In [2]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Load the dataset
df = pd.read_csv("C:\\Users\\prems\\Videos\\power bi\\Telco_Cusomer_Churn.csv")

# Basic information about the dataset
print("Dataset Shape:", df.shape)
print("\n" + "="*60)
print("\nFirst 5 rows:")
print(df.head())
print("\n" + "="*60)
print("\nDataset Information:")
print(df.info())
print("\n" + "="*60)
print("\nMissing Values:")
print(df.isnull().sum())

# Convert TotalCharges to numeric (handling empty strings)
df['TotalCharges'] = pd.to_numeric(df['TotalCharges'], errors='coerce')
```

```
Dataset Shape: (7043, 21)

============================================================

First 5 rows:
   customerID  gender  SeniorCitizen Partner Dependents  tenure PhoneService  \
0  7590-VHVEG  Female              0     Yes         No       1           No
1  5575-GNVDE    Male              0      No         No      34          Yes
2  3668-QPYBK    Male              0      No         No       2          Yes
3  7795-CFOCW    Male              0      No         No      45           No
4  9237-HQITU  Female              0      No         No       2          Yes

      MultipleLines InternetService OnlineSecurity  ... DeviceProtection  \
0  No phone service             DSL             No  ...               No
1                No             DSL            Yes  ...              Yes
2                No             DSL            Yes  ...               No
3  No phone service             DSL            Yes  ...              Yes
4                No     Fiber optic             No  ...               No

  TechSupport StreamingTV StreamingMovies        Contract PaperlessBilling  \
0          No          No              No  Month-to-month              Yes
1          No          No              No        One year               No
2          No          No              No  Month-to-month              Yes
3         Yes          No              No        One year               No
4          No          No              No  Month-to-month              Yes

               PaymentMethod MonthlyCharges  TotalCharges Churn
0           Electronic check          29.85         29.85    No
1               Mailed check          56.95        1889.5    No
2               Mailed check          53.85        108.15   Yes
3  Bank transfer (automatic)          42.30       1840.75    No
4           Electronic check          70.70        151.65   Yes

[5 rows x 21 columns]
```

```
============================================================

Dataset Information:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7043 entries, 0 to 7042
Data columns (total 21 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   customerID        7043 non-null   object 
 1   gender            7043 non-null   object 
 2   SeniorCitizen     7043 non-null   int64  
 3   Partner           7043 non-null   object 
 4   Dependents        7043 non-null   object 
 5   tenure            7043 non-null   int64  
 6   PhoneService      7043 non-null   object 
 7   MultipleLines     7043 non-null   object 
 8   InternetService   7043 non-null   object 
 9   OnlineSecurity    7043 non-null   object 
 10  OnlineBackup      7043 non-null   object 
 11  DeviceProtection  7043 non-null   object 
 12  TechSupport       7043 non-null   object 
 13  StreamingTV       7043 non-null   object 
 14  StreamingMovies   7043 non-null   object 
 15  Contract          7043 non-null   object 
 16  PaperlessBilling  7043 non-null   object 
 17  PaymentMethod     7043 non-null   object 
 18  MonthlyCharges    7043 non-null   float64
 19  TotalCharges      7043 non-null   object 
 20  Churn             7043 non-null   object 
dtypes: float64(1), int64(2), object(18)
memory usage: 1.1+ MB
None

============================================================

Missing Values:
customerID          0
gender              0
SeniorCitizen       0
Partner             0
Dependents          0
tenure              0
PhoneService        0
MultipleLines       0
InternetService     0
OnlineSecurity      0
OnlineBackup        0
DeviceProtection    0
TechSupport         0
StreamingTV         0
StreamingMovies     0
Contract            0
PaperlessBilling    0
PaymentMethod       0
MonthlyCharges      0
TotalCharges        0
Churn               0
dtype: int64
```

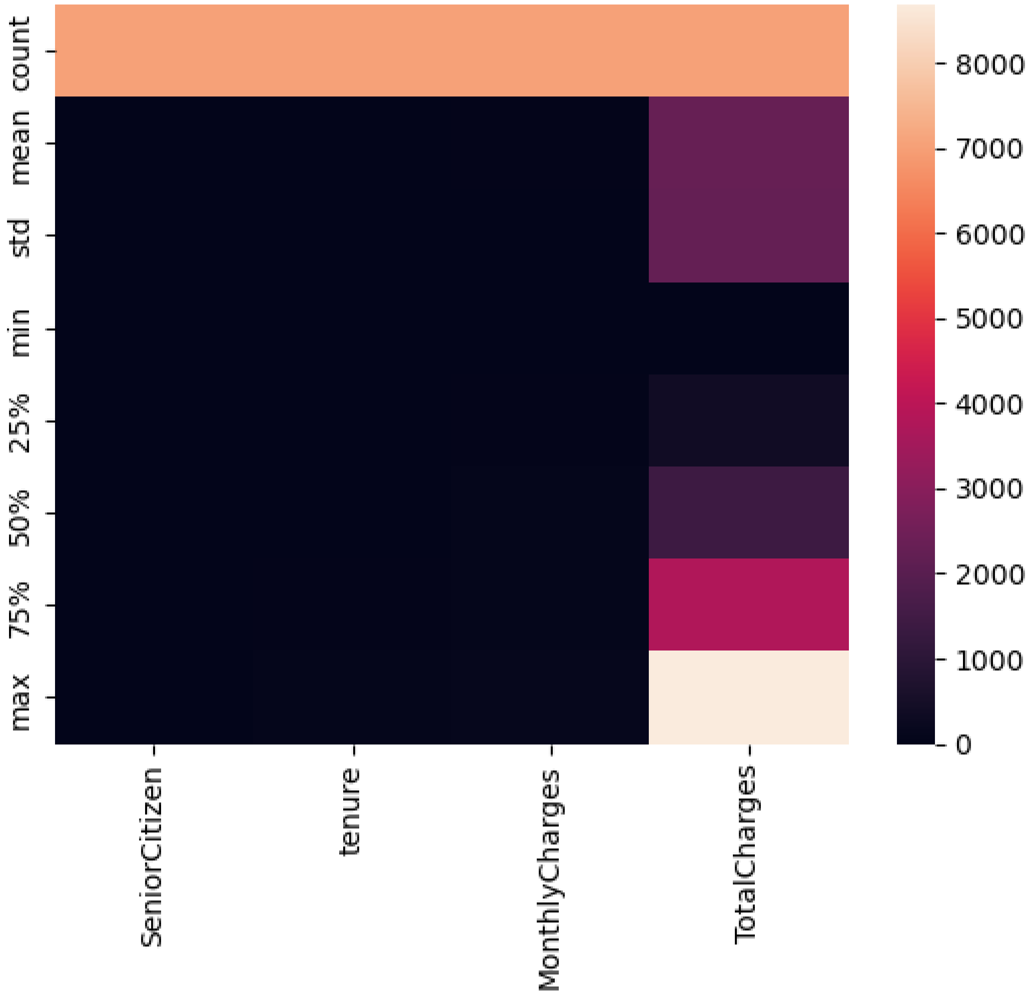
In [18]:

```
# Check basic statistics
print("\nBasic Statistics:")
print(df.describe())
sns.heatmap(df.describe())
```

```
Basic Statistics:
       SeniorCitizen       tenure  MonthlyCharges  TotalCharges
count    7043.000000  7043.000000     7043.000000   7032.000000
mean        0.162147    32.371149       64.761692   2283.300441
std         0.368612    24.559481       30.090047   2266.771362
min         0.000000     0.000000       18.250000     18.800000
25%         0.000000     9.000000       35.500000    401.450000
50%         0.000000    29.000000       70.350000   1397.475000
75%         0.000000    55.000000       89.850000   3794.737500
max         1.000000    72.000000      118.750000   8684.800000
```

Out[18]:

```
<Axes: >
```



In [ ]:

```
df.style.set_caption('describe the data')
```

In [30]:

```
print("\nisnull:")
print(df.isnull())
print("\n" + "="*60)
print("\nisnullsum:")
print(df.isnull().sum())
```

```
df.add_prefix('lefttone_')
df.add_suffix('_righttone')
```

```
isnull:
      customerID  gender  SeniorCitizen  Partner  Dependents  tenure  \
0          False   False          False    False       False   False   
1          False   False          False    False       False   False   
2          False   False          False    False       False   False   
3          False   False          False    False       False   False   
4          False   False          False    False       False   False   
...          ...     ...            ...      ...         ...     ...   
7038       False   False          False    False       False   False   
7039       False   False          False    False       False   False   
7040       False   False          False    False       False   False   
7041       False   False          False    False       False   False   
7042       False   False          False    False       False   False   

      PhoneService  MultipleLines  InternetService  OnlineSecurity  ...  \
0            False          False            False           False  ...   
1            False          False            False           False  ...   
2            False          False            False           False  ...   
3            False          False            False           False  ...   
4            False          False            False           False  ...   
...            ...            ...              ...             ...  ...   
7038         False          False            False           False  ...   
7039         False          False            False           False  ...   
7040         False          False            False           False  ...   
7041         False          False            False           False  ...   
7042         False          False            False           False  ...   

      DeviceProtection  TechSupport  StreamingTV  StreamingMovies  Contract  \
0                False        False        False            False     False   
1                False        False        False            False     False   
2                False        False        False            False     False   
3                False        False        False            False     False   
4                False        False        False            False     False   
...                ...          ...          ...              ...       ...   
7038             False        False        False            False     False   
7039             False        False        False            False     False   
7040             False        False        False            False     False   
7041             False        False        False            False     False   
7042             False        False        False            False     False   

      PaperlessBilling  PaymentMethod  MonthlyCharges  TotalCharges  Churn  
0                False          False           False         False  False  
1                False          False           False         False  False  
2                False          False           False         False  False  
3                False          False           False         False  False  
4                False          False           False         False  False  
...                ...            ...             ...           ...    ...  
7038             False          False           False         False  False  
7039             False          False           False         False  False  
7040             False          False           False         False  False  
7041             False          False           False         False  False  
7042             False          False           False         False  False  

[7043 rows x 21 columns]

============================================================
isnullsum:
customerID           0
gender               0
SeniorCitizen        0
Partner              0
Dependents           0
tenure               0
PhoneService         0
MultipleLines        0
InternetService      0
OnlineSecurity       0
OnlineBackup         0
DeviceProtection     0
TechSupport          0
StreamingTV          0
StreamingMovies      0
Contract             0
PaperlessBilling     0
PaymentMethod        0
MonthlyCharges       0
TotalCharges        11
Churn                0
dtype: int64
```

Out[30]:

	customerID_righttone	gender_righttone	SeniorCitizen_righttone	Partner_righttone	Dependents_rig
0	7590-VHVEG	Female	0	Yes	
1	5575-GNVDE	Male	0	No	
2	3668-QPYBK	Male	0	No	
3	7795-CFOCW	Male	0	No	
4	9237-HQITU	Female	0	No	
...	...	...	...	...	
7038	6840-RESVB	Male	0	Yes	
7039	2234-XADUH	Female	0	Yes	
7040	4801-JZAZL	Female	0	Yes	
7041	8361-LTMKD	Male	1	Yes	
7042	3186-AJIEK	Male	0	No	

7043 rows × 21 columns

In [13]:

```
print("\nremoving the missing values:")
print(df.dropna())
```

```
removing the missing values:
      customerID  gender  SeniorCitizen Partner Dependents  tenure  \
0     7590-VHVEG  Female              0     Yes         No       1
1     5575-GNVDE    Male              0      No         No      34
2     3668-QPYBK    Male              0      No         No       2
3     7795-CFOCW    Male              0      No         No      45
4     9237-HQITU  Female              0      No         No       2
```

```
...          ...     ...            ...     ...        ...     ...
7038  6840-RESVB    Male              0     Yes        Yes      24
7039  2234-XADUH  Female              0     Yes        Yes      72
7040  4801-JZAZL  Female              0     Yes        Yes      11
7041  8361-LTMKD    Male              1     Yes         No       4
7042  3186-AJIEK    Male              0      No         No      66

     PhoneService     MultipleLines InternetService OnlineSecurity  ...  \
0              No  No phone service             DSL             No  ...   
1             Yes                No             DSL            Yes  ...   
2             Yes                No             DSL            Yes  ...   
3              No  No phone service             DSL            Yes  ...   
4             Yes                No     Fiber optic             No  ...   
...           ...               ...             ...            ...  ...   
7038          Yes               Yes             DSL            Yes  ...   
7039          Yes               Yes     Fiber optic             No  ...   
7040           No  No phone service             DSL            Yes  ...   
7041          Yes               Yes     Fiber optic             No  ...   
7042          Yes                No     Fiber optic            Yes  ...   

     DeviceProtection TechSupport StreamingTV StreamingMovies        Contract  \
0                  No          No          No              No  Month-to-month   
1                 Yes          No          No              No        One year   
2                  No          No          No              No  Month-to-month   
3                 Yes         Yes          No              No        One year   
4                  No          No          No              No  Month-to-month   
...               ...         ...         ...             ...             ...   
7038              Yes         Yes         Yes             Yes        One year   
7039              Yes          No         Yes             Yes        One year   
7040               No          No          No              No  Month-to-month   
7041               No          No          No              No  Month-to-month   
7042              Yes         Yes         Yes             Yes        Two year   

     PaperlessBilling              PaymentMethod MonthlyCharges  TotalCharges  \
0                 Yes           Electronic check          29.85         29.85   
1                  No               Mailed check          56.95       1889.50   
2                 Yes               Mailed check          53.85        108.15   
3                  No  Bank transfer (automatic)          42.30       1840.75   
4                 Yes           Electronic check          70.70        151.65   
...               ...                        ...            ...           ...   
7038              Yes               Mailed check          84.80       1990.50   
7039              Yes    Credit card (automatic)         103.20       7362.90   
7040              Yes           Electronic check          29.60        346.45   
7041              Yes               Mailed check          74.40        306.60   
7042              Yes  Bank transfer (automatic)         105.65       6844.50   

      Churn  
0        No  
1        No  
2       Yes  
3        No  
4       Yes  
...     ...  
7038     No  
7039     No  
7040     No  
7041    Yes  
7042     No  
```

```
[7032 rows x 21 columns]
```

In [4]:

```
print(df.columns)
```

```
Index(['customerID', 'gender', 'SeniorCitizen', 'Partner', 'Dependents',
       'tenure', 'PhoneService', 'MultipleLines', 'InternetService',
       'OnlineSecurity', 'OnlineBackup', 'DeviceProtection', 'TechSupport',
       'StreamingTV', 'StreamingMovies', 'Contract', 'PaperlessBilling',
       'PaymentMethod', 'MonthlyCharges', 'TotalCharges', 'Churn'],
      dtype='object')
```

gender is male

In [34]:

```
df[df.gender=='Male']
```

Out[34]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	Int
1	5575-GNVDE	Male	0	No	No	34	Yes	No	
2	3668-QPYBK	Male	0	No	No	2	Yes	No	
3	7795-CFOCW	Male	0	No	No	45	No	No phone service	
6	1452-KIOVK	Male	0	No	Yes	22	Yes	Yes	
9	6388-TABGU	Male	0	No	Yes	62	Yes	No	
...	...	...	...	...	...	...	...	...	
7033	9767-FFLEM	Male	0	No	No	38	Yes	No	
7035	8456-QDAVC	Male	0	No	No	19	Yes	No	
7038	6840-RESVB	Male	0	Yes	Yes	24	Yes	Yes	
7041	8361-LTMKD	Male	1	Yes	No	4	Yes	Yes	
7042	3186-AJIEK	Male	0	No	No	66	Yes	No	

3555 rows × 21 columns

In [ ]:

```
sns.displot(df.gender=='male')
```

rename columns like customerid to id

In [13]:

```
df.rename(columns={'customerID':'id'}) #inplace=true (permenent )
```

Out[13]:

	id	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	Intern
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service	
1	5575-GNVDE	Male	0	No	No	34	Yes	No	
2	3668-QPYBK	Male	0	No	No	2	Yes	No	
3	7795-CFOCW	Male	0	No	No	45	No	No phone service	
4	9237-HQITU	Female	0	No	No	2	Yes	No	I
...	...	...	...	...	...	...	...	...	
7038	6840-RESVB	Male	0	Yes	Yes	24	Yes	Yes	
7039	2234-XADUH	Female	0	Yes	Yes	72	Yes	Yes	I
7040	4801-JZAZL	Female	0	Yes	Yes	11	No	No phone service	
7041	8361-LTMKD	Male	1	Yes	No	4	Yes	Yes	I
7042	3186-AJIEK	Male	0	No	No	66	Yes	No	I

7043 rows × 21 columns

In [16]:

```
df[df['InternetService'].str.contains('DSL')].head(10)
```

Out[16]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	Inter
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service	
1	5575-GNVDE	Male	0	No	No	34	Yes	No	
2	3668-QPYBK	Male	0	No	No	2	Yes	No	
3	7795-CFOCW	Male	0	No	No	45	No	No phone service	
7	6713-OKOMC	Female	0	No	No	10	No	No phone service	
9	6388-TABGU	Male	0	No	Yes	62	Yes	No	
10	9763-GRSKD	Male	0	Yes	Yes	13	Yes	No	

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	Inter
18	4190-MFLUW	Female	0	Yes	Yes	10	Yes	No	
20	8779-QRDMV	Male	1	No	No	1	No	No phone service	
23	3638-WEABW	Female	0	Yes	No	58	Yes	Yes	

10 rows × 21 columns

find all instance when 'gender is male' and InternetService is DSL'

In [17]:

