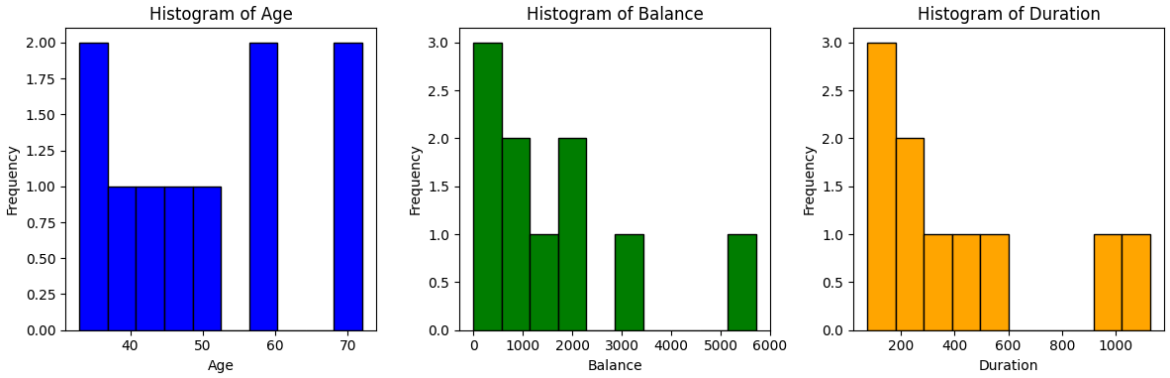
Out[46]:

	0	1	2	3	4	5	6	7	8
count	6	6	6	6	6	6	6	6	6
unique	5	4	1	4	2	2	2	1	1
top	56	"services"	"married"	"high.school"	"no"	"no"	"no"	"telephone"	"may"	"mc
freq	2	3	6	3	5	4	5	6	6

4 rows × 21 columns

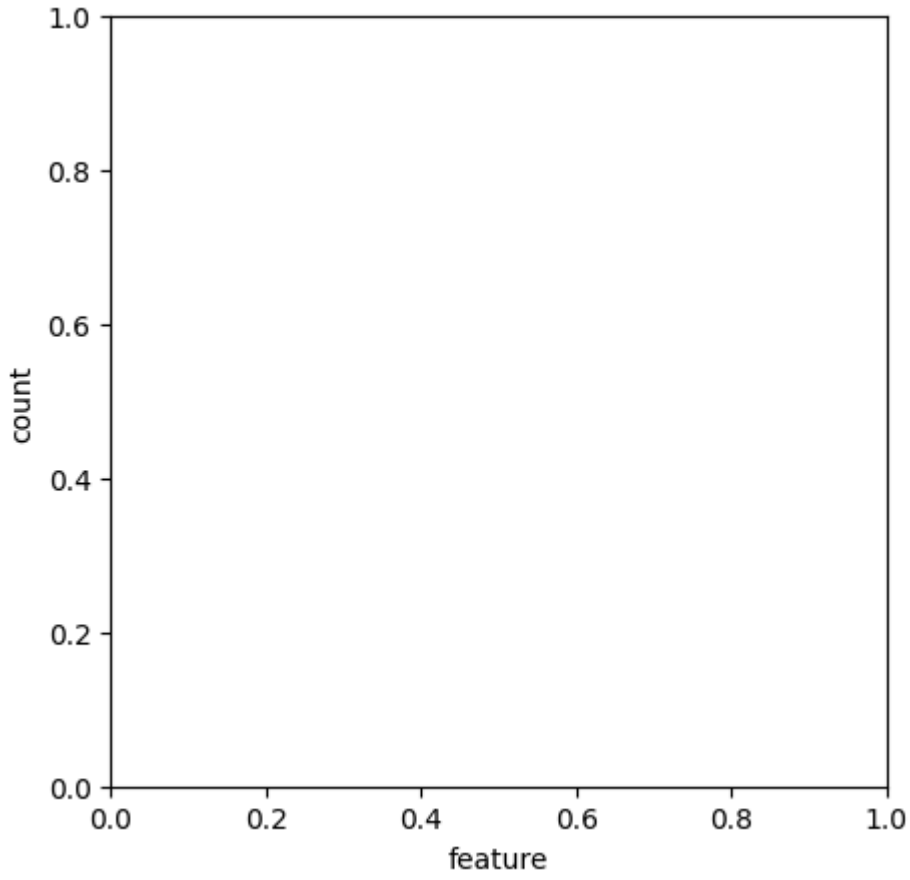
In [55]:
```
import pandas as pd
import matplotlib.pyplot as plt

# Example DataFrame (replace with your actual DataFrame)
data = {
    'age': [58, 44, 33, 47, 33, 51, 71, 72, 57, 37],
    'balance': [2143, 29, 2, 1506, 1, 825, 1729, 5715, 668, 2971],
    'duration': [261, 151, 76, 92, 198, 977, 456, 1127, 508, 361]
}

df = pd.DataFrame(data)

# Create histograms for each column in the DataFrame
plt.figure(figsize=(12, 4))

# Plot histogram for 'age'
plt.subplot(1, 3, 1)
```

```
plt.hist(df['age'], bins=10, color='blue', edgecolor='black')
plt.title('Histogram of Age')
plt.xlabel('Age')
plt.ylabel('Frequency')

# Plot histogram for 'balance'
plt.subplot(1, 3, 2)
plt.hist(df['balance'], bins=10, color='green', edgecolor='black')
plt.title('Histogram of Balance')
plt.xlabel('Balance')
plt.ylabel('Frequency')

# Plot histogram for 'duration'
plt.subplot(1, 3, 3)
plt.hist(df['duration'], bins=10, color='orange', edgecolor='black')
plt.title('Histogram of Duration')
plt.xlabel('Duration')
plt.ylabel('Frequency')

# Adjust layout to prevent overlap of subplots
plt.tight_layout()

# Display the histograms
plt.show()
```