```
df[(df['gender']=='Male')&(df['InternetService']=='DSL')]
```

Out[17]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	Int
1	5575-GNVDE	Male	0	No	No	34	Yes	No	
2	3668-QPYBK	Male	0	No	No	2	Yes	No	
3	7795-CFOCW	Male	0	No	No	45	No	No phone service	
9	6388-TABGU	Male	0	No	Yes	62	Yes	No	
10	9763-GRSKD	Male	0	Yes	Yes	13	Yes	No	
...	...	...	...	...	...	...	...	...	
7007	2274-XUATA	Male	1	Yes	No	72	No	No phone service	
7021	1699-HPSBG	Male	0	No	No	12	Yes	No	
7027	0550-DCXLH	Male	0	No	No	13	Yes	No	
7031	3605-JISKB	Male	1	Yes	No	55	Yes	Yes	
7038	6840-RESVB	Male	0	Yes	Yes	24	Yes	Yes	

1233 rows × 21 columns

In [20]:

```
df[(df['gender']=='Male')&(df['InternetService']=='DSL')|(df['OnlineSecurity']=='Yes')]
```

Out[20]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	Int
1	5575-GNVDE	Male	0	No	No	34	Yes	No	
2	3668-QPYBK	Male	0	No	No	2	Yes	No	
3	7795-CFOCW	Male	0	No	No	45	No	No phone service	
7	6713-OKOMC	Female	0	No	No	10	No	No phone service	
9	6388-TABGU	Male	0	No	Yes	62	Yes	No	
...	...	...	...	...	...	...	...	...	
7031	3605-JISKB	Male	1	Yes	No	55	Yes	Yes	
7034	0639-TSIQW	Female	0	No	No	67	Yes	Yes	
7038	6840-RESVB	Male	0	Yes	Yes	24	Yes	Yes	
7040	4801-JZAZL	Female	0	Yes	Yes	11	No	No phone service	
7042	3186-AJIEK	Male	0	No	No	66	Yes	No	

2666 rows × 21 columns

In [23]:

```
df[df['PaymentMethod'].isin(['Electronic check','Mailed check'])]
```

Out[23]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	Int
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service	
1	5575-GNVDE	Male	0	No	No	34	Yes	No	
2	3668-QPYBK	Male	0	No	No	2	Yes	No	
4	9237-HQITU	Female	0	No	No	2	Yes	No	
5	9305-CDSKC	Female	0	No	No	8	Yes	Yes	
...	...	...	...	...	...	...	...	...	
7032	6894-LFHLY	Male	1	No	No	1	Yes	Yes	
7036	7750-EYXWZ	Female	0	No	No	12	No	No phone service	
7038	6840-	Male	0	Yes	Yes	24	Yes	Yes	

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	Int
	RESVB								
7040	4801-JZAZL	Female	0	Yes	Yes	11	No	No phone service	
7041	8361-LTMKD	Male	1	Yes	No	4	Yes	Yes	

3977 rows × 21 columns

remove duplicates

In [24]:

```
df[~(df['gender']=='Male')]
```

Out[24]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	Int
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service	
4	9237-HQITU	Female	0	No	No	2	Yes	No	
5	9305-CDSKC	Female	0	No	No	8	Yes	Yes	
7	6713-OKOMC	Female	0	No	No	10	No	No phone service	
8	7892-POOKP	Female	0	Yes	No	28	Yes	Yes	
...	...	...	...	...	...	...	...	...	
7034	0639-TSIQW	Female	0	No	No	67	Yes	Yes	
7036	7750-EYXWZ	Female	0	No	No	12	No	No phone service	
7037	2569-WGERO	Female	0	No	No	72	Yes	No	
7039	2234-XADUH	Female	0	Yes	Yes	72	Yes	Yes	
7040	4801-JZAZL	Female	0	Yes	Yes	11	No	No phone service	

3488 rows × 21 columns

In [31]:

```
df['tenure']= df['tenure'].apply(lambda x:x+3)
```

In [32]:

```
df
```

Out[32]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	Int
0	7590-VHVEG	Female	0	Yes	No	4	No	No phone service	
1	5575-GNVDE	Male	0	No	No	37	Yes	No	
2	3668-QPYBK	Male	0	No	No	5	Yes	No	
3	7795-CFOCW	Male	0	No	No	48	No	No phone service	
4	9237-HQITU	Female	0	No	No	5	Yes	No	
...	...	...	...	...	...	...	...	...	
7038	6840-RESVB	Male	0	Yes	Yes	27	Yes	Yes	
7039	2234-XADUH	Female	0	Yes	Yes	75	Yes	Yes	
7040	4801-JZAZL	Female	0	Yes	Yes	14	No	No phone service	
7041	8361-LTMKD	Male	1	Yes	No	7	Yes	Yes	
7042	3186-AJIEK	Male	0	No	No	69	Yes	No	

7043 rows × 21 columns

In [33]:

```
df[df.PaymentMethod=='Mailed check'].gender.value_counts()
```

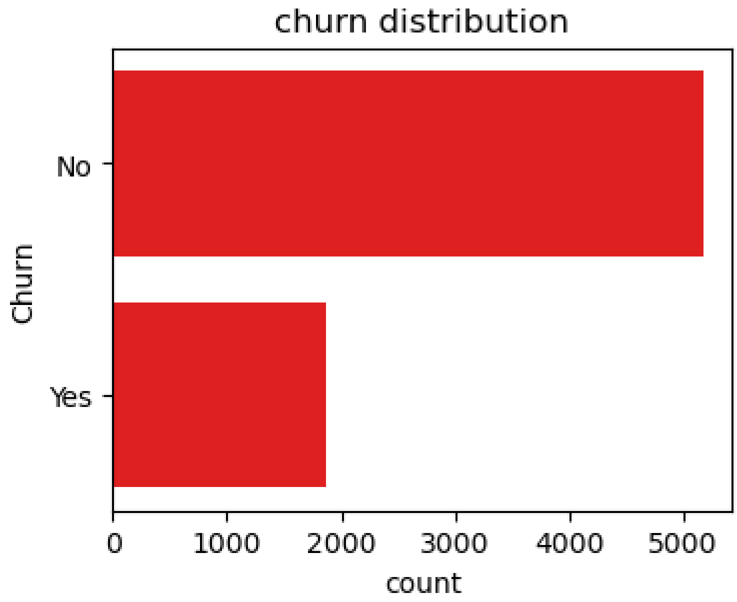
Out[33]:

```
gender
Male      834
Female    778
Name: count, dtype: int64
```

what is the distribution of churn(yes/no)?

In [41]:

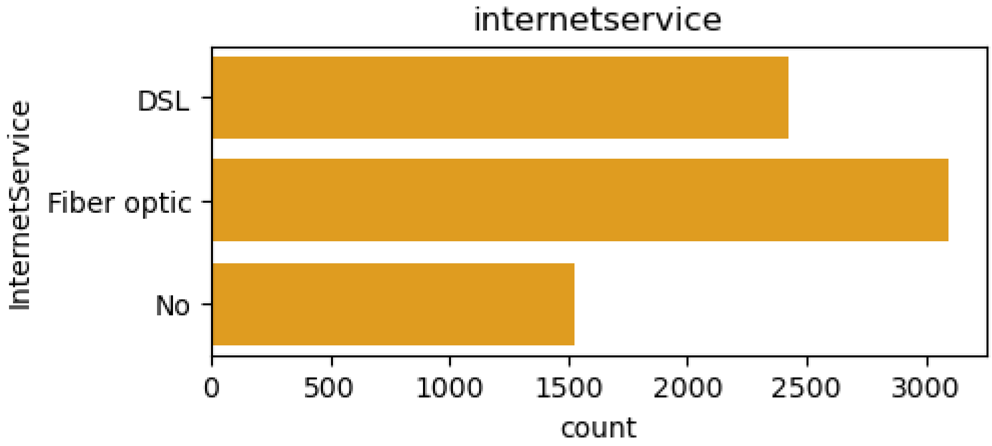
```
plt.figure(figsize=(4,3))
sns.countplot(df['Churn'],color='red')
plt.title('churn distribution')
plt.show()
```



what is the distribution of internet service types?

In [42]:

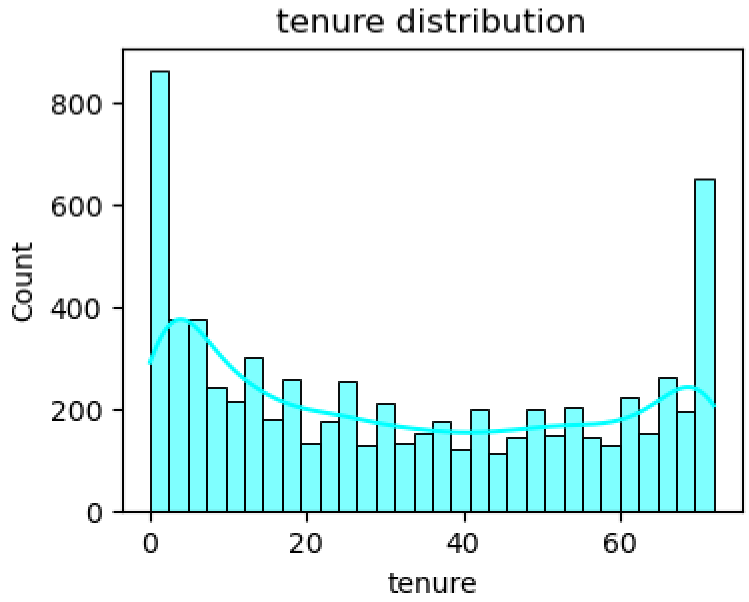
```
plt.figure(figsize=(5,2))
sns.countplot(df['InternetService'],color='orange')
plt.title('internetservice')
plt.show()
```



how is the tenure distribution among customers?

In [43]:

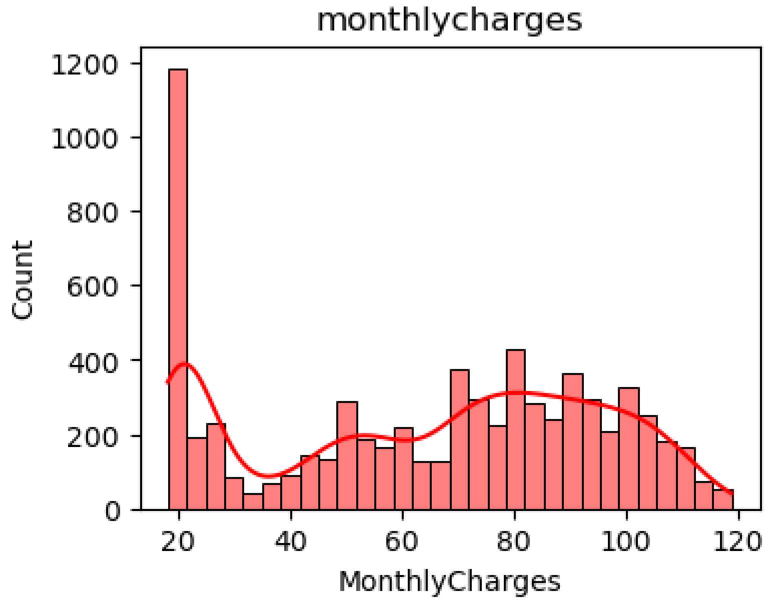
```
plt.figure(figsize=(4,3))
sns.histplot(df['tenure'],color='cyan',kde=True,bins=30)
plt.title('tenure distribution')
plt.show()
```



what is the distribution of monthly charges?

In [44]:

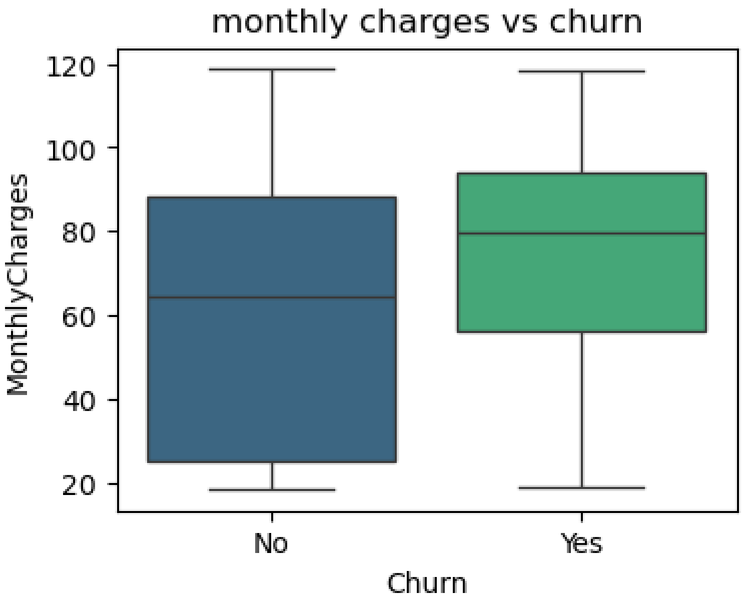
```
plt.figure(figsize=(4,3))
sns.histplot(df['MonthlyCharges'],color='red',bins=30,kde=True,)
plt.title('monthlycharges')
plt.show()
```



do monthly charges differ between churned and non-churned customers?

In [46]:

```
plt.figure(figsize=(4,3))
sns.boxplot(df,x='Churn',y='MonthlyCharges',hue='Churn',palette='viridis')
plt.title('monthly charges vs churn')
plt.show()
```



In [24]:

```
print("Duplicate Customer IDs:")
duplicate_ids = df[df['customerID'].duplicated(keep=False)]
print(f"Found {len(duplicate_ids)} duplicate customer IDs")
```

```
Duplicate Customer IDs:
Found 0 duplicate customer IDs
```

In [4]:

```
print("\nduplicates:")
df[df.duplicated()]
```

```
duplicates:
```

Out[4]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	Interne

0 rows × 21 columns

In [8]:

```
print("\ndrop duplicates :")
print(df.drop_duplicates())
```

```
drop duplicates :
      customerID  gender  SeniorCitizen Partner Dependents  tenure  \
0     7590-VHVEG  Female              0     Yes         No       1   
1     5575-GNVDE    Male              0      No         No      34   
2     3668-QPYBK    Male              0      No         No       2   
3     7795-CFOCW    Male              0      No         No      45   
4     9237-HQITU  Female              0      No         No       2   
...          ...     ...            ...     ...        ...     ...   
7038  6840-RESVB    Male              0     Yes        Yes      24   
7039  2234-XADUH  Female              0     Yes        Yes      72   
7040  4801-JZAZL  Female              0     Yes        Yes      11   
7041  8361-LTMKD    Male              1     Yes         No       4   
7042  3186-AJIEK    Male              0      No         No      66   

     PhoneService     MultipleLines InternetService OnlineSecurity  ...  \
0              No  No phone service             DSL             No  ...   
1             Yes                No             DSL            Yes  ...   
```

```
2             Yes                No          DSL            Yes  ...
3              No  No phone service          DSL            Yes  ...
4             Yes                No  Fiber optic             No  ...
...           ...               ...          ...            ...  ...
7038          Yes               Yes          DSL            Yes  ...
7039          Yes               Yes  Fiber optic             No  ...
7040           No  No phone service          DSL            Yes  ...
7041          Yes               Yes  Fiber optic             No  ...
7042          Yes                No  Fiber optic            Yes  ...

     DeviceProtection TechSupport StreamingTV StreamingMovies        Contract  \
0                  No          No          No              No  Month-to-month   
1                 Yes          No          No              No        One year   
2                  No          No          No              No  Month-to-month   
3                 Yes         Yes          No              No        One year   
4                  No          No          No              No  Month-to-month   
...               ...         ...         ...             ...             ...   
7038              Yes         Yes         Yes             Yes        One year   
7039              Yes          No         Yes             Yes        One year   
7040               No          No          No              No  Month-to-month   
7041               No          No          No              No  Month-to-month   
7042              Yes         Yes         Yes             Yes        Two year   

     PaperlessBilling              PaymentMethod MonthlyCharges  TotalCharges  \
0                 Yes           Electronic check          29.85         29.85   
1                  No               Mailed check          56.95       1889.50   
2                 Yes               Mailed check          53.85        108.15   
3                  No  Bank transfer (automatic)          42.30       1840.75   
4                 Yes           Electronic check          70.70        151.65   
...               ...                        ...            ...           ...   
7038              Yes               Mailed check          84.80       1990.50   
7039              Yes    Credit card (automatic)         103.20       7362.90   
7040              Yes           Electronic check          29.60        346.45   
7041              Yes               Mailed check          74.40        306.60   
7042              Yes  Bank transfer (automatic)         105.65       6844.50   

      Churn  
0        No  
1        No  
2       Yes  
3        No  
4       Yes  
...     ...  
7038     No  
7039     No  
7040     No  
7041    Yes  
7042     No  

[7043 rows x 21 columns]
```

In [16]:

```
print("\n transpose the data:")
print(df.T)
```

```
 transpose the data:
                              0             1               2     \
customerID              7590-VHVEG    5575-GNVDE      3668-QPYBK   
gender                      Female          Male            Male   
```

```
SeniorCitizen                    0             0               0
Partner                        Yes            No              No
Dependents                      No            No              No
tenure                           1            34               2
PhoneService                    No           Yes             Yes
MultipleLines     No phone service            No              No
InternetService                DSL           DSL             DSL
OnlineSecurity                  No           Yes             Yes
OnlineBackup                   Yes            No             Yes
DeviceProtection                No           Yes              No
TechSupport                     No            No              No
StreamingTV                     No            No              No
StreamingMovies                 No            No              No
Contract            Month-to-month      One year  Month-to-month
PaperlessBilling               Yes            No             Yes
PaymentMethod     Electronic check  Mailed check    Mailed check
MonthlyCharges               29.85         56.95           53.85
TotalCharges                 29.85        1889.5          108.15
Churn                           No            No             Yes

                                       3                 4     \
customerID                        7795-CFOCW        9237-HQITU
gender                                  Male            Female
SeniorCitizen                              0                 0
Partner                                   No                No
Dependents                                No                No
tenure                                    45                 2
PhoneService                              No               Yes
MultipleLines               No phone service                No
InternetService                          DSL       Fiber optic
OnlineSecurity                           Yes                No
OnlineBackup                              No                No
DeviceProtection                         Yes                No
TechSupport                              Yes                No
StreamingTV                               No                No
StreamingMovies                           No                No
Contract                            One year    Month-to-month
PaperlessBilling                          No               Yes
PaymentMethod     Bank transfer (automatic)  Electronic check
MonthlyCharges                          42.3              70.7
TotalCharges                         1840.75            151.65
Churn                                     No               Yes

                              5                        6                 7     \
customerID               9305-CDSKC               1452-KIOVK        6713-OKOMC
gender                       Female                     Male            Female
SeniorCitizen                     0                        0                 0
Partner                          No                       No                No
Dependents                       No                      Yes                No
tenure                            8                       22                10
PhoneService                    Yes                      Yes                No
MultipleLines                   Yes                      Yes  No phone service
InternetService         Fiber optic              Fiber optic               DSL
OnlineSecurity                   No                       No               Yes
OnlineBackup                     No                      Yes                No
DeviceProtection                Yes                       No                No
TechSupport                      No                       No                No
StreamingTV                     Yes                      Yes                No
StreamingMovies                 Yes                       No                No
```

```
Contract            Month-to-month           Month-to-month    Month-to-month
PaperlessBilling               Yes                      Yes                No
PaymentMethod     Electronic check  Credit card (automatic)      Mailed check
MonthlyCharges               99.65                     89.1             29.75
TotalCharges                 820.5                   1949.4             301.9
Churn                          Yes                       No                No

                                 8                          9  ...  \
customerID              7892-POOKP                 6388-TABGU  ...
gender                      Female                       Male  ...
SeniorCitizen                    0                          0  ...
Partner                        Yes                         No  ...
Dependents                      No                        Yes  ...
tenure                          28                         62  ...
PhoneService                   Yes                        Yes  ...
MultipleLines                  Yes                         No  ...
InternetService        Fiber optic                        DSL  ...
OnlineSecurity                  No                        Yes  ...
OnlineBackup                    No                        Yes  ...
DeviceProtection               Yes                         No  ...
TechSupport                    Yes                         No  ...
StreamingTV                    Yes                         No  ...
StreamingMovies                Yes                         No  ...
Contract            Month-to-month                   One year  ...
PaperlessBilling               Yes                         No  ...
PaymentMethod     Electronic check  Bank transfer (automatic)  ...
MonthlyCharges               104.8                      56.15  ...
TotalCharges               3046.05                    3487.95  ...
Churn                          Yes                         No  ...

                                        7033                     7034  \
customerID                        9767-FFLEM               0639-TSIQW
gender                                  Male                   Female
SeniorCitizen                              0                        0
Partner                                   No                       No
Dependents                                No                       No
tenure                                    38                       67
PhoneService                             Yes                      Yes
MultipleLines                             No                      Yes
InternetService                  Fiber optic              Fiber optic
OnlineSecurity                            No                      Yes
OnlineBackup                              No                      Yes
DeviceProtection                          No                      Yes
TechSupport                               No                       No
StreamingTV                               No                      Yes
StreamingMovies                           No                       No
Contract                      Month-to-month           Month-to-month
PaperlessBilling                         Yes                      Yes
PaymentMethod        Credit card (automatic)  Credit card (automatic)
MonthlyCharges                          69.5                   102.95
TotalCharges                         2625.25                  6886.25
Churn                                     No                      Yes

                                          7035          7036  \
customerID                          8456-QDAVC    7750-EYXWZ
gender                                    Male        Female
SeniorCitizen                                0             0
Partner                                     No            No
Dependents                                  No            No
```

```
tenure                                   19                12
PhoneService                            Yes                No
MultipleLines                            No  No phone service
InternetService                 Fiber optic               DSL
OnlineSecurity                           No                No
OnlineBackup                             No               Yes
DeviceProtection                         No               Yes
TechSupport                              No               Yes
StreamingTV                             Yes               Yes
StreamingMovies                          No               Yes
Contract                     Month-to-month          One year
PaperlessBilling                        Yes                No
PaymentMethod     Bank transfer (automatic)  Electronic check
MonthlyCharges                         78.7             60.65
TotalCharges                         1495.1             743.3
Churn                                    No                No

                                       7037          7038  \
customerID                       2569-WGERO    6840-RESVB
gender                               Female          Male
SeniorCitizen                             0             0
Partner                                  No           Yes
Dependents                               No           Yes
tenure                                   72            24
PhoneService                            Yes           Yes
MultipleLines                            No           Yes
InternetService                          No           DSL
OnlineSecurity          No internet service           Yes
OnlineBackup            No internet service            No
DeviceProtection        No internet service           Yes
TechSupport             No internet service           Yes
StreamingTV             No internet service           Yes
StreamingMovies         No internet service           Yes
Contract                           Two year      One year
PaperlessBilling                        Yes           Yes
PaymentMethod     Bank transfer (automatic)  Mailed check
MonthlyCharges                        21.15          84.8
TotalCharges                         1419.4        1990.5
Churn                                    No            No

                                     7039              7040            7041  \
customerID                     2234-XADUH        4801-JZAZL      8361-LTMKD
gender                             Female            Female            Male
SeniorCitizen                           0                 0               1
Partner                               Yes               Yes             Yes
Dependents                            Yes               Yes              No
tenure                                 72                11               4
PhoneService                          Yes                No             Yes
MultipleLines                         Yes  No phone service             Yes
InternetService               Fiber optic               DSL     Fiber optic
OnlineSecurity                         No               Yes              No
OnlineBackup                          Yes                No              No
DeviceProtection                      Yes                No              No
TechSupport                            No                No              No
StreamingTV                           Yes                No              No
StreamingMovies                       Yes                No              No
Contract                         One year    Month-to-month  Month-to-month
PaperlessBilling                      Yes               Yes             Yes
PaymentMethod     Credit card (automatic)  Electronic check    Mailed check
```

```
MonthlyCharges                      103.2              29.6            74.4
TotalCharges                       7362.9            346.45           306.6
Churn                                  No                No             Yes

                                       7042
customerID                       3186-AJIEK
gender                                 Male
SeniorCitizen                             0
Partner                                  No
Dependents                               No
tenure                                   66
PhoneService                            Yes
MultipleLines                            No
InternetService                 Fiber optic
OnlineSecurity                          Yes
OnlineBackup                             No
DeviceProtection                        Yes
TechSupport                             Yes
StreamingTV                             Yes
StreamingMovies                         Yes
Contract                           Two year
PaperlessBilling                        Yes
PaymentMethod     Bank transfer (automatic)
MonthlyCharges                       105.65
TotalCharges                         6844.5
Churn                                    No

[21 rows x 7043 columns]
```

In [38]:

```
# Create a summary table and transpose it
summary_by_gender = df.groupby('gender').agg({
    'customerID': 'count',
    'MonthlyCharges': 'mean',
    'TotalCharges': 'sum',
}).round(2)

print("Summary by Gender:")
print(summary_by_gender)
print("\n" + "="*60)

# Transpose the table
transposed = summary_by_gender.T
print("\nTransposed Summary (Gender as columns):")
print(transposed)
```

```
Summary by Gender:
        customerID  MonthlyCharges  TotalCharges
gender
Female        3488           65.20     7952354.2
Male          3555           64.33     8103814.5

============================================================

Transposed Summary (Gender as columns):
gender             Female        Male
customerID         3488.0     3555.00
MonthlyCharges       65.2       64.33
TotalCharges    7952354.2  8103814.50
```

In [7]:

```
print(df.tenure.value_counts())
```

```
tenure
1     613
72    362
2     238
3     200
4     176
     ... 
28     57
39     56
44     51
36     50
0      11
Name: count, Length: 73, dtype: int64
```

In [6]:

```
# Basic value_counts
print("Value Counts - Contract Type:")
contract_counts = df['Contract'].value_counts()
print(contract_counts)
print("\n" + "="*60)

# Value_counts with percentages
print("\nValue Counts with Percentages - Payment Method:")
payment_counts = df['PaymentMethod'].value_counts(normalize=True).round(4) * 100
print(payment_counts)
print("\n" + "="*60)

# Value_counts with custom sorting
print("\nValue Counts Sorted by Values - Online Services:")
online_services = ['OnlineSecurity', 'OnlineBackup', 'DeviceProtection',
                   'TechSupport', 'StreamingTV', 'StreamingMovies']

for service in online_services:
    counts = df[service].value_counts()
    print(f"\n{service}:")
    print(counts)

# Value_counts for multiple columns
print("\n" + "="*60)
print("\nValue Counts for Multiple Columns (first 3):")
multi_counts = {}
for col in ['Partner', 'Dependents', 'PhoneService']:
    multi_counts[col] = df[col].value_counts()

for key, value in multi_counts.items():
    print(f"\n{key}:")
    print(value)
```

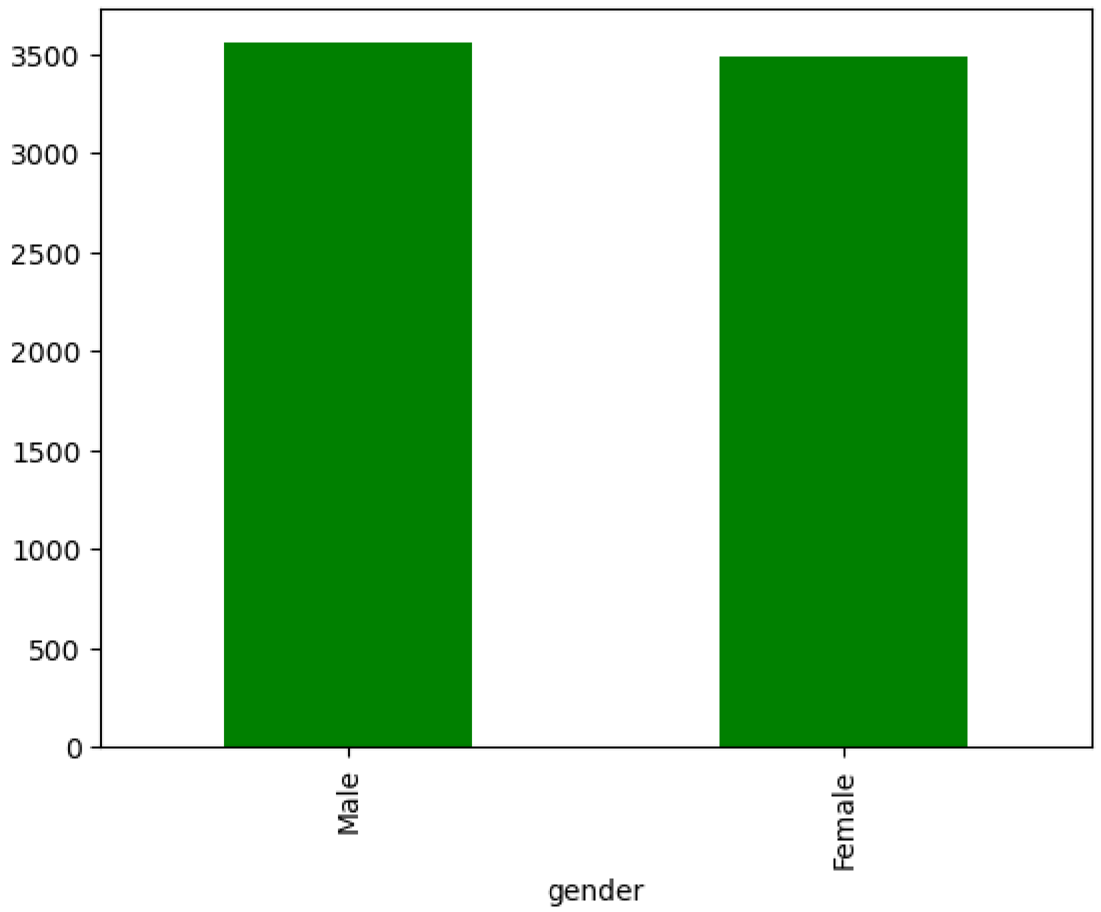
```
Value Counts - Contract Type:
Contract
Month-to-month    3875
Two year          1695
One year          1473
Name: count, dtype: int64

============================================================
Value Counts with Percentages - Payment Method:
PaymentMethod
Electronic check             33.58
Mailed check                 22.89
Bank transfer (automatic)    21.92
Credit card (automatic)      21.61
Name: proportion, dtype: float64

============================================================

Value Counts Sorted by Values - Online Services:

OnlineSecurity:
OnlineSecurity
No                     3498
Yes                    2019
No internet service    1526
Name: count, dtype: int64

OnlineBackup:
OnlineBackup
No                     3088
Yes                    2429
No internet service    1526
Name: count, dtype: int64

DeviceProtection:
DeviceProtection
No                     3095
Yes                    2422
No internet service    1526
Name: count, dtype: int64

TechSupport:
TechSupport
No                     3473
Yes                    2044
No internet service    1526
Name: count, dtype: int64

StreamingTV:
StreamingTV
No                     2810
Yes                    2707
No internet service    1526
Name: count, dtype: int64

StreamingMovies:
StreamingMovies
No                     2785
Yes                    2732
No internet service    1526
Name: count, dtype: int64

============================================================

Value Counts for Multiple Columns (first 3):
```

```
Partner:
Partner
No     3641
Yes    3402
Name: count, dtype: int64

Dependents:
Dependents
No     4933
Yes    2110
Name: count, dtype: int64

PhoneService:
PhoneService
Yes    6361
No      682
Name: count, dtype: int64
```

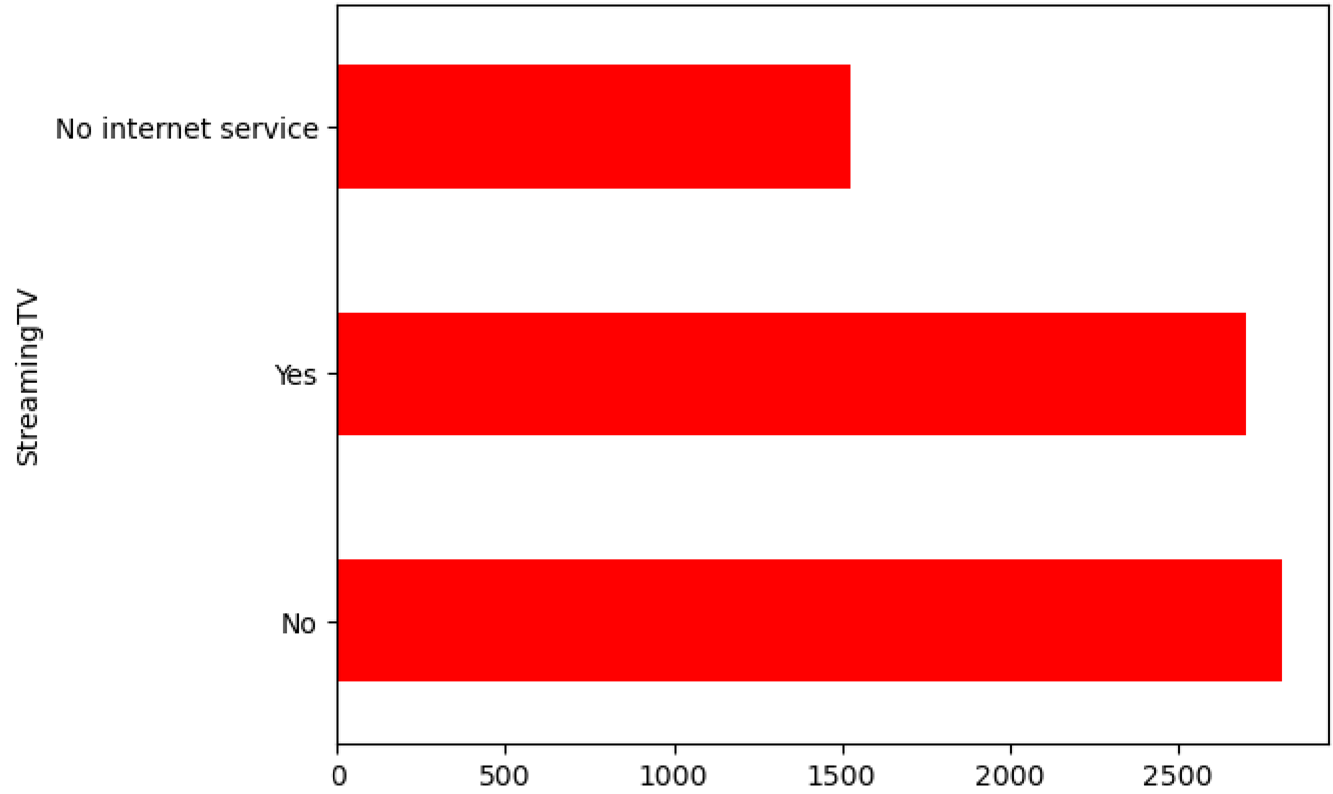
In [9]:

```
print("\n value count by gender:")
c=df['gender'].value_counts()
print(c)
c.plot(kind='bar',color='g')
plt.show()
print("\n" + "="*60)
print("\n value count by streaming tv:")
d=df['StreamingTV'].value_counts()
print(d)
d.plot(kind='barh',color='r')
plt.show()
print("\n" + "="*60)
print("\n value count by phoneservie :")
k=df['PhoneService'].value_counts()
print(k)

df['PhoneService'].value_counts().plot(kind='pie',autopct='%1.1f%%',colors='myr',shadow=
```

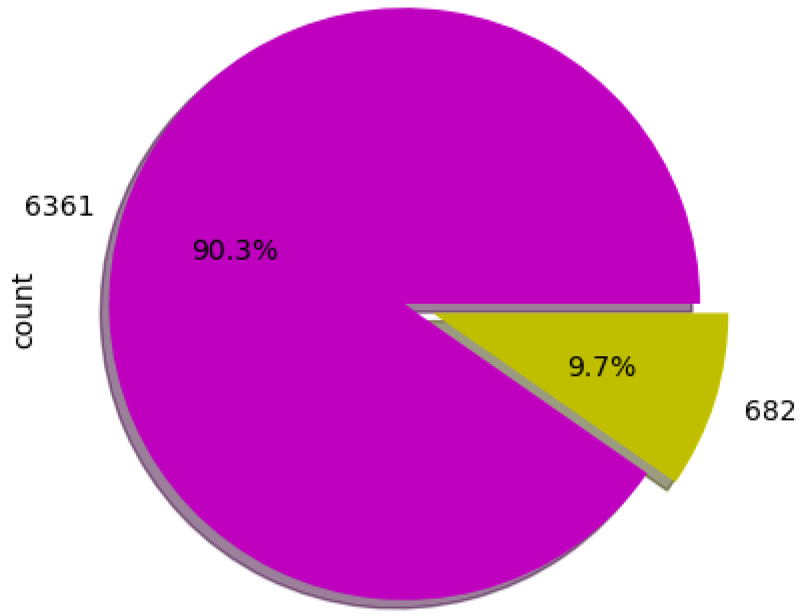
```
 value count by gender:
gender
Male      3555
Female    3488
Name: count, dtype: int64
```



```
============================================================

 value count by streaming tv:
StreamingTV
No                     2810
Yes                    2707
No internet service    1526
Name: count, dtype: int64
```



```
============================================================

 value count by phoneservie :
PhoneService
Yes    6361
No      682
Name: count, dtype: int64
Out[9]:
<Axes: ylabel='count'>
```



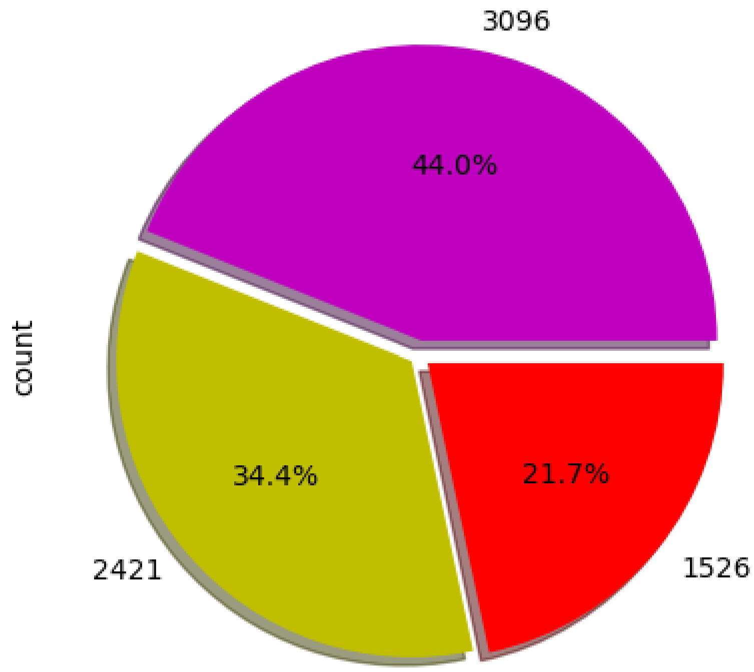
```
In [12]:
```

```
print("\n unique values by internetservice :\n")
print(df['InternetService'].unique())
print("\n" + "="*60)
print("\n unique values by internetservice and its count:\n")
print(df['InternetService'].nunique())
print("\n" + "="*60)
print("\n value count by internetservice:\n")
print(df['InternetService'].value_counts())
df['InternetService'].value_counts().plot(kind='pie',autopct='%1.1f%%',colors='myr',shad
```

```
 unique values by internetservice :

['DSL' 'Fiber optic' 'No']

============================================================

 unique values by internetservice and its count:

3

============================================================

 value count by internetservice:

InternetService
Fiber optic    3096
DSL            2421
No             1526
Name: count, dtype: int64
```

Out[12]:

```
<Axes: ylabel='count'>
```



In [23]:

```
# Get unique values
print("Unique Payment Methods:")
```

```
unique_payments = df['PaymentMethod'].unique()
print(unique_payments)
print("\n" + "="*60)

# Number of unique values
print("\nNumber of unique values in each categorical column:")
categorical_cols=df.select_dtypes(include=['object','category']).columns.tolist()
for col in categorical_cols:
    unique_count = df[col].nunique()
    print(f"{col}: {unique_count} unique values")
print("\n" + "="*60)

# Unique combinations
print("\nUnique Combinations of Contract and PaperlessBilling:")
unique_combinations = df[['Contract', 'PaperlessBilling']].drop_duplicates()
print(unique_combinations.sort_values(['Contract', 'PaperlessBilling']))
```

```
Unique Payment Methods:
['Electronic check' 'Mailed check' 'Bank transfer (automatic)'
 'Credit card (automatic)']

============================================================

Number of unique values in each categorical column:
customerID: 7043 unique values
gender: 2 unique values
Partner: 2 unique values
Dependents: 2 unique values
PhoneService: 2 unique values
MultipleLines: 3 unique values
InternetService: 3 unique values
OnlineSecurity: 3 unique values
OnlineBackup: 3 unique values
DeviceProtection: 3 unique values
TechSupport: 3 unique values
StreamingTV: 3 unique values
StreamingMovies: 3 unique values
Contract: 3 unique values
PaperlessBilling: 2 unique values
PaymentMethod: 4 unique values
Churn: 2 unique values

============================================================

Unique Combinations of Contract and PaperlessBilling:
          Contract PaperlessBilling
7   Month-to-month               No
0   Month-to-month              Yes
1         One year               No
54        One year              Yes
11        Two year               No
23        Two year              Yes
```

In [55]:

```
print(df.sort_values(by='MonthlyCharges',ascending=False))
```

```
      customerID  gender  SeniorCitizen Partner Dependents  tenure  \
4586  7569-NMZYQ  Female              0     Yes        Yes      72
2115  8984-HPEMB  Female              0      No         No      71
3894  5989-AXPUC  Female              0     Yes         No      68
```

```
4804  5734-EJKXG  Female              0      No         No      61
5127  8199-ZLLSA    Male              0      No         No      67
...          ...     ...            ...     ...        ...     ...
6906  9945-PSVIP  Female              0     Yes        Yes      25
1156  0621-CXBKL  Female              0      No         No      53
6652  0827-ITJPH    Male              0      No         No      36
1529  9764-REAFF  Female              0     Yes         No      59
3719  6823-SIDFQ    Male              0      No         No      28

     PhoneService MultipleLines InternetService       OnlineSecurity  ...  \
4586          Yes           Yes     Fiber optic                  Yes  ...
2115          Yes           Yes     Fiber optic                  Yes  ...
3894          Yes           Yes     Fiber optic                  Yes  ...
4804          Yes           Yes     Fiber optic                  Yes  ...
5127          Yes           Yes     Fiber optic                  Yes  ...
...           ...           ...             ...                  ...  ...
6906          Yes            No              No  No internet service  ...
1156          Yes            No              No  No internet service  ...
6652          Yes            No              No  No internet service  ...
1529          Yes            No              No  No internet service  ...
3719          Yes            No              No  No internet service  ...

         DeviceProtection          TechSupport          StreamingTV  \
4586                  Yes                  Yes                  Yes
2115                  Yes                  Yes                  Yes
3894                  Yes                  Yes                  Yes
4804                  Yes                  Yes                  Yes
5127                  Yes                  Yes                  Yes
...                   ...                  ...                  ...
6906  No internet service  No internet service  No internet service
1156  No internet service  No internet service  No internet service
6652  No internet service  No internet service  No internet service
1529  No internet service  No internet service  No internet service
3719  No internet service  No internet service  No internet service

          StreamingMovies  Contract PaperlessBilling  \
4586                  Yes  Two year              Yes
2115                  Yes  Two year              Yes
3894                  Yes  Two year               No
4804                  Yes  One year              Yes
5127                  Yes  One year              Yes
...                   ...       ...              ...
6906  No internet service  Two year              Yes
1156  No internet service  Two year               No
6652  No internet service  Two year              Yes
1529  No internet service  Two year               No
3719  No internet service  One year               No

                  PaymentMethod MonthlyCharges  TotalCharges  Churn
4586  Bank transfer (automatic)         118.75       8672.45     No
2115           Electronic check         118.65       8477.60     No
3894               Mailed check         118.60       7990.05     No
4804           Electronic check         118.60       7365.70     No
5127  Bank transfer (automatic)         118.35       7804.15    Yes
...                         ...            ...           ...    ...
6906               Mailed check          18.70        383.65     No
1156               Mailed check          18.70       1005.70     No
6652    Credit card (automatic)          18.55        689.00     No
1529  Bank transfer (automatic)          18.40       1057.85     No
```

```
3719    Credit card (automatic)         18.25         534.70      No

[7043 rows x 21 columns]
```

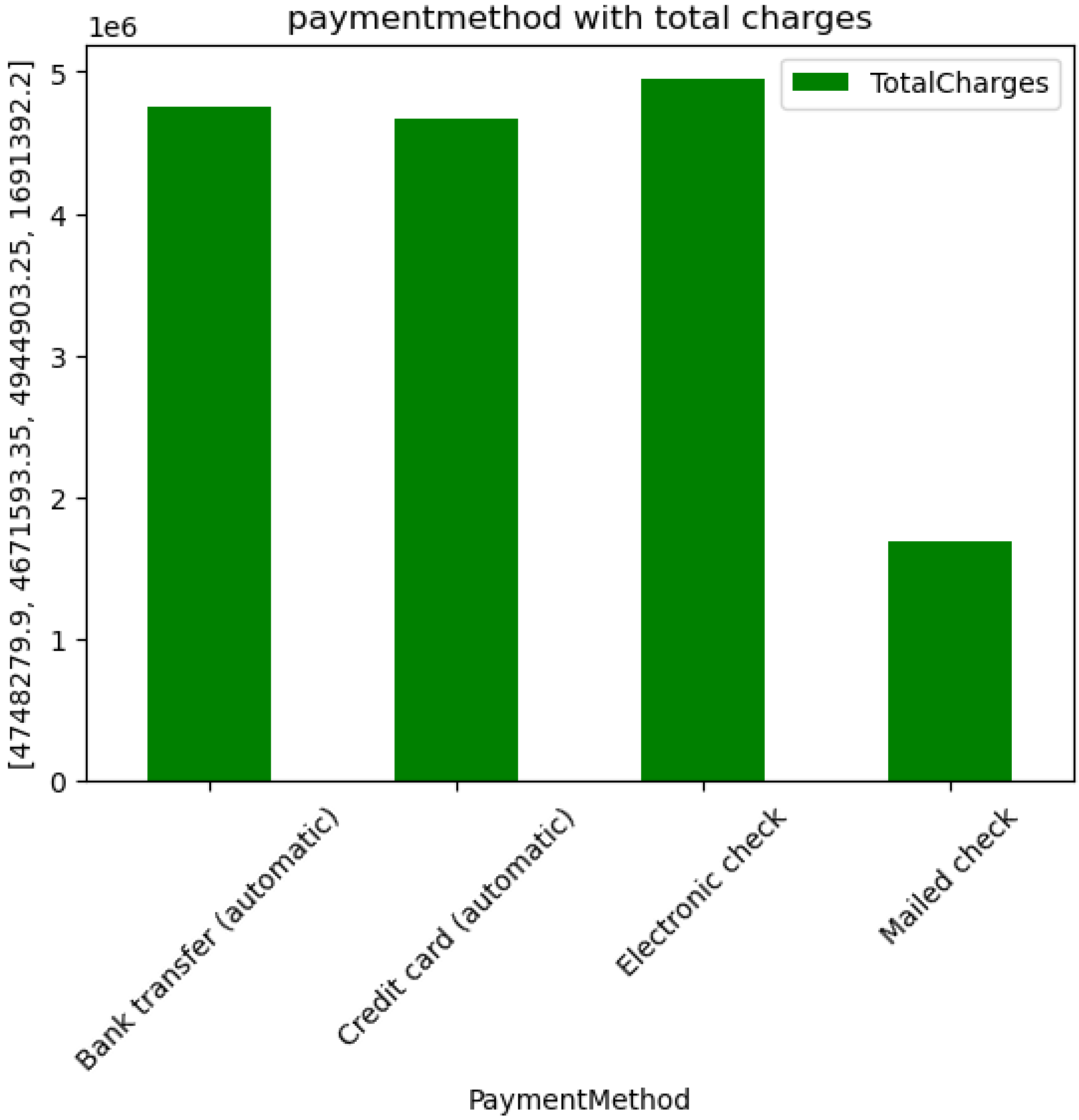
In [9]:

```
print("\n group by paymentmethod with totalcharges :\n")
print(df.groupby('PaymentMethod')['TotalCharges'].sum())
df.groupby('PaymentMethod')['TotalCharges'].sum().plot(kind='bar',rot=45,ylabel=df.group
plt.title('paymentmethod with total charges')
plt.legend()
```

```
 group by paymentmethod with totalcharges :

PaymentMethod
Bank transfer (automatic)    4748279.90
Credit card (automatic)      4671593.35
Electronic check             4944903.25
Mailed check                 1691392.20
Name: TotalCharges, dtype: float64
```

Out[9]:

```
<matplotlib.legend.Legend at 0x1ab97bfe480>
```

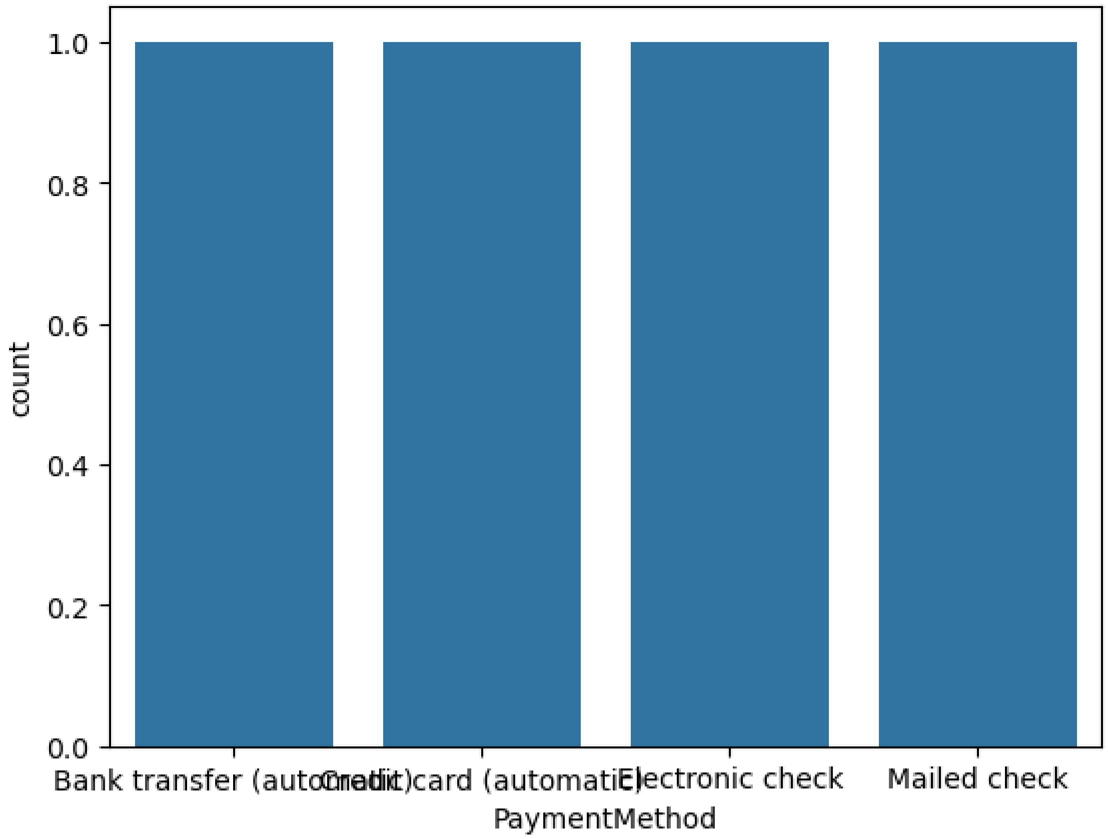


In [14]:

```
import seaborn as sns
sns.countplot(df.groupby('PaymentMethod')['TotalCharges'].sum())
```

Out[14]:

```
<Axes: xlabel='PaymentMethod', ylabel='count'>
```



In [4]:

```
print(df.groupby('PaymentMethod')['TotalCharges'].sum().reset_index())
```

```
               PaymentMethod  TotalCharges
0  Bank transfer (automatic)    4748279.90
1    Credit card (automatic)    4671593.35
2           Electronic check    4944903.25
3               Mailed check    1691392.20
```

In [22]:

```
print("\n group by paymentmethod with totalcharges along with maximum value with name :\
print(df.groupby('PaymentMethod')['TotalCharges'].sum().sort_values(ascending=False).hea
```

```
 group by paymentmethod with totalcharges along with maximum value with name :

PaymentMethod
Electronic check    4944903.25
Name: TotalCharges, dtype: float64
```

In [105]:

```
print(df.groupby(['PaymentMethod','InternetService'])['TotalCharges'].sum())
```

```
PaymentMethod              InternetService
Bank transfer (automatic)  DSL                1655766.90
                           Fiber optic        2783830.65
                           No                  308682.35
Credit card (automatic)    DSL                1730501.05
                           Fiber optic        2647442.45
                           No                  293649.85
Electronic check           DSL                 914329.00
                           Fiber optic        3964264.10
                           No                   66310.15
```

```
Mailed check               DSL                 820813.90
                           Fiber optic         528085.75
                           No                  342492.55
Name: TotalCharges, dtype: float64
```

In [9]:

```python
# Fix 1: Correct way to handle missing values
df['TotalCharges'] = pd.to_numeric(df['TotalCharges'], errors='coerce')
df['TotalCharges'] = df['TotalCharges'].fillna(df['TotalCharges'].median())

# Fix 2: Create binary churn column
df['Churn_binary'] = df['Churn'].apply(lambda x: 1 if x == 'Yes' else 0)

# Multiple aggregation functions with groupby
groupby_results = df.groupby('Churn').agg({
    'tenure': ['mean', 'median', 'min', 'max', 'count'],
    'MonthlyCharges': ['mean', 'median', 'std'],
    'TotalCharges': ['mean', 'sum']
})

print("GroupBy with Multiple Aggregations:")
print(groupby_results)
print("\n" + "="*60)

# Group by multiple columns
multi_group = df.groupby(['InternetService', 'Contract']).agg({
    'customerID': 'count',
    'MonthlyCharges': 'mean',
    'Churn_binary': 'mean'
}).round(2)

multi_group = multi_group.rename(columns={
    'customerID': 'Customer_Count',
    'MonthlyCharges': 'Avg_Monthly_Charge',
    'Churn_binary': 'Churn_Rate'
})

print("\nGroupBy InternetService and Contract:")
print(multi_group)
```

```
GroupBy with Multiple Aggregations:
          tenure                      MonthlyCharges                     \
            mean median min max count           mean  median        std
Churn
No     37.569965   38.0   0  72  5174      61.265124  64.425  31.092648
Yes    17.979133   10.0   1  72  1869      74.441332  79.650  24.666053

      TotalCharges
              mean           sum
Churn
No     2552.882494  1.320861e+07
Yes    1531.796094  2.862927e+06

============================================================

GroupBy InternetService and Contract:
                                Customer_Count  Avg_Monthly_Charge  Churn_Rate
InternetService Contract
DSL             Month-to-month            1223               50.22        0.32
```

```
                One year                   570                 61.40        0.09
                Two year                   628                 70.46        0.02
Fiber optic     Month-to-month            2128                 87.02        0.55
                One year                   539                 98.78        0.19
                Two year                   429                104.57        0.07
No              Month-to-month             524                 20.41        0.19
                One year                   364                 20.82        0.02
                Two year                   638                 21.78        0.01
```

In [24]:

```
# Create a pivot table first, then transpose
pivot_complex = pd.pivot_table(
    df,
    values='MonthlyCharges',
    index=['Contract', 'PaperlessBilling'],
    columns='Churn',
    aggfunc=['sum', 'count']
)

print("\nComplex Pivot Table:")
print(pivot_complex)
print("\n" + "="*60)

print("\nTransposed Complex Pivot Table:")
transposed_complex = pivot_complex.T
print(transposed_complex)
```

```
Complex Pivot Table:
                                      sum           count     
Churn                                  No       Yes    No  Yes
Contract       PaperlessBilling                               
Month-to-month No                44372.70  24910.65   883  406
               Yes               92074.35  95936.45  1337 1249
One year       No                31492.70   3673.30   625   48
               Yes               50205.45  10445.15   682  118
Two year       No                43837.55   1036.90   895   15
               Yes               55003.00   3128.40   752   33

============================================================

Transposed Complex Pivot Table:
Contract         Month-to-month           One year            Two year  \
PaperlessBilling             No       Yes       No       Yes        No   
      Churn                                                             
sum   No               44372.70  92074.35  31492.7  50205.45  43837.55   
      Yes              24910.65  95936.45   3673.3  10445.15   1036.90   
count No                 883.00   1337.00    625.0    682.00    895.00   
      Yes                406.00   1249.00     48.0    118.00     15.00   

Contract                
PaperlessBilling      Yes
      Churn              
sum   No          55003.0
      Yes          3128.4
count No            752.0
      Yes            33.0
```

In [17]:

```
# Select a subset of columns for melt/unpivot demonstration
melt_df = df[['customerID', 'InternetService', 'Contract', 'PaymentMethod', 'MonthlyChar
```

```
print("Original Data (before melt):")
print(melt_df)
print("\n" + "="*60)

# Melt/Unpivot the data
melted = pd.melt(
    melt_df,
    id_vars=['customerID', 'MonthlyCharges'],
    value_vars=['InternetService', 'Contract', 'PaymentMethod'],
    var_name='Service_Category',
    value_name='Service_Value'
)

print("\nMelted/Unpivoted Data:")
print(melted)
print("\n" + "="*60)

# More practical melt example
# Create a summary table first, then melt it
summary_df = df.groupby(['InternetService', 'Churn']).size().reset_index(name='Count')

pivot_summary = summary_df.pivot(index='InternetService', columns='Churn', values='Count
print("\nPivot Summary (before melt):")
print(pivot_summary)

# Melt the pivot table
melted_summary = pivot_summary.reset_index().melt(
    id_vars='InternetService',
    value_vars=['No', 'Yes'],
    var_name='Churn',
    value_name='Customer_Count'
)

print("\nMelted Summary:")
print(melted_summary.sort_values(['InternetService', 'Churn']))
```

```
Original Data (before melt):
   customerID InternetService        Contract             PaymentMethod  \
0  7590-VHVEG             DSL  Month-to-month          Electronic check   
1  5575-GNVDE             DSL        One year              Mailed check   
2  3668-QPYBK             DSL  Month-to-month              Mailed check   
3  7795-CFOCW             DSL        One year  Bank transfer (automatic)   
4  9237-HQITU     Fiber optic  Month-to-month          Electronic check   

   MonthlyCharges  
0           29.85  
1           56.95  
2           53.85  
3           42.30  
4           70.70  

============================================================

Melted/Unpivoted Data:
    customerID  MonthlyCharges Service_Category              Service_Value
0   7590-VHVEG           29.85  InternetService                        DSL
1   5575-GNVDE           56.95  InternetService                        DSL
2   3668-QPYBK           53.85  InternetService                        DSL
```

```
3   7795-CFOCW           42.30  InternetService                        DSL
4   9237-HQITU           70.70  InternetService                Fiber optic
5   7590-VHVEG           29.85         Contract             Month-to-month
6   5575-GNVDE           56.95         Contract                   One year
7   3668-QPYBK           53.85         Contract             Month-to-month
8   7795-CFOCW           42.30         Contract                   One year
9   9237-HQITU           70.70         Contract             Month-to-month
10  7590-VHVEG           29.85    PaymentMethod           Electronic check
11  5575-GNVDE           56.95    PaymentMethod               Mailed check
12  3668-QPYBK           53.85    PaymentMethod               Mailed check
13  7795-CFOCW           42.30    PaymentMethod  Bank transfer (automatic)
14  9237-HQITU           70.70    PaymentMethod           Electronic check

============================================================

Pivot Summary (before melt):
Churn              No   Yes
InternetService            
DSL              1962   459
Fiber optic      1799  1297
No               1413   113

Melted Summary:
  InternetService Churn  Customer_Count
0             DSL    No            1962
3             DSL   Yes             459
1     Fiber optic    No            1799
4     Fiber optic   Yes            1297
2              No    No            1413
5              No   Yes             113
```

In [18]:

```python
# Basic crosstab
ct_basic = pd.crosstab(df['InternetService'], df['Churn'])
print("Crosstab - InternetService vs Churn:")
print(ct_basic)
print("\n" + "="*60)

# Crosstab with margins and percentages
ct_margins = pd.crosstab(
    df['InternetService'],
    df['Churn'],
    margins=True,
    margins_name='Total',
    normalize='index'  # Row percentages
).round(4) * 100

print("\nCrosstab with Percentages (Row %):")
print(ct_margins)
print("\n" + "="*60)

# Multi-dimensional crosstab
ct_multi = pd.crosstab(
    [df['Contract'], df['PaperlessBilling']],
    [df['Churn'], df['SeniorCitizen']],
    margins=True
)

print("\nMulti-dimensional Crosstab:")
print(ct_multi)
print("\n" + "="*60)

# Crosstab with aggregation
ct_agg = pd.crosstab(
    df['Contract'],
    df['Churn'],
    values=df['MonthlyCharges'],
    aggfunc='mean'
).round(2)

print("\nCrosstab with Average Monthly Charges:")
print(ct_agg)
```

```
Crosstab - InternetService vs Churn:
Churn              No   Yes
InternetService            
DSL              1962   459
Fiber optic      1799  1297
No               1413   113

============================================================

Crosstab with Percentages (Row %):
Churn               No    Yes
InternetService              
DSL              81.04  18.96
Fiber optic      58.11  41.89
No               92.60   7.40
Total            73.46  26.54

============================================================

Multi-dimensional Crosstab:
Churn                              No        Yes         All
SeniorCitizen                       0    1     0    1       
Contract       PaperlessBilling                             
Month-to-month No                 795   88   335   71  1289
               Yes               1059  278   879  370  2586
One year       No                 573   52    43    5   673
               Yes                573  109    94   24   800
Two year       No                 847   48    13    2   910
               Yes                661   91    29    4   785
All                              4508  666  1393  476  7043

============================================================

Crosstab with Average Monthly Charges:
Churn              No    Yes
Contract                    
Month-to-month  61.46  73.02
One year        62.51  85.05
Two year        60.01  86.78
```

In [39]:

```
# Check for duplicate rows based on specific columns
print("\nDuplicate Rows based on key service columns:")
key_columns = ['InternetService', 'Contract', 'PaymentMethod', 'MonthlyCharges']
duplicate_rows = df[df.duplicated(subset=key_columns, keep=False)]
```

```
print(f"Found {len(duplicate_rows)} rows with duplicate service combinations")
print(f"That's {len(duplicate_rows)/len(df)*100:.2f}% of total data")
print("\n" + "="*60)

# Find and display some duplicate examples
if len(duplicate_rows) > 0:
    print("\nSample Duplicate Rows:")
    sample_duplicates = duplicate_rows.sort_values(key_columns).head(10)
    print(sample_duplicates[['customerID'] + key_columns])

# Remove duplicates (creating a new dataframe for demonstration)
df_no_duplicates = df.drop_duplicates(subset=key_columns)
print(f"\nOriginal shape: {df.shape}")
print(f"After removing service duplicates: {df_no_duplicates.shape}")
print(f"Rows removed: {len(df) - len(df_no_duplicates)}")
```

```
Duplicate Rows based on key service columns:
Found 3634 rows with duplicate service combinations
That's 51.60% of total data

============================================================

Sample Duplicate Rows:
      customerID InternetService        Contract              PaymentMethod  \
4024  1329-VHWNP             DSL  Month-to-month  Bank transfer (automatic)
5665  6345-HOVES             DSL  Month-to-month  Bank transfer (automatic)
3940  1559-DTODC             DSL  Month-to-month  Bank transfer (automatic)
6204  3058-WQDRE             DSL  Month-to-month  Bank transfer (automatic)
667   5533-RJFTJ             DSL  Month-to-month  Bank transfer (automatic)
4897  2533-TIBIX             DSL  Month-to-month  Bank transfer (automatic)
5329  2894-QOJRX             DSL  Month-to-month  Bank transfer (automatic)
6522  7233-IOQNP             DSL  Month-to-month  Bank transfer (automatic)
4961  6954-OOYZZ             DSL  Month-to-month  Bank transfer (automatic)
5463  6142-VSJQO             DSL  Month-to-month  Bank transfer (automatic)

      MonthlyCharges
4024           25.05
5665           25.05
3940           25.15
6204           25.15
667            30.20
4897           30.20
5329           34.00
6522           34.00
4961           44.35
5463           44.35

Original shape: (7043, 22)
After removing service duplicates: (4666, 22)
Rows removed: 2377
```

In [25]:

```
# Basic sorting
print("Top 10 Highest Monthly Charges:")
top_charges = df.sort_values('MonthlyCharges', ascending=False).head(10)
print(top_charges[['customerID', 'MonthlyCharges', 'Contract', 'InternetService']])
print("\n" + "="*60)

# Sorting by multiple columns
```

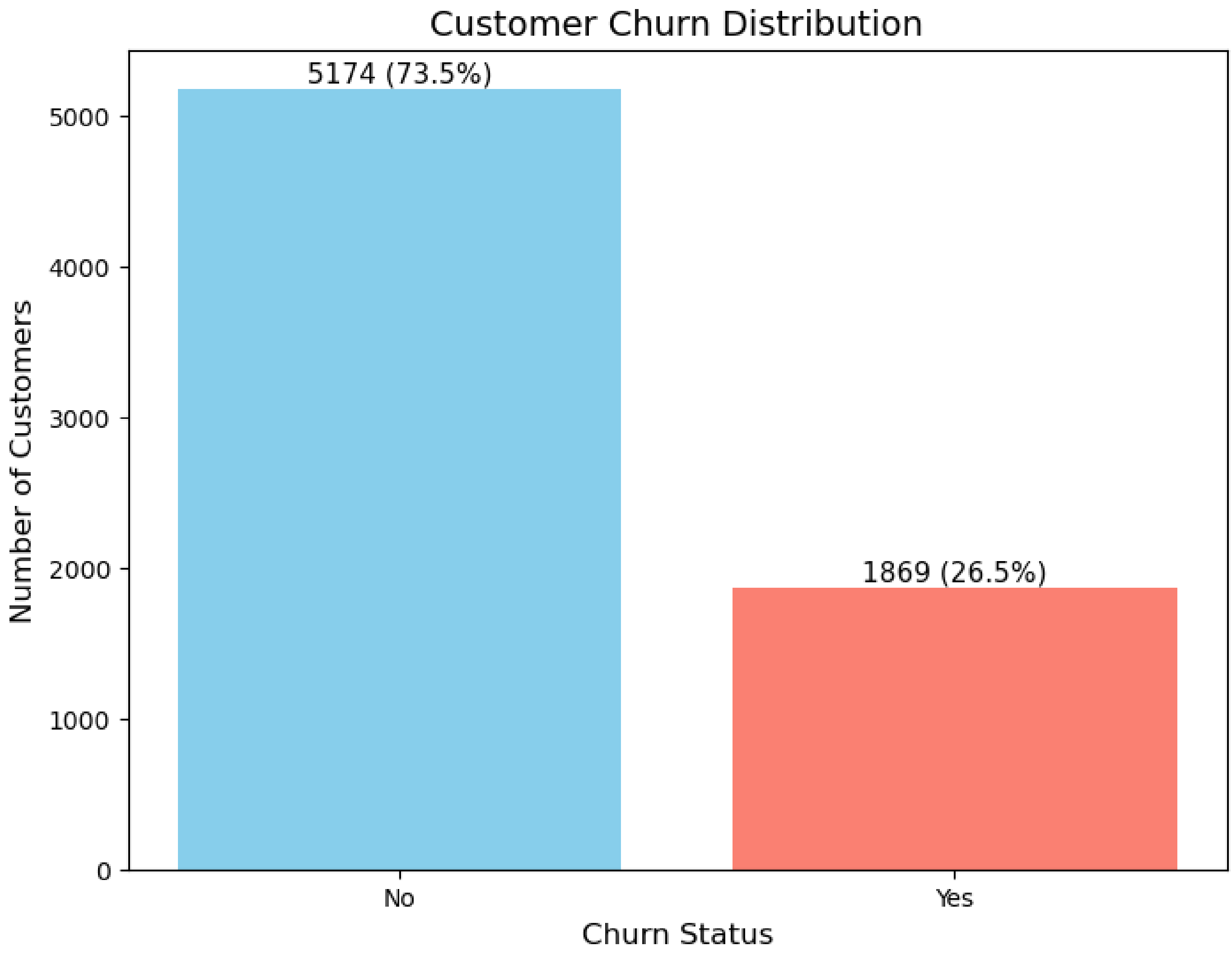
```python
print("\nCustomers sorted by Tenure (desc) and Monthly Charges (desc):")
multi_sort = df.sort_values(['tenure', 'MonthlyCharges'], ascending=[False, False]).head
print(multi_sort[['customerID', 'tenure', 'MonthlyCharges', 'Churn']])
print("\n" + "="*60)