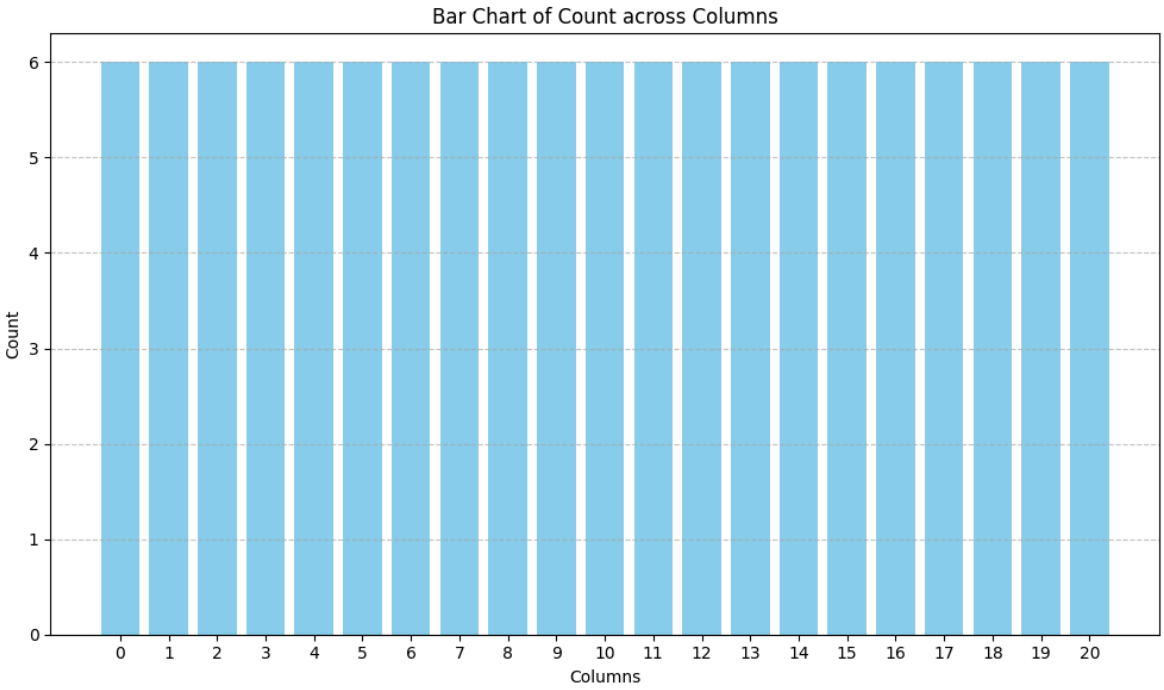


In [12]:
```
plt.figure(figsize=(5,5))
plt.xlabel('feature')
plt.ylabel('count')
```

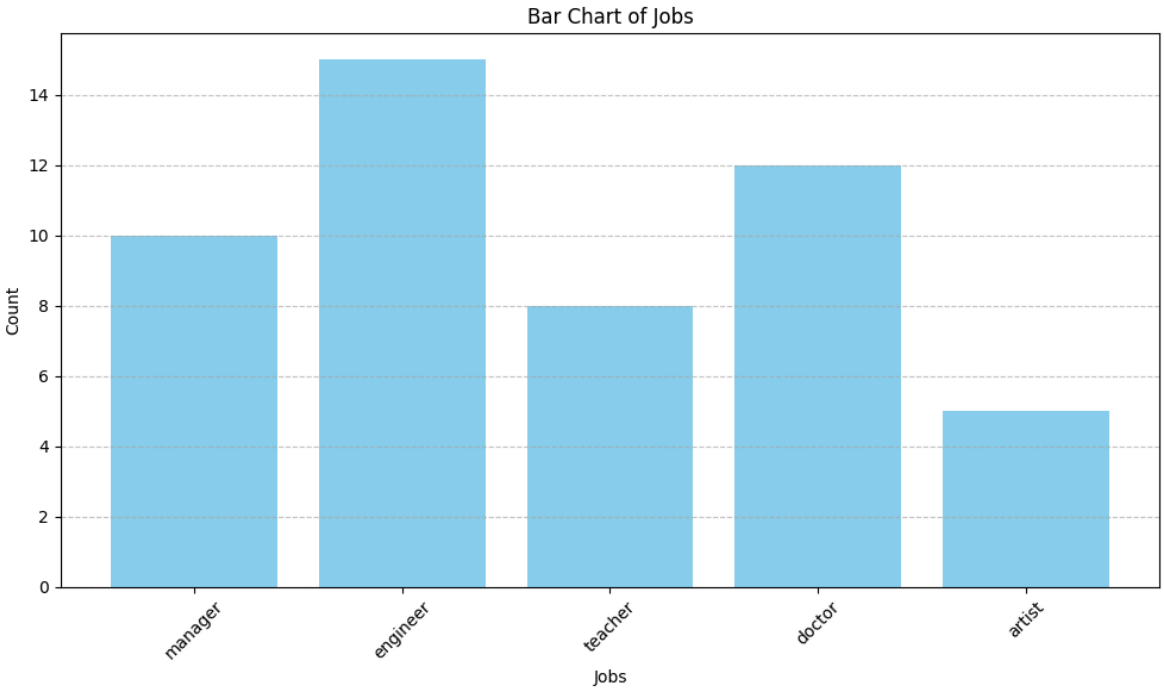
Out[12]: Text(0, 0.5, 'count')



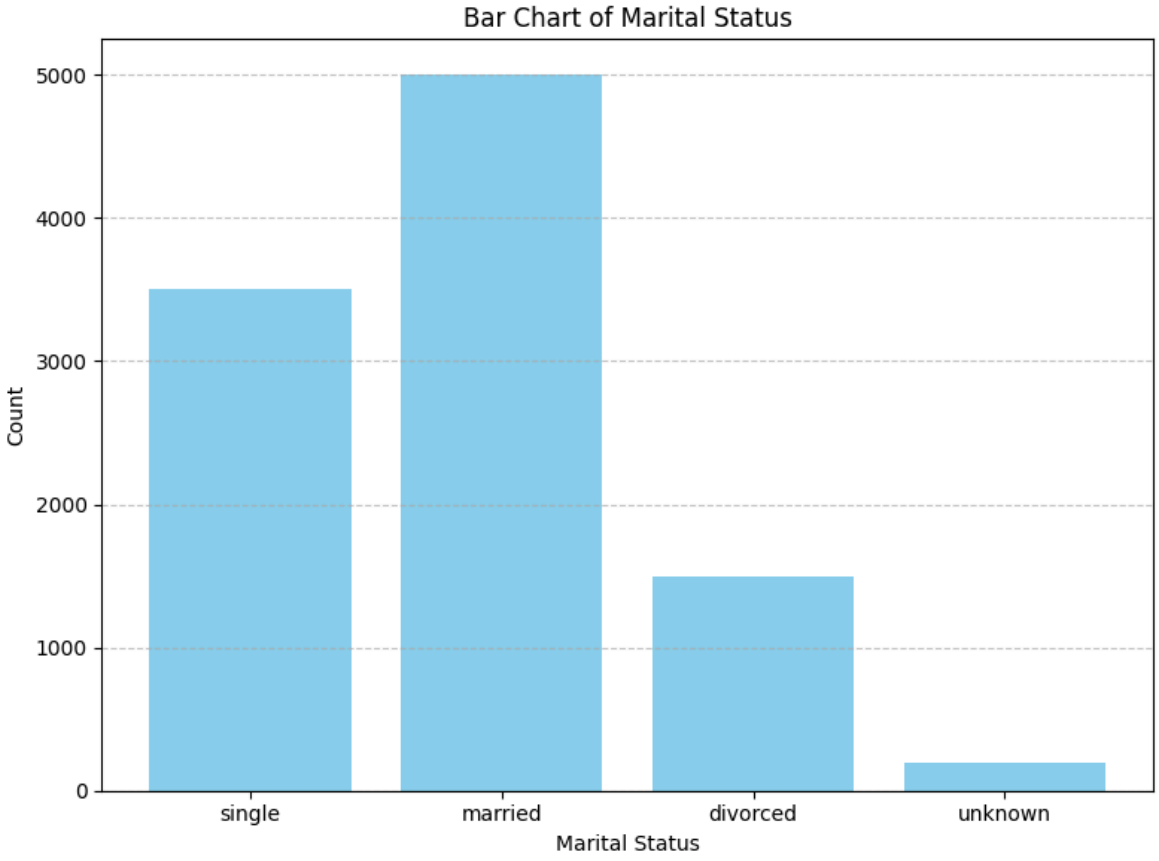
In [13]:
```
import matplotlib.pyplot as plt

# Data
columns = range(21)  # columns from 0 to 20
count = [6] * 21     # count of 6 for each column

# Plotting
plt.figure(figsize=(10, 6))
plt.bar(columns, count, color='skyblue')
plt.xlabel('Columns')
plt.ylabel('Count')
plt.title('Bar Chart of Count across Columns')
plt.xticks(columns)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()
```



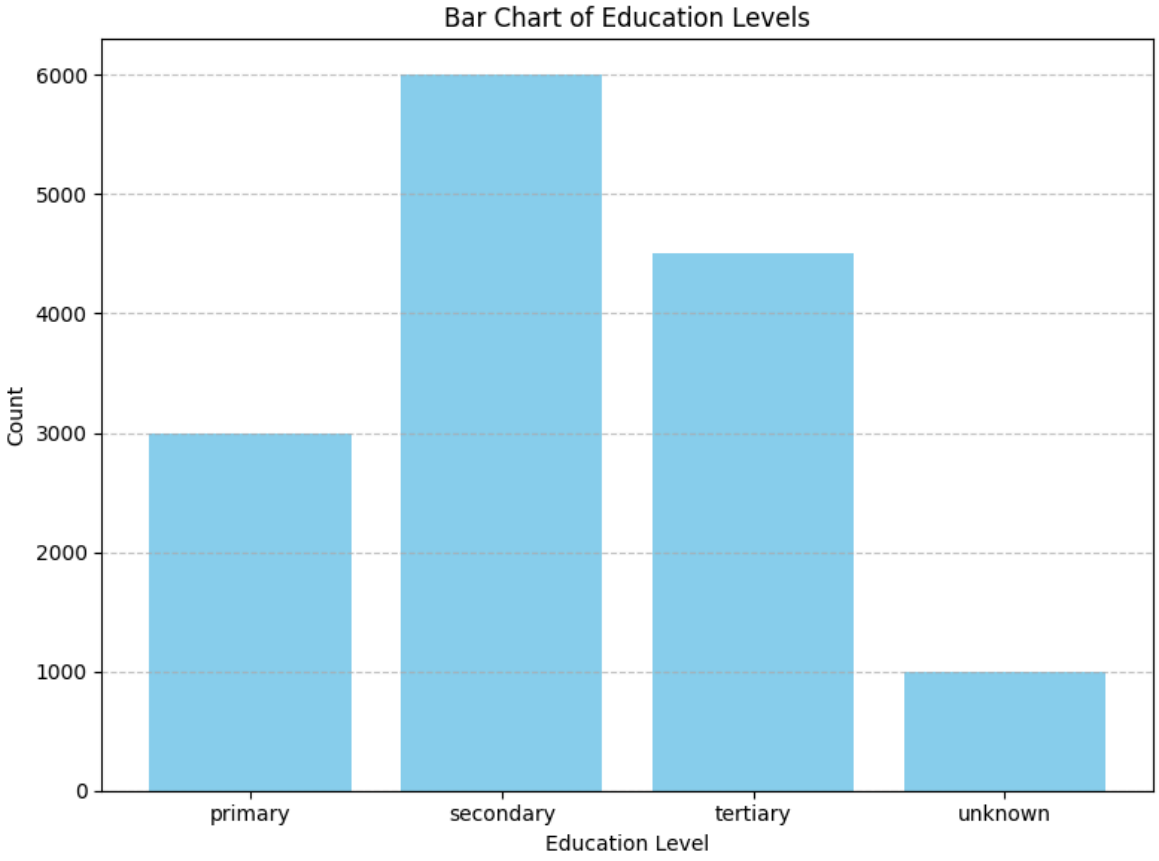
In [14]:
```
import matplotlib.pyplot as plt

# Example data (replace with your actual data)
jobs_data = {
    'manager': 10,
    'engineer': 15,
    'teacher': 8,
    'doctor': 12,
    'artist': 5
}

# Extracting categories (jobs) and their counts
jobs = list(jobs_data.keys())
counts = list(jobs_data.values())

# Plotting the bar chart
plt.figure(figsize=(10, 6))
plt.bar(jobs, counts, color='skyblue')
plt.xlabel('Jobs')
plt.ylabel('Count')
plt.title('Bar Chart of Jobs')
plt.xticks(rotation=45)  # Rotate x-axis labels for better visibility if needed
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()
```