# Sorting with groupby results
group_sorted = df.groupby('InternetService').agg({
    'MonthlyCharges': 'mean',
    'Churn_binary': 'mean',
    'customerID': 'count'
}).round(2)

group_sorted = group_sorted.rename(columns={
    'MonthlyCharges': 'Avg_Monthly_Charge',
    'Churn_binary': 'Churn_Rate',
    'customerID': 'Customer_Count'
})

print("\nInternet Service Analysis (sorted by Churn Rate):")
sorted_by_churn = group_sorted.sort_values('Churn_Rate', ascending=False)
print(sorted_by_churn)
print("\n" + "="*60)

# Sorting with custom order (using categorical)
contract_order = ['Month-to-month', 'One year', 'Two year']
df['Contract_ordered'] = pd.Categorical(df['Contract'], categories=contract_order, order

print("\nData sorted by Custom Contract Order:")
contract_sorted = df.sort_values('Contract_ordered').head(10)
print(contract_sorted[['customerID', 'Contract', 'MonthlyCharges']])
```

```
Top 10 Highest Monthly Charges:
      customerID  MonthlyCharges        Contract InternetService
4586  7569-NMZYQ          118.75        Two year     Fiber optic
2115  8984-HPEMB          118.65        Two year     Fiber optic
3894  5989-AXPUC          118.60        Two year     Fiber optic
4804  5734-EJKXG          118.60        One year     Fiber optic
5127  8199-ZLLSA          118.35        One year     Fiber optic
6118  9924-JPRMC          118.20        Two year     Fiber optic
4610  2889-FPWRM          117.80        One year     Fiber optic
3205  3810-DVDQQ          117.60        Two year     Fiber optic
6768  9739-JLPQJ          117.50        Two year     Fiber optic
4875  2302-ANTDP          117.45  Month-to-month     Fiber optic

============================================================

Customers sorted by Tenure (desc) and Monthly Charges (desc):
      customerID  tenure  MonthlyCharges Churn
4586  7569-NMZYQ      72          118.75    No
6118  9924-JPRMC      72          118.20    No
4610  2889-FPWRM      72          117.80   Yes
3205  3810-DVDQQ      72          117.60    No
6768  9739-JLPQJ      72          117.50    No
4155  6904-JLBGY      72          117.35    No
2368  6650-BWFRT      72          117.15    No
5347  9788-HNGUT      72          116.95    No
2025  1488-PBLJN      72          116.85    No
4206  0017-IUDMW      72          116.80    No
```

```
============================================================

Internet Service Analysis (sorted by Churn Rate):
                 Avg_Monthly_Charge  Churn_Rate  Customer_Count
InternetService
Fiber optic                   91.50        0.42            3096
DSL                           58.10        0.19            2421
No                            21.08        0.07            1526

============================================================

Data sorted by Custom Contract Order:
      customerID        Contract  MonthlyCharges
0     7590-VHVEG  Month-to-month           29.85
5322  8731-WBBMB  Month-to-month           81.90
5317  1213-NGCUN  Month-to-month           49.65
5315  2082-CEFLT  Month-to-month           45.60
2856  2740-TVLFN  Month-to-month           50.15
5313  1935-IMVBB  Month-to-month           89.70
2858  9512-PHSMG  Month-to-month           20.55
5312  9564-KCLHR  Month-to-month           51.25
2860  2452-KDRRH  Month-to-month          101.40
2861  2004-OCQXK  Month-to-month           81.95
```

In [38]:

```python
# Calculate overall churn rate
churn_counts = df['Churn'].value_counts()
churn_rate = (churn_counts['Yes'] / len(df)) * 100

plt.figure(figsize=(8, 6))
plt.bar(churn_counts.index, churn_counts.values, color=['skyblue', 'salmon'])
plt.title('Customer Churn Distribution', fontsize=14)
plt.xlabel('Churn Status', fontsize=12)
plt.ylabel('Number of Customers', fontsize=12)
plt.text(0, churn_counts['No'] + 50, f"{churn_counts['No']} ({100-churn_rate:.1f}%)",
         ha='center', fontsize=11)
plt.text(1, churn_counts['Yes'] + 50, f"{churn_counts['Yes']} ({churn_rate:.1f}%)",
         ha='center', fontsize=11)
plt.show()

print(f"Overall Churn Rate: {churn_rate:.2f}%")
```

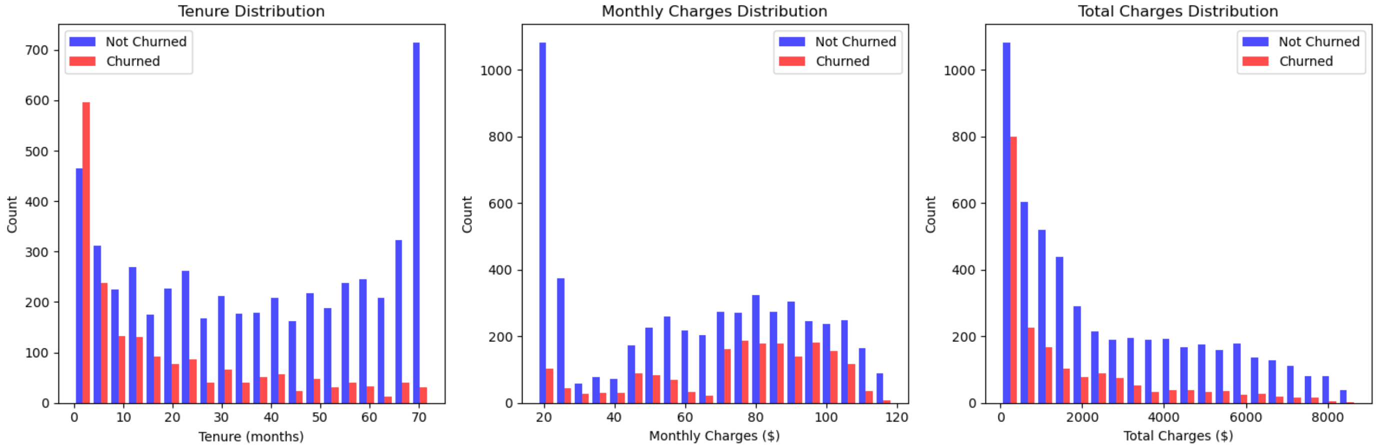


```
Overall Churn Rate: 26.54%
```

In [8]:

```python
# Compare numerical features for churned vs non-churned customers
churned = df[df['Churn'] == 'Yes']
not_churned = df[df['Churn'] == 'No']

fig, axes = plt.subplots(1, 3, figsize=(15, 5))

# Tenure comparison
axes[0].hist([not_churned['tenure'], churned['tenure']],
             bins=20, label=['Not Churned', 'Churned'],
             alpha=0.7, color=['blue', 'red'])
axes[0].set_title('Tenure Distribution')
axes[0].set_xlabel('Tenure (months)')
axes[0].set_ylabel('Count')
axes[0].legend()

# Monthly Charges comparison
axes[1].hist([not_churned['MonthlyCharges'], churned['MonthlyCharges']],
             bins=20, label=['Not Churned', 'Churned'],
             alpha=0.7, color=['blue', 'red'])
axes[1].set_title('Monthly Charges Distribution')
axes[1].set_xlabel('Monthly Charges ($)')
axes[1].set_ylabel('Count')
axes[1].legend()
```

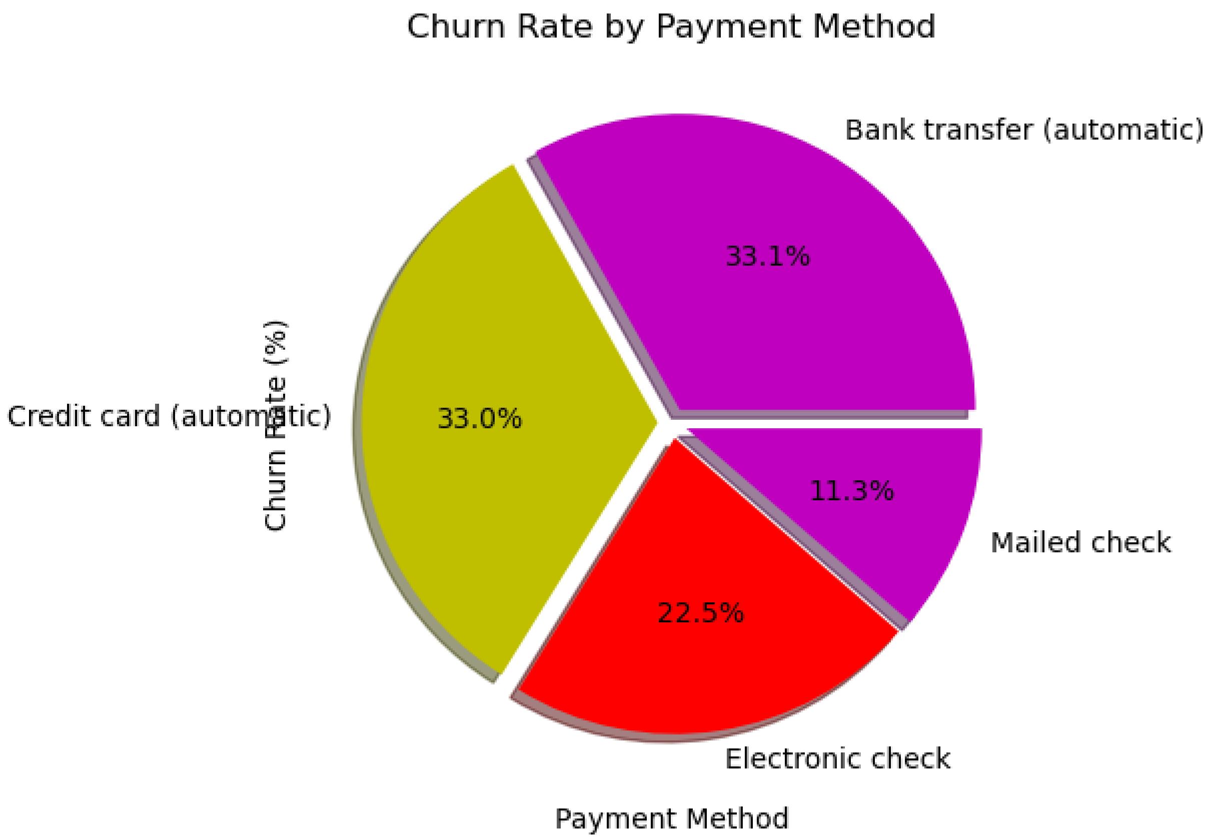
```
# Total Charges comparison
axes[2].hist([not_churned['TotalCharges'], churned['TotalCharges']],
             bins=20, label=['Not Churned', 'Churned'],
             alpha=0.7, color=['blue', 'red'])
axes[2].set_title('Total Charges Distribution')
axes[2].set_xlabel('Total Charges ($)')
axes[2].set_ylabel('Count')
axes[2].legend()

plt.tight_layout()
plt.show()

# Calculate average values
print("\nAverage Values Comparison:")
print(f"Average Tenure - Churned: {churned['tenure'].mean():.1f} months")
print(f"Average Tenure - Not Churned: {not_churned['tenure'].mean():.1f} months")
print(f"Average Monthly Charges - Churned: ${churned['MonthlyCharges'].mean():.2f}")
print(f"Average Monthly Charges - Not Churned: ${not_churned['MonthlyCharges'].mean():.2
```



```
Average Values Comparison:
Average Tenure - Churned: 18.0 months
Average Tenure - Not Churned: 37.6 months
Average Monthly Charges - Churned: $74.44
Average Monthly Charges - Not Churned: $61.27
```

In [14]:

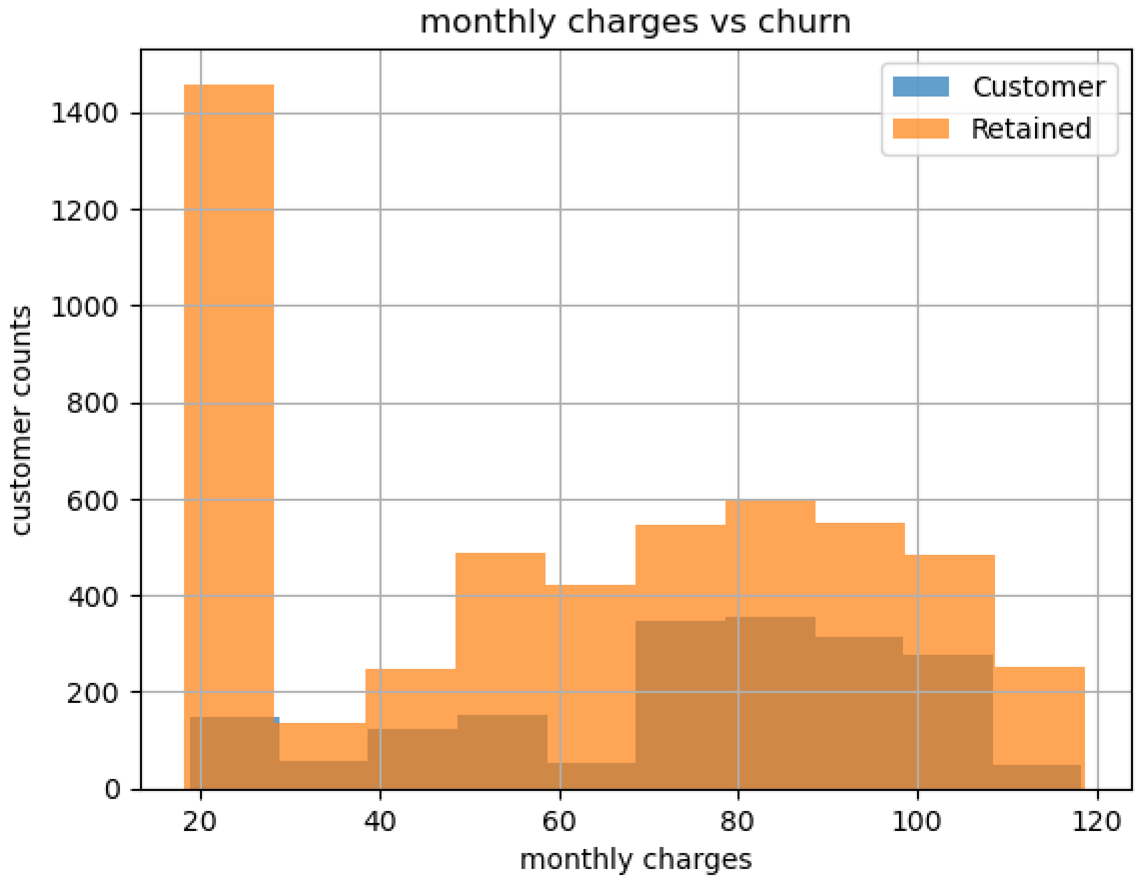
```
# Churn by Payment Method
payment_churn = df.groupby('PaymentMethod')['TotalCharges'].mean() * 100
plt.figure(figsize=(6, 6))
payment_churn.sort_values(ascending=False).plot(kind='pie',autopct='%1.1f%%',colors='myr
plt.title('Churn Rate by Payment Method')
plt.xlabel('Payment Method')
plt.ylabel('Churn Rate (%)')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



does monthly charge impact churn

In [37]:

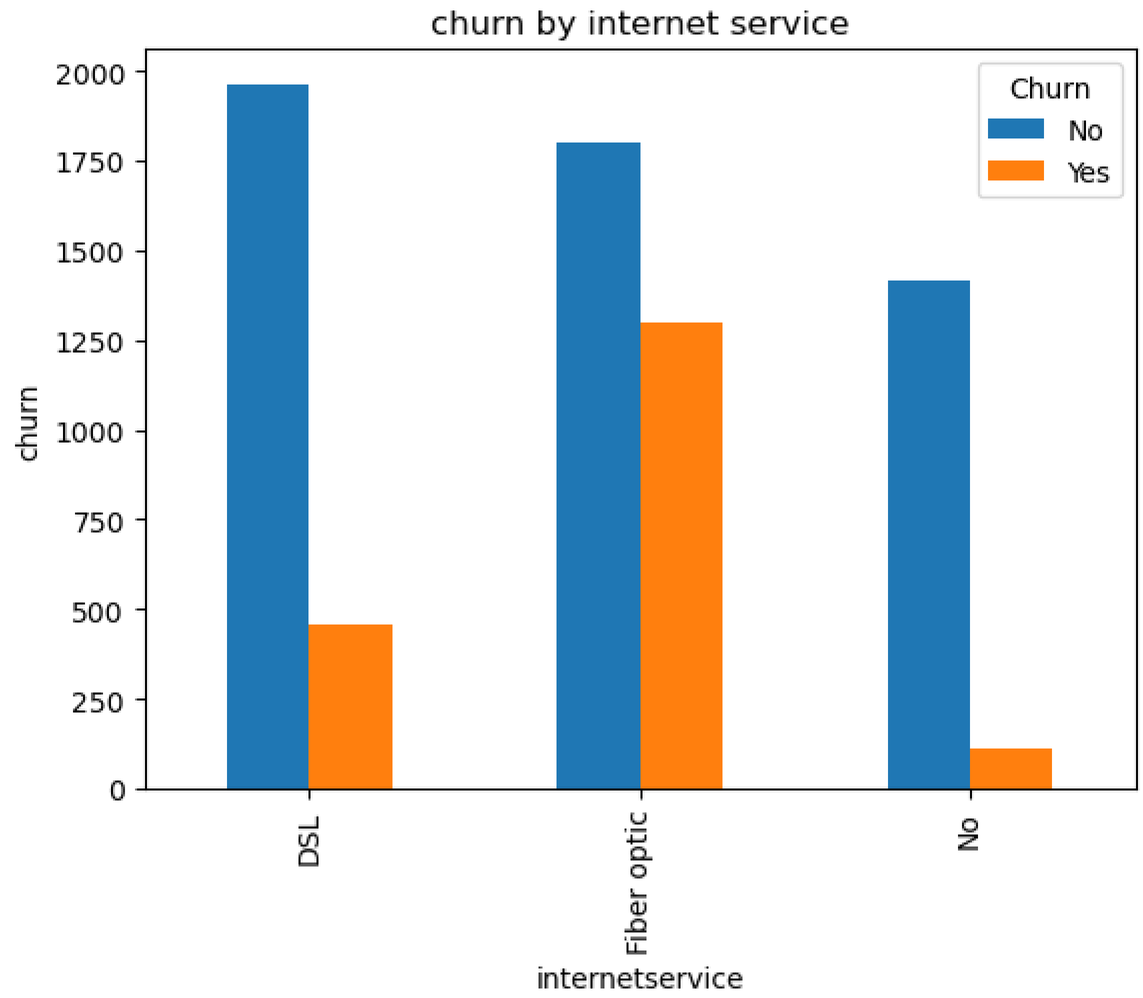
```
plt.figure()
df[df['Churn']=='Yes']['MonthlyCharges'].hist(alpha=0.7)
df[df['Churn']=='No']['MonthlyCharges'].hist(alpha=0.7)
plt.legend(['Customer','Retained'])
plt.xlabel('monthly charges')
plt.ylabel('customer counts')
plt.title('monthly charges vs churn')
plt.show()
```



which internet service has highest churn

In [52]:

```
internet_churn=df.groupby('InternetService')['Churn'].value_counts().unstack()
internet_churn.plot(kind='bar')
plt.title('churn by internet service')
plt.xlabel('internetservice')
plt.ylabel('churn')
plt.show()
```



In [10]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

# Load and prepare data
df = pd.read_csv("C:\\Users\\prems\\Videos\\power bi\\Telco_Cusomer_Churn.csv")

# Data preparation
df['TotalCharges'] = pd.to_numeric(df['TotalCharges'], errors='coerce')
df['TotalCharges'] = df['TotalCharges'].fillna(df['TotalCharges'].median())
df['Churn_binary'] = df['Churn'].apply(lambda x: 1 if x == 'Yes' else 0)
churned = df[df['Churn'] == 'Yes']
not_churned = df[df['Churn'] == 'No']

# Create figure with all visualizations
plt.figure(figsize=(22, 30))

# ==================== 1. CHURN RATE BY INTERNET SERVICE ====================
plt.subplot(5, 4, 1)
churn_by_internet = df.groupby('InternetService')['Churn_binary'].mean() * 100
bars1 = plt.bar(churn_by_internet.index, churn_by_internet.values,
               color=['#FF9999', '#66B2FF', '#99FF99'])
plt.title('1. Churn Rate by Internet Service', fontsize=12, fontweight='bold')
plt.xlabel('Internet Service')
plt.ylabel('Churn Rate (%)')
plt.ylim(0, 50)
plt.grid(True, alpha=0.3)
```

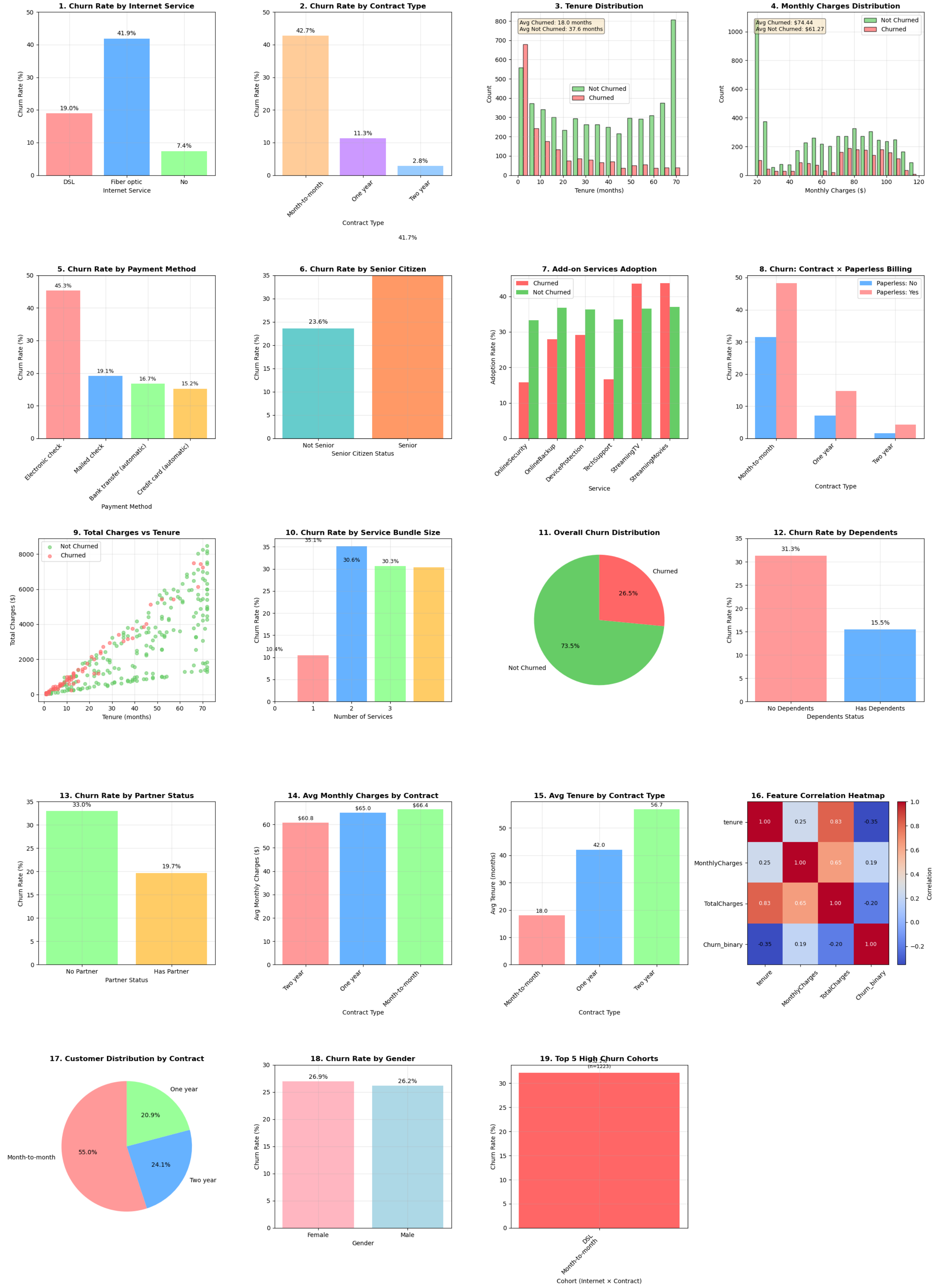
```
for i, v in enumerate(churn_by_internet.values):
    plt.text(i, v + 1, f'{v:.1f}%', ha='center', fontsize=10)

# ==================== 2. CHURN RATE BY CONTRACT TYPE ====================
plt.subplot(5, 4, 2)
churn_by_contract = df.groupby('Contract')['Churn_binary'].mean() * 100
bars2 = plt.bar(churn_by_contract.index, churn_by_contract.values,
               color=['#FFCC99', '#CC99FF', '#99CCFF'])
plt.title('2. Churn Rate by Contract Type', fontsize=12, fontweight='bold')
plt.xlabel('Contract Type')
plt.ylabel('Churn Rate (%)')
plt.ylim(0, 50)
plt.xticks(rotation=45)
plt.grid(True, alpha=0.3)
for i, v in enumerate(churn_by_contract.values):
    plt.text(i, v + 1, f'{v:.1f}%', ha='center', fontsize=10)

# ==================== 3. TENURE DISTRIBUTION COMPARISON ====================
plt.subplot(5, 4, 3)
plt.hist([not_churned['tenure'], churned['tenure']],
         bins=15, label=['Not Churned', 'Churned'],
         alpha=0.7, color=['#66CC66', '#FF6666'], edgecolor='black')
plt.title('3. Tenure Distribution', fontsize=12, fontweight='bold')
plt.xlabel('Tenure (months)')
plt.ylabel('Count')
plt.legend()
plt.grid(True, alpha=0.3)

# Add text with averages
avg_tenure_churned = churned['tenure'].mean()
avg_tenure_not = not_churned['tenure'].mean()
plt.text(0.05, 0.95, f'Avg Churned: {avg_tenure_churned:.1f} months\nAvg Not Churned: {a
         transform=plt.gca().transAxes, fontsize=9,
         verticalalignment='top', bbox=dict(boxstyle='round', facecolor='wheat', alpha=0

# ==================== 4. MONTHLY CHARGES COMPARISON ====================
plt.subplot(5, 4, 4)
plt.hist([not_churned['MonthlyCharges'], churned['MonthlyCharges']],
         bins=20, label=['Not Churned', 'Churned'],
         alpha=0.7, color=['#66CC66', '#FF6666'], edgecolor='black')
plt.title('4. Monthly Charges Distribution', fontsize=12, fontweight='bold')
plt.xlabel('Monthly Charges ($)')
plt.ylabel('Count')
plt.legend()
plt.grid(True, alpha=0.3)

# Add text with averages
avg_monthly_churned = churned['MonthlyCharges'].mean()
avg_monthly_not = not_churned['MonthlyCharges'].mean()
plt.text(0.05, 0.95, f'Avg Churned: ${avg_monthly_churned:.2f}\nAvg Not Churned: ${avg_m
         transform=plt.gca().transAxes, fontsize=9,
         verticalalignment='top', bbox=dict(boxstyle='round', facecolor='wheat', alpha=0

# ==================== 5. CHURN RATE BY PAYMENT METHOD ====================
plt.subplot(5, 4, 5)
payment_churn = df.groupby('PaymentMethod')['Churn_binary'].mean() * 100
sorted_payments = payment_churn.sort_values(ascending=False)
bars5 = plt.bar(range(len(sorted_payments)), sorted_payments.values,
               color=['#FF9999', '#66B2FF', '#99FF99', '#FFCC66'])
```

```
plt.title('5. Churn Rate by Payment Method', fontsize=12, fontweight='bold')
plt.xlabel('Payment Method')
plt.ylabel('Churn Rate (%)')
plt.xticks(range(len(sorted_payments)), sorted_payments.index, rotation=45, ha='right')
plt.ylim(0, 50)
plt.grid(True, alpha=0.3)
for i, v in enumerate(sorted_payments.values):
    plt.text(i, v + 1, f'{v:.1f}%', ha='center', fontsize=9)

# ==================== 6. CHURN RATE BY SENIOR CITIZEN STATUS ====================
plt.subplot(5, 4, 6)
senior_churn = df.groupby('SeniorCitizen')['Churn_binary'].mean() * 100
bars6 = plt.bar(['Not Senior', 'Senior'], senior_churn.values,
               color=['#66CCCC', '#FF9966'])
plt.title('6. Churn Rate by Senior Citizen', fontsize=12, fontweight='bold')
plt.xlabel('Senior Citizen Status')
plt.ylabel('Churn Rate (%)')
plt.ylim(0, 35)
plt.grid(True, alpha=0.3)
for i, v in enumerate(senior_churn.values):
    plt.text(i, v + 1, f'{v:.1f}%', ha='center', fontsize=10)

# ==================== 7. ADD-ON SERVICES ADOPTION ====================
plt.subplot(5, 4, 7)
services = ['OnlineSecurity', 'OnlineBackup', 'DeviceProtection',
            'TechSupport', 'StreamingTV', 'StreamingMovies']

service_adoption = []
for service in services:
    churned_adoption = (churned[service] == 'Yes').mean() * 100
    not_churned_adoption = (not_churned[service] == 'Yes').mean() * 100
    service_adoption.append([churned_adoption, not_churned_adoption])

service_adoption = np.array(service_adoption)
x = np.arange(len(services))
width = 0.35

plt.bar(x - width/2, service_adoption[:, 0], width, label='Churned', color='#FF6666')
plt.bar(x + width/2, service_adoption[:, 1], width, label='Not Churned', color='#66CC66'
plt.title('7. Add-on Services Adoption', fontsize=12, fontweight='bold')
plt.xlabel('Service')
plt.ylabel('Adoption Rate (%)')
plt.xticks(x, services, rotation=45, ha='right')
plt.legend()
plt.grid(True, alpha=0.3)

# ==================== 8. PAPERLESS BILLING + CONTRACT CHURN ====================
plt.subplot(5, 4, 8)
# Create pivot table
pivot_churn = df.pivot_table(values='Churn_binary',
                             index='Contract',
                             columns='PaperlessBilling',
                             aggfunc='mean') * 100

x = np.arange(len(pivot_churn.index))
width = 0.35

plt.bar(x - width/2, pivot_churn['No'].values, width, label='Paperless: No', color='#66B
plt.bar(x + width/2, pivot_churn['Yes'].values, width, label='Paperless: Yes', color='#F
plt.title('8. Churn: Contract × Paperless Billing', fontsize=12, fontweight='bold')
plt.xlabel('Contract Type')
plt.ylabel('Churn Rate (%)')
plt.xticks(x, pivot_churn.index, rotation=45, ha='right')
plt.legend()
plt.grid(True, alpha=0.3)

# ==================== 9. TOTAL CHARGES VS TENURE SCATTER ====================
plt.subplot(5, 4, 9)
# Sample for better visualization
sample_df = df.sample(n=300, random_state=42)

plt.scatter(sample_df[sample_df['Churn'] == 'No']['tenure'],
           sample_df[sample_df['Churn'] == 'No']['TotalCharges'],
           alpha=0.6, c='#66CC66', label='Not Churned', s=30)

plt.scatter(sample_df[sample_df['Churn'] == 'Yes']['tenure'],
           sample_df[sample_df['Churn'] == 'Yes']['TotalCharges'],
           alpha=0.6, c='#FF6666', label='Churned', s=30)

plt.title('9. Total Charges vs Tenure', fontsize=12, fontweight='bold')
plt.xlabel('Tenure (months)')
plt.ylabel('Total Charges ($)')
plt.legend()
plt.grid(True, alpha=0.3)

# ==================== 10. SERVICE BUNDLES CHURN RATE ====================
plt.subplot(5, 4, 10)
# Create service bundles
def create_service_bundle(row):
    services_count = 0
    for service in ['PhoneService', 'StreamingTV', 'StreamingMovies']:
        if row[service] == 'Yes':
            services_count += 1
    if row['InternetService'] != 'No':
        services_count += 1
    return services_count

df['ServiceBundle'] = df.apply(create_service_bundle, axis=1)
bundle_churn = df.groupby('ServiceBundle')['Churn_binary'].mean() * 100

plt.bar(bundle_churn.index, bundle_churn.values,
       color=['#FF9999', '#66B2FF', '#99FF99', '#FFCC66', '#CC99FF'])
plt.title('10. Churn Rate by Service Bundle Size', fontsize=12, fontweight='bold')
plt.xlabel('Number of Services')
plt.ylabel('Churn Rate (%)')
plt.xticks(range(len(bundle_churn)))
plt.grid(True, alpha=0.3)

for i, v in enumerate(bundle_churn.values):
    plt.text(i, v + 1, f'{v:.1f}%', ha='center', fontsize=9)

# ==================== 11. OVERALL CHURN DISTRIBUTION ====================
plt.subplot(5, 4, 11)
churn_counts = df['Churn'].value_counts()
colors = ['#66CC66', '#FF6666']
plt.pie(churn_counts.values, labels=['Not Churned', 'Churned'],
        colors=colors, autopct='%1.1f%%', startangle=90)
plt.title('11. Overall Churn Distribution', fontsize=12, fontweight='bold')
```

```
# ==================== 12. CHURN BY DEPENDENTS ====================
plt.subplot(5, 4, 12)
dependents_churn = df.groupby('Dependents')['Churn_binary'].mean() * 100
bars12 = plt.bar(['No Dependents', 'Has Dependents'], dependents_churn.values,
                color=['#FF9999', '#66B2FF'])
plt.title('12. Churn Rate by Dependents', fontsize=12, fontweight='bold')
plt.xlabel('Dependents Status')
plt.ylabel('Churn Rate (%)')
plt.ylim(0, 35)
plt.grid(True, alpha=0.3)
for i, v in enumerate(dependents_churn.values):
    plt.text(i, v + 1, f'{v:.1f}%', ha='center', fontsize=10)

# ==================== 13. CHURN BY PARTNER STATUS ====================
plt.subplot(5, 4, 13)
partner_churn = df.groupby('Partner')['Churn_binary'].mean() * 100
bars13 = plt.bar(['No Partner', 'Has Partner'], partner_churn.values,
                color=['#99FF99', '#FFCC66'])
plt.title('13. Churn Rate by Partner Status', fontsize=12, fontweight='bold')
plt.xlabel('Partner Status')
plt.ylabel('Churn Rate (%)')
plt.ylim(0, 35)
plt.grid(True, alpha=0.3)
for i, v in enumerate(partner_churn.values):
    plt.text(i, v + 1, f'{v:.1f}%', ha='center', fontsize=10)

# ==================== 14. MONTHLY CHARGES BY CONTRACT ====================
plt.subplot(5, 4, 14)
contract_charges = df.groupby('Contract')['MonthlyCharges'].mean().sort_values()
plt.bar(contract_charges.index, contract_charges.values,
       color=['#FF9999', '#66B2FF', '#99FF99'])
plt.title('14. Avg Monthly Charges by Contract', fontsize=12, fontweight='bold')
plt.xlabel('Contract Type')
plt.ylabel('Avg Monthly Charges ($)')
plt.xticks(rotation=45, ha='right')
plt.grid(True, alpha=0.3)

for i, v in enumerate(contract_charges.values):
    plt.text(i, v + 1, f'${v:.1f}', ha='center', fontsize=9)

# ==================== 15. TENURE BY CONTRACT TYPE ====================
plt.subplot(5, 4, 15)
tenure_by_contract = df.groupby('Contract')['tenure'].mean()
bars15 = plt.bar(tenure_by_contract.index, tenure_by_contract.values,
                color=['#FF9999', '#66B2FF', '#99FF99'])
plt.title('15. Avg Tenure by Contract Type', fontsize=12, fontweight='bold')
plt.xlabel('Contract Type')
plt.ylabel('Avg Tenure (months)')
plt.xticks(rotation=45, ha='right')
plt.grid(True, alpha=0.3)

for i, v in enumerate(tenure_by_contract.values):
    plt.text(i, v + 1, f'{v:.1f}', ha='center', fontsize=9)

# ==================== 16. CHURN HEATMAP (SIMULATED) ====================
plt.subplot(5, 4, 16)
# Create correlation matrix for key numerical features
numerical_cols = ['tenure', 'MonthlyCharges', 'TotalCharges', 'Churn_binary']
```

```
corr_matrix = df[numerical_cols].corr()

# Create heatmap using imshow
plt.imshow(corr_matrix, cmap='coolwarm', aspect='auto')
plt.colorbar(label='Correlation')
plt.title('16. Feature Correlation Heatmap', fontsize=12, fontweight='bold')
plt.xticks(range(len(numerical_cols)), numerical_cols, rotation=45)
plt.yticks(range(len(numerical_cols)), numerical_cols)

# Add correlation values
for i in range(len(numerical_cols)):
    for j in range(len(numerical_cols)):
        plt.text(j, i, f'{corr_matrix.iloc[i, j]:.2f}',
                ha='center', va='center',
                color='white' if abs(corr_matrix.iloc[i, j]) > 0.5 else 'black',
                fontsize=9)

# ==================== 17. CUSTOMER DISTRIBUTION BY CONTRACT ====================
plt.subplot(5, 4, 17)
contract_dist = df['Contract'].value_counts()
colors17 = ['#FF9999', '#66B2FF', '#99FF99']
plt.pie(contract_dist.values, labels=contract_dist.index,
        colors=colors17, autopct='%1.1f%%', startangle=90)
plt.title('17. Customer Distribution by Contract', fontsize=12, fontweight='bold')

# ==================== 18. CHURN RATE BY GENDER ====================
plt.subplot(5, 4, 18)
gender_churn = df.groupby('gender')['Churn_binary'].mean() * 100
bars18 = plt.bar(gender_churn.index, gender_churn.values,
                color=['#FFB6C1', '#ADD8E6'])  # Pink for Female, Light Blue for Male
plt.title('18. Churn Rate by Gender', fontsize=12, fontweight='bold')
plt.xlabel('Gender')
plt.ylabel('Churn Rate (%)')
plt.ylim(0, 30)
plt.grid(True, alpha=0.3)
for i, v in enumerate(gender_churn.values):
    plt.text(i, v + 0.5, f'{v:.1f}%', ha='center', fontsize=10)

# ==================== 19. TOP 5 HIGHEST CHURN COHORTS ====================
plt.subplot(5, 4, 19)
# Create cohorts based on multiple factors
cohort_data = []
cohorts = [
    ('Fiber Optic', 'Month-to-month'),
    ('Fiber Optic', 'One year'),
    ('DSL', 'Month-to-month'),
    ('No Internet', 'Month-to-month'),
    ('Fiber Optic', 'Two year')
]

for internet, contract in cohorts:
    cohort_df = df[(df['InternetService'] == internet) & (df['Contract'] == contract)]
    if len(cohort_df) > 0:
        churn_rate = cohort_df['Churn_binary'].mean() * 100
        cohort_data.append((f'{internet}\n{contract}', churn_rate, len(cohort_df)))

# Sort by churn rate
cohort_data.sort(key=lambda x: x[1], reverse=True)
top_5 = cohort_data[:5]
```

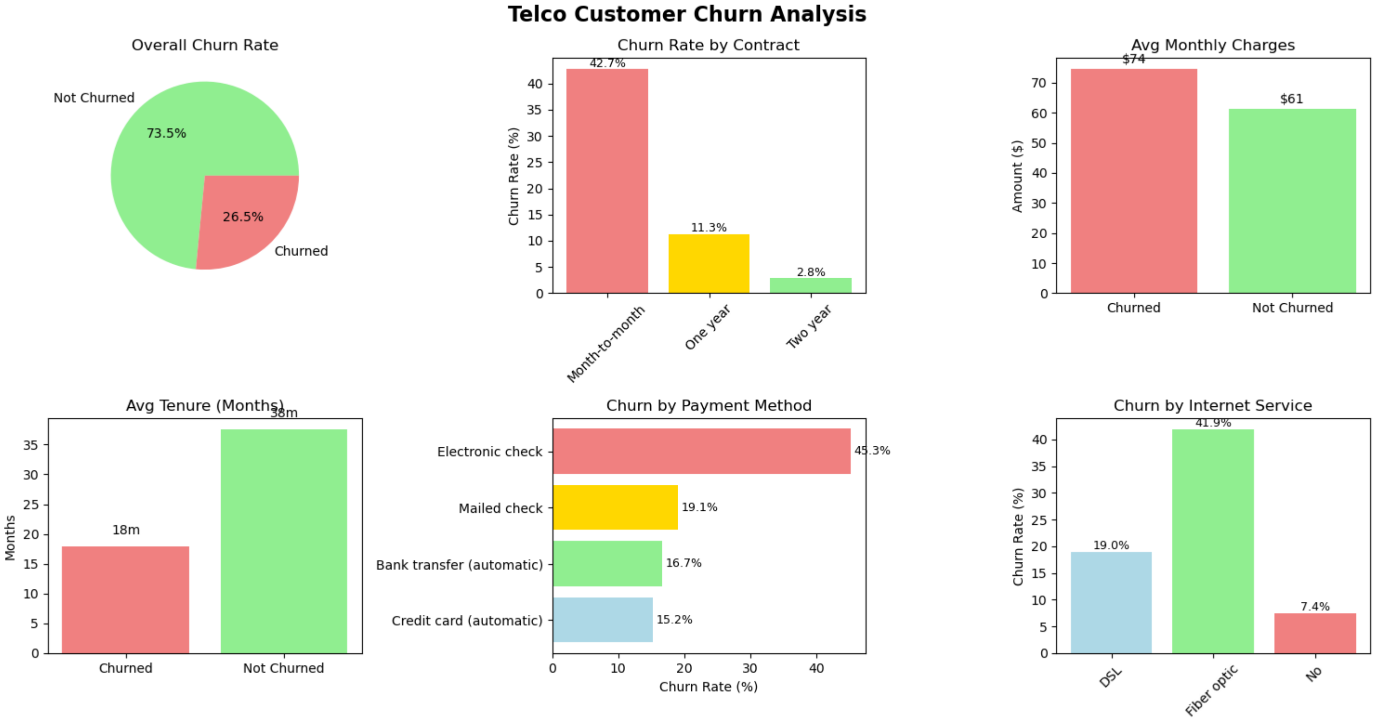
```python
cohort_names = [x[0] for x in top_5]
churn_rates = [x[1] for x in top_5]
sizes = [x[2] for x in top_5]

bars19 = plt.bar(cohort_names, churn_rates, color='#FF6666')
plt.title('19. Top 5 High Churn Cohorts', fontsize=12, fontweight='bold')
plt.xlabel('Cohort (Internet × Contract)')
plt.ylabel('Churn Rate (%)')
plt.xticks(rotation=45, ha='right')
plt.grid(True, alpha=0.3)

for i, (v, s) in enumerate(zip(churn_rates, sizes)):
    plt.text(i, v + 1, f'{v:.1f}%\n(n={s})', ha='center', fontsize=8)

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


1. Churn Rate by Internet Service
Churn Rate (%)
19.0%
41.9%
7.4%
DSL
Fiber optic
No
Internet Service
2. Churn Rate by Contract Type
42.7%
11.3%
2.8%
Month-to-month
One year
Two year
Contract Type
41.7%
3. Tenure Distribution
Avg Churned: 18.0 months
Avg Not Churned: 37.6 months
Not Churned
Churned
Count
Tenure (months)
4. Monthly Charges Distribution
Avg Churned: $74.44
Avg Not Churned: $61.27
Monthly Charges ($)
5. Churn Rate by Payment Method
45.3%
19.1%
16.7%
15.2%
Electronic check
Mailed check
Bank transfer (automatic)
Credit card (automatic)
Payment Method
6. Churn Rate by Senior Citizen
23.6%
Not Senior
Senior
Senior Citizen Status
7. Add-on Services Adoption
Adoption Rate (%)
OnlineSecurity
OnlineBackup
DeviceProtection
TechSupport
StreamingTV
StreamingMovies
Service
8. Churn: Contract × Paperless Billing
Paperless: No
Paperless: Yes
9. Total Charges vs Tenure
Total Charges ($)
10. Churn Rate by Service Bundle Size
35.1%
30.6%
30.3%
10.4%
Number of Services
11. Overall Churn Distribution
Churned
26.5%
73.5%
Not Churned
12. Churn Rate by Dependents
31.3%
15.5%
No Dependents
Has Dependents
Dependents Status
13. Churn Rate by Partner Status
33.0%
19.7%
No Partner
Has Partner
Partner Status
14. Avg Monthly Charges by Contract
Avg Monthly Charges ($)
$60.8
$65.0
$66.4
15. Avg Tenure by Contract Type
Avg Tenure (months)
18.0
42.0
56.7
16. Feature Correlation Heatmap
tenure
MonthlyCharges
TotalCharges
Churn_binary
1.00 0.25 0.83 -0.35
0.25 1.00 0.65 0.19
0.83 0.65 1.00 -0.20
-0.35 0.19 -0.20 1.00
Correlation
17. Customer Distribution by Contract
One year
20.9%
Month-to-month
55.0%
24.1%
Two year
18. Churn Rate by Gender
26.9%
26.2%
Female
Male
Gender
19. Top 5 High Churn Cohorts
(n=1223)
DSL
Month-to-month
Cohort (Internet × Contract)

In [11]:

```python
import pandas as pd
import matplotlib.pyplot as plt

# Load and prepare data
df = pd.read_csv("C:\\Users\\prems\\Videos\\power bi\\Telco_Cusomer_Churn.csv")

# Simple data prep
df['TotalCharges'] = pd.to_numeric(df['TotalCharges'], errors='coerce')
df['TotalCharges'] = df['TotalCharges'].fillna(0)
df['Churn_binary'] = df['Churn'].map({'Yes': 1, 'No': 0})

# Create 6 simple visualizations
fig, axes = plt.subplots(2, 3, figsize=(15, 8))
fig.suptitle('Telco Customer Churn Analysis', fontsize=16, fontweight='bold')

# 1. Overall Churn Rate
churn_counts = df['Churn'].value_counts()
axes[0,0].pie(churn_counts.values, labels=['Not Churned', 'Churned'],
              colors=['lightgreen', 'lightcoral'], autopct='%1.1f%%')
axes[0,0].set_title('Overall Churn Rate')

# 2. Churn by Contract
contract_churn = df.groupby('Contract')['Churn_binary'].mean() * 100
axes[0,1].bar(contract_churn.index, contract_churn.values,
              color=['lightcoral', 'gold', 'lightgreen'])
axes[0,1].set_title('Churn Rate by Contract')
axes[0,1].set_ylabel('Churn Rate (%)')
axes[0,1].tick_params(axis='x', rotation=45)
# Add labels
for i, v in enumerate(contract_churn.values):
    axes[0,1].text(i, v + 0.5, f'{v:.1f}%', ha='center', fontsize=9)

# 3. Monthly Charges Comparison
churned_avg = df[df['Churn'] == 'Yes']['MonthlyCharges'].mean()
not_churned_avg = df[df['Churn'] == 'No']['MonthlyCharges'].mean()
axes[0,2].bar(['Churned', 'Not Churned'], [churned_avg, not_churned_avg],
              color=['lightcoral', 'lightgreen'])
axes[0,2].set_title('Avg Monthly Charges')
axes[0,2].set_ylabel('Amount ($)')
# Add labels
axes[0,2].text(0, churned_avg + 2, f'${churned_avg:.0f}', ha='center', fontsize=10)
axes[0,2].text(1, not_churned_avg + 2, f'${not_churned_avg:.0f}', ha='center', fontsize=

# 4. Tenure Distribution
tenure_churned = df[df['Churn'] == 'Yes']['tenure'].mean()
tenure_not = df[df['Churn'] == 'No']['tenure'].mean()
axes[1,0].bar(['Churned', 'Not Churned'], [tenure_churned, tenure_not],
              color=['lightcoral', 'lightgreen'])
axes[1,0].set_title('Avg Tenure (Months)')
axes[1,0].set_ylabel('Months')
# Add labels
axes[1,0].text(0, tenure_churned + 2, f'{tenure_churned:.0f}m', ha='center', fontsize=10
axes[1,0].text(1, tenure_not + 2, f'{tenure_not:.0f}m', ha='center', fontsize=10)

# 5. Payment Method Churn
payment_churn = df.groupby('PaymentMethod')['Churn_binary'].mean() * 100
payment_churn = payment_churn.sort_values()
bars = axes[1,1].barh(range(len(payment_churn)), payment_churn.values,
                     color=['lightblue', 'lightgreen', 'gold', 'lightcoral'])
```

```
axes[1,1].set_title('Churn by Payment Method')
axes[1,1].set_xlabel('Churn Rate (%)')
axes[1,1].set_yticks(range(len(payment_churn)))
axes[1,1].set_yticklabels(payment_churn.index)
# Add labels
for i, v in enumerate(payment_churn.values):
    axes[1,1].text(v + 0.5, i, f'{v:.1f}%', va='center', fontsize=9)

# 6. Internet Service Churn
internet_churn = df.groupby('InternetService')['Churn_binary'].mean() * 100
axes[1,2].bar(internet_churn.index, internet_churn.values,
              color=['lightblue', 'lightgreen', 'lightcoral'])
axes[1,2].set_title('Churn by Internet Service')
axes[1,2].set_ylabel('Churn Rate (%)')
axes[1,2].tick_params(axis='x', rotation=45)
# Add labels
for i, v in enumerate(internet_churn.values):
    axes[1,2].text(i, v + 0.5, f'{v:.1f}%', ha='center', fontsize=9)

plt.tight_layout()
plt.show()
# Print key insights
print("="*50)
print("KEY INSIGHTS")
print("="*50)
print(f"1. Overall Churn Rate: {(df['Churn_binary'].mean()*100):.1f}%")
print(f"2. Highest Churn by Contract: {contract_churn.idxmax()} ({contract_churn.max():.
print(f"3. Avg Monthly Charges: Churned pay ${churned_avg - not_churned_avg:.0f} more")
print(f"4. Avg Tenure: Churned customers stay {tenure_not - tenure_churned:.0f} months l
print(f"5. Highest Churn Payment: {payment_churn.idxmax()} ({payment_churn.max():.1f}%)"
print(f"6. Highest Churn Internet: {internet_churn.idxmax()} ({internet_churn.max():.1f}
print("="*50)
```