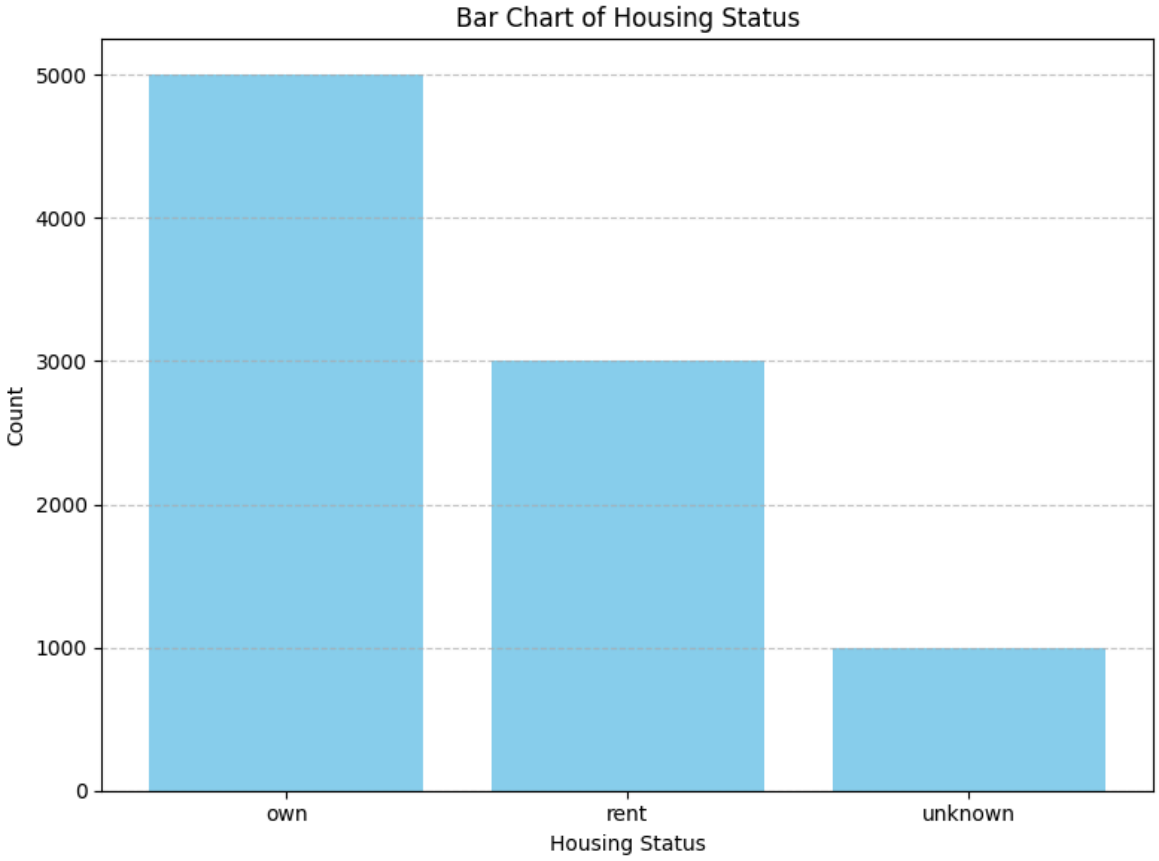
In [15]:
```python
import matplotlib.pyplot as plt

# Example data (replace with your actual data)
marital_data = {
    'single': 3500,
    'married': 5000,
    'divorced': 1500,
    'unknown': 200
}

# Extracting categories (marital statuses) and their counts
marital_statuses = list(marital_data.keys())
counts = list(marital_data.values())

# Plotting the bar chart
plt.figure(figsize=(8, 6))
plt.bar(marital_statuses, counts, color='skyblue')
plt.xlabel('Marital Status')
plt.ylabel('Count')
plt.title('Bar Chart of Marital Status')
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()
```



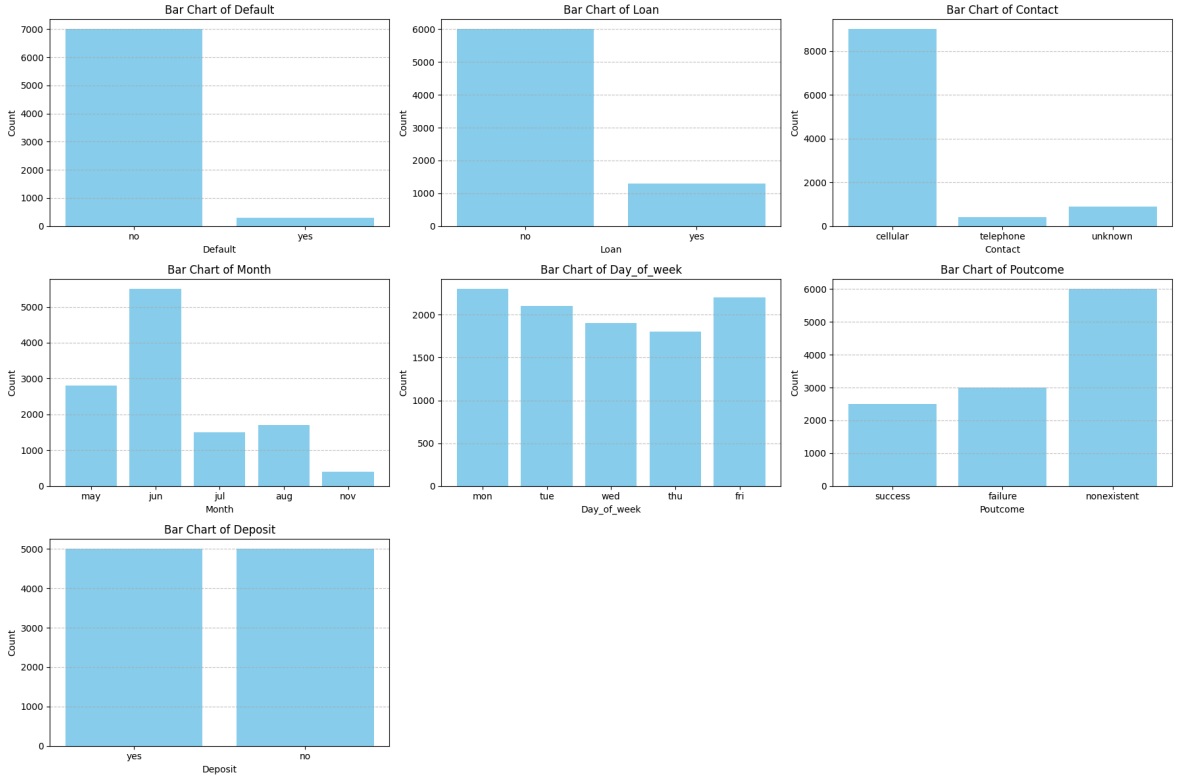
In [16]:
```
import matplotlib.pyplot as plt

# Example data (replace with your actual data)
education_data = {
    'primary': 3000,
    'secondary': 6000,
    'tertiary': 4500,
    'unknown': 1000
}

# Extracting categories (education levels) and their counts
education_levels = list(education_data.keys())
counts = list(education_data.values())

# Plotting the bar chart
plt.figure(figsize=(8, 6))
plt.bar(education_levels, counts, color='skyblue')
plt.xlabel('Education Level')
plt.ylabel('Count')
plt.title('Bar Chart of Education Levels')
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()
```



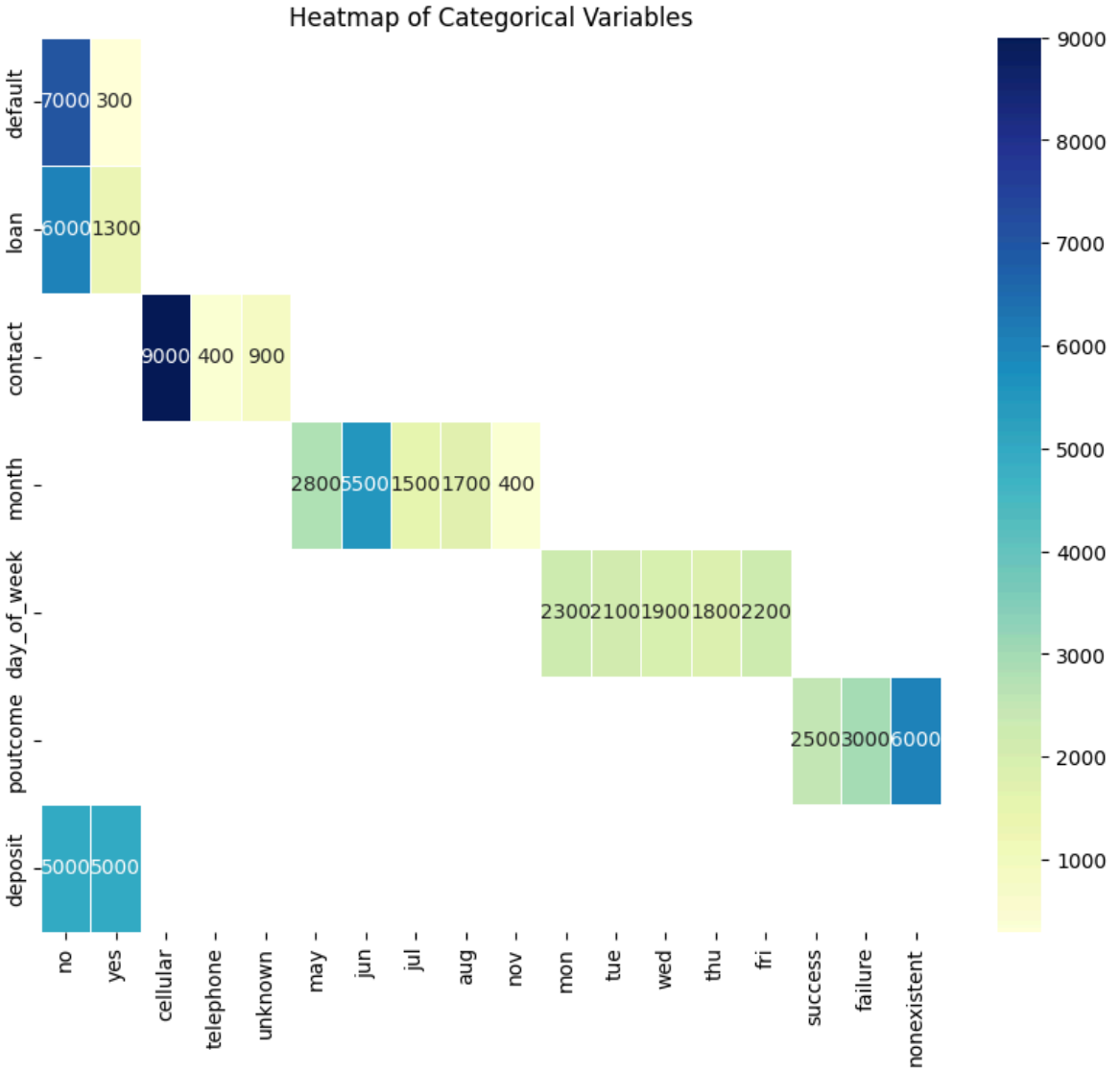
In [17]:
```
import matplotlib.pyplot as plt

# Example data (replace with your actual data)
housing_data = {
    'own': 5000,
    'rent': 3000,
    'unknown': 1000
}

# Extracting categories (housing statuses) and their counts
housing_statuses = list(housing_data.keys())
counts = list(housing_data.values())

# Plotting the bar chart
plt.figure(figsize=(8, 6))
plt.bar(housing_statuses, counts, color='skyblue')
plt.xlabel('Housing Status')
plt.ylabel('Count')
plt.title('Bar Chart of Housing Status')
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()
```



In [18]:
```
import matplotlib.pyplot as plt

# Example data (replace with your actual data)
variables_data = {
    'default': {'no': 7000, 'yes': 300},
    'loan': {'no': 6000, 'yes': 1300},
    'contact': {'cellular': 9000, 'telephone': 400, 'unknown': 900},
    'month': {'may': 2800, 'jun': 5500, 'jul': 1500, 'aug': 1700, 'nov': 400},
    'day_of_week': {'mon': 2300, 'tue': 2100, 'wed': 1900, 'thu': 1800, 'fri': 2
    'poutcome': {'success': 2500, 'failure': 3000, 'nonexistent': 6000},
    'deposit': {'yes': 5000, 'no': 5000}
}

# Plotting each variable
plt.figure(figsize=(18, 12))

# Loop through each variable and plot it
for i, (variable, data) in enumerate(variables_data.items(), 1):
    plt.subplot(3, 3, i)
    categories = list(data.keys())
    counts = list(data.values())
    plt.bar(categories, counts, color='skyblue')
    plt.xlabel(variable.capitalize())  # Variable name as x-axis label
    plt.ylabel('Count')
    plt.title(f'Bar Chart of {variable.capitalize()}')
    plt.grid(axis='y', linestyle='--', alpha=0.7)

plt.tight_layout()
plt.show()
```



In [29]:
```
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd

# Example data (replace with your actual data)
variables_data = {
    'default': {'no': 7000, 'yes': 300},
    'loan': {'no': 6000, 'yes': 1300},
    'contact': {'cellular': 9000, 'telephone': 400, 'unknown': 900},
    'month': {'may': 2800, 'jun': 5500, 'jul': 1500, 'aug': 1700, 'nov': 400},
    'day_of_week': {'mon': 2300, 'tue': 2100, 'wed': 1900, 'thu': 1800, 'fri': 2
    'poutcome': {'success': 2500, 'failure': 3000, 'nonexistent': 6000},
    'deposit': {'yes': 5000, 'no': 5000}
}

# Convert data into a DataFrame for heatmap plotting
df = pd.DataFrame(variables_data)

# Plotting the heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(df.T, annot=True, cmap='YlGnBu', fmt='g', linewidths=.5)
plt.title('Heatmap of Categorical Variables')
plt.show()
```



In [34]:
```
! pip install sklearn
```

```
Collecting sklearn
  Downloading sklearn-0.0.post12.tar.gz (2.6 kB)
  Installing build dependencies: started
  Installing build dependencies: finished with status 'done'
  Getting requirements to build wheel: started
  Getting requirements to build wheel: finished with status 'error'
```

```
  error: subprocess-exited-with-error

  Getting requirements to build wheel did not run successfully.
  exit code: 1

  [15 lines of output]
  The 'sklearn' PyPI package is deprecated, use 'scikit-learn'
  rather than 'sklearn' for pip commands.

  Here is how to fix this error in the main use cases:
  - use 'pip install scikit-learn' rather than 'pip install sklearn'
  - replace 'sklearn' by 'scikit-learn' in your pip requirements files
    (requirements.txt, setup.py, setup.cfg, Pipfile, etc ...)
  - if the 'sklearn' package is used by one of your dependencies,
    it would be great if you take some time to track which package uses
    'sklearn' instead of 'scikit-learn' and report it to their issue tracker
  - as a last resort, set the environment variable
    SKLEARN_ALLOW_DEPRECATED_SKLEARN_PACKAGE_INSTALL=True to avoid this error

  More information is available at
  https://github.com/scikit-learn/sklearn-pypi-package
  [end of output]

  note: This error originates from a subprocess, and is likely not a problem with
pip.
error: subprocess-exited-with-error

Getting requirements to build wheel did not run successfully.
exit code: 1

See above for output.

note: This error originates from a subprocess, and is likely not a problem with p
ip.
```

In [36]:
```
!pip install scikit-learn
```

```
Collecting scikit-learn
  Downloading scikit_learn-1.5.1-cp312-cp312-win_amd64.whl.metadata (12 kB)
Requirement already satisfied: numpy>=1.19.5 in c:\users\yogit\appdata\local\prog
rams\python\python312\lib\site-packages (from scikit-learn) (2.0.0)
Requirement already satisfied: scipy>=1.6.0 in c:\users\yogit\appdata\local\progr
ams\python\python312\lib\site-packages (from scikit-learn) (1.14.0)
Requirement already satisfied: joblib>=1.2.0 in c:\users\yogit\appdata\local\prog
rams\python\python312\lib\site-packages (from scikit-learn) (1.4.2)
Collecting threadpoolctl>=3.1.0 (from scikit-learn)
  Downloading threadpoolctl-3.5.0-py3-none-any.whl.metadata (13 kB)
Downloading scikit_learn-1.5.1-cp312-cp312-win_amd64.whl (10.9 MB)
   ---------------------------------------- 0.0/10.9 MB ? eta -:--:--
   --- ------------------------------------ 0.9/10.9 MB 28.4 MB/s eta 0:00:01
   ----- ---------------------------------- 1.4/10.9 MB 23.1 MB/s eta 0:00:01
   ----- ---------------------------------- 1.4/10.9 MB 23.1 MB/s eta 0:00:01
   ----- ---------------------------------- 1.4/10.9 MB 23.1 MB/s eta 0:00:01
   ----- ---------------------------------- 1.4/10.9 MB 23.1 MB/s eta 0:00:01
   ----- ---------------------------------- 1.4/10.9 MB 23.1 MB/s eta 0:00:01
   ------ --------------------------------- 1.6/10.9 MB 5.3 MB/s eta 0:00:02
   -------- ------------------------------- 2.4/10.9 MB 6.6 MB/s eta 0:00:02
   ------------ --------------------------- 3.3/10.9 MB 8.2 MB/s eta 0:00:01
   ------------ --------------------------- 3.5/10.9 MB 7.6 MB/s eta 0:00:01
   ------------- -------------------------- 3.8/10.9 MB 7.5 MB/s eta 0:00:01
   ------------- -------------------------- 3.8/10.9 MB 7.1 MB/s eta 0:00:02
   -------------- ------------------------- 4.0/10.9 MB 6.6 MB/s eta 0:00:02
   --------------- ------------------------ 4.3/10.9 MB 6.8 MB/s eta 0:00:01
   ---------------- ----------------------- 4.7/10.9 MB 6.8 MB/s eta 0:00:01
   ---------------- ----------------------- 4.9/10.9 MB 6.7 MB/s eta 0:00:01
   ----------------- ---------------------- 5.0/10.9 MB 6.5 MB/s eta 0:00:01
   ------------------ --------------------- 5.2/10.9 MB 6.3 MB/s eta 0:00:01
   ------------------ --------------------- 5.4/10.9 MB 6.2 MB/s eta 0:00:01
   ------------------- -------------------- 5.7/10.9 MB 6.1 MB/s eta 0:00:01
   ------------------- -------------------- 5.9/10.9 MB 6.1 MB/s eta 0:00:01
   -------------------- ------------------- 6.2/10.9 MB 6.1 MB/s eta 0:00:01
   --------------------- ------------------ 6.4/10.9 MB 6.1 MB/s eta 0:00:01
   --------------------- ------------------ 6.7/10.9 MB 6.1 MB/s eta 0:00:01
   ---------------------- ----------------- 7.1/10.9 MB 6.2 MB/s eta 0:00:01
   ---------------------- ----------------- 7.5/10.9 MB 6.2 MB/s eta 0:00:01
   ----------------------- ---------------- 7.9/10.9 MB 6.3 MB/s eta 0:00:01
   ------------------------ --------------- 8.3/10.9 MB 6.4 MB/s eta 0:00:01
   ------------------------ --------------- 8.7/10.9 MB 6.5 MB/s eta 0:00:01
   ------------------------- -------------- 9.1/10.9 MB 6.6 MB/s eta 0:00:01
   ------------------------- -------------- 9.4/10.9 MB 6.5 MB/s eta 0:00:01
   -------------------------- ------------- 9.6/10.9 MB 6.5 MB/s eta 0:00:01
   -------------------------- ------------- 9.7/10.9 MB 6.4 MB/s eta 0:00:01
   --------------------------- ------------ 9.9/10.9 MB 6.2 MB/s eta 0:00:01
   --------------------------- ------------ 10.0/10.9 MB 6.1 MB/s eta 0:00:01
   --------------------------- ------------ 10.1/10.9 MB 6.0 MB/s eta 0:00:01
   ---------------------------- ----------- 10.2/10.9 MB 6.0 MB/s eta 0:00:01
   ---------------------------- ----------- 10.4/10.9 MB 5.8 MB/s eta 0:00:01
   ---------------------------- ----------- 10.6/10.9 MB 5.7 MB/s eta 0:00:01
   ---------------------------- -----------  10.8/10.9 MB 5.6 MB/s eta 0:00:01
   ---------------------------- -----------  10.9/10.9 MB 5.5 MB/s eta 0:00:01
   ---------------------------------------- 10.9/10.9 MB 5.4 MB/s eta 0:00:00
Downloading threadpoolctl-3.5.0-py3-none-any.whl (18 kB)
Installing collected packages: threadpoolctl, scikit-learn
Successfully installed scikit-learn-1.5.1 threadpoolctl-3.5.0
```

In [37]:
```
from sklearn.metrics import classification_report
```

```
# Example data (replace with your actual data)
true_labels = ['yes', 'no', 'yes', 'yes', 'no', 'yes', 'no', 'no', 'yes', 'yes']
predicted_labels = ['yes', 'yes', 'yes', 'yes', 'no', 'no', 'yes', 'no', 'yes',
# Compute classification report
report = classification_report(true_labels, predicted_labels)

# Display the report
print(report)
```

```
              precision    recall  f1-score   support

          no       0.67      0.50      0.57         4
         yes       0.71      0.83      0.77         6

    accuracy                           0.70        10
   macro avg       0.69      0.67      0.67        10
weighted avg       0.70      0.70      0.69        10
```

In [38]:
```
from sklearn.tree import plot_tree
```

In [40]:
```
cn = ['no','yes']
print(cn)
```

```
['no', 'yes']
```

In [41]:
```
import numpy as np
from sklearn.datasets import load_iris

# Load example dataset (you can replace this with your own dataset)
iris = load_iris()
X = iris.data
y = iris.target
feature_names = iris.feature_names
target_names = iris.target_names
```

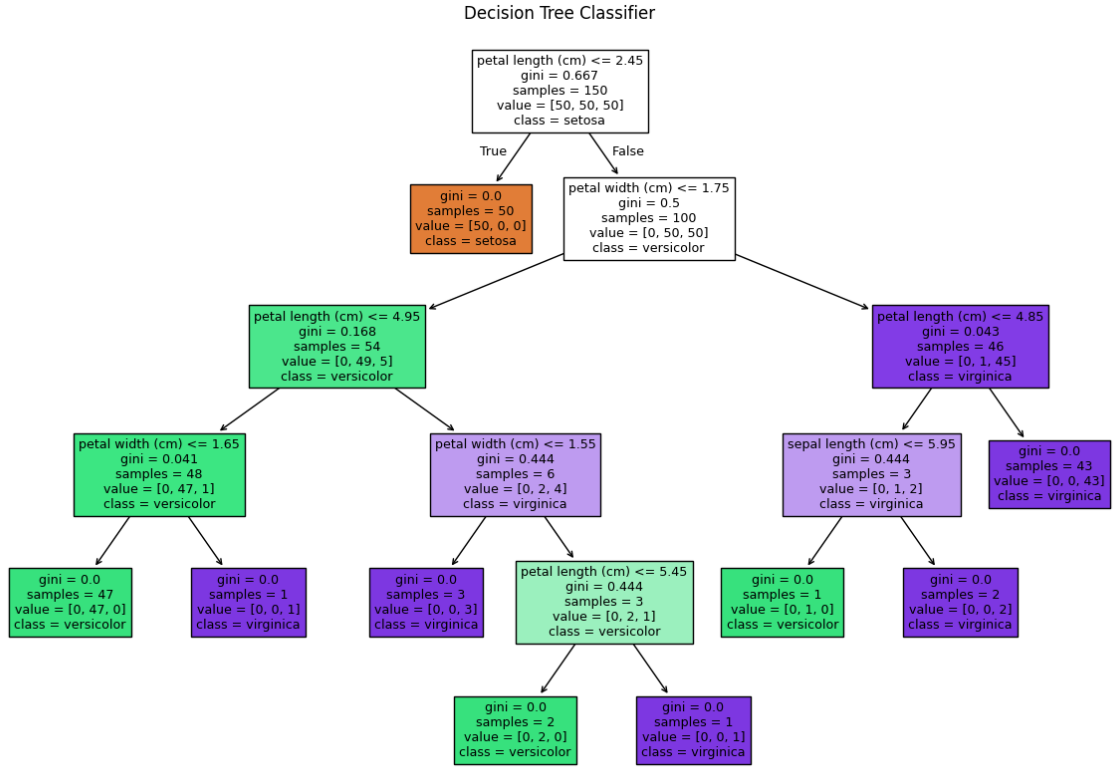
In [42]:
```
from sklearn.tree import DecisionTreeClassifier

# Create a decision tree classifier
clf = DecisionTreeClassifier(random_state=42)

# Train the classifier on the data
clf.fit(X, y)
```

Out[42]:

▾ DecisionTreeClassifier

```
DecisionTreeClassifier(random_state=42)
```

In [43]:
```
from sklearn.tree import plot_tree
import matplotlib.pyplot as plt

plt.figure(figsize=(15, 10))
plot_tree(clf, feature_names=feature_names, class_names=target_names, filled=Tru
plt.title("Decision Tree Classifier")
plt.show()
```


Decision Tree Classifier
petal length (cm) <= 2.45
gini = 0.667
samples = 150
value = [50, 50, 50]
class = setosa
True
False
gini = 0.0
samples = 50
value = [50, 0, 0]
class = setosa
petal width (cm) <= 1.75
gini = 0.5
samples = 100
value = [0, 50, 50]
class = versicolor
petal length (cm) <= 4.95
gini = 0.168
samples = 54
value = [0, 49, 5]
class = versicolor
petal length (cm) <= 4.85
gini = 0.043
samples = 46
value = [0, 1, 45]
class = virginica
petal width (cm) <= 1.65
gini = 0.041
samples = 48
value = [0, 47, 1]
class = versicolor
petal width (cm) <= 1.55
gini = 0.444
samples = 6
value = [0, 2, 4]
class = virginica
sepal length (cm) <= 5.95
gini = 0.444
samples = 3
value = [0, 1, 2]
class = virginica
gini = 0.0
samples = 43
value = [0, 0, 43]
class = virginica
gini = 0.0
samples = 47
value = [0, 47, 0]
class = versicolor
gini = 0.0
samples = 1
value = [0, 0, 1]
class = virginica
gini = 0.0
samples = 3
value = [0, 0, 3]
class = virginica
petal length (cm) <= 5.45
gini = 0.444
samples = 3
value = [0, 2, 1]
class = versicolor
gini = 0.0
samples = 1
value = [0, 1, 0]
class = versicolor
gini = 0.0
samples = 2
value = [0, 0, 2]
class = virginica
gini = 0.0
samples = 2
value = [0, 2, 0]
class = versicolor
gini = 0.0
samples = 1
value = [0, 0, 1]
class = virginica

In [ ]: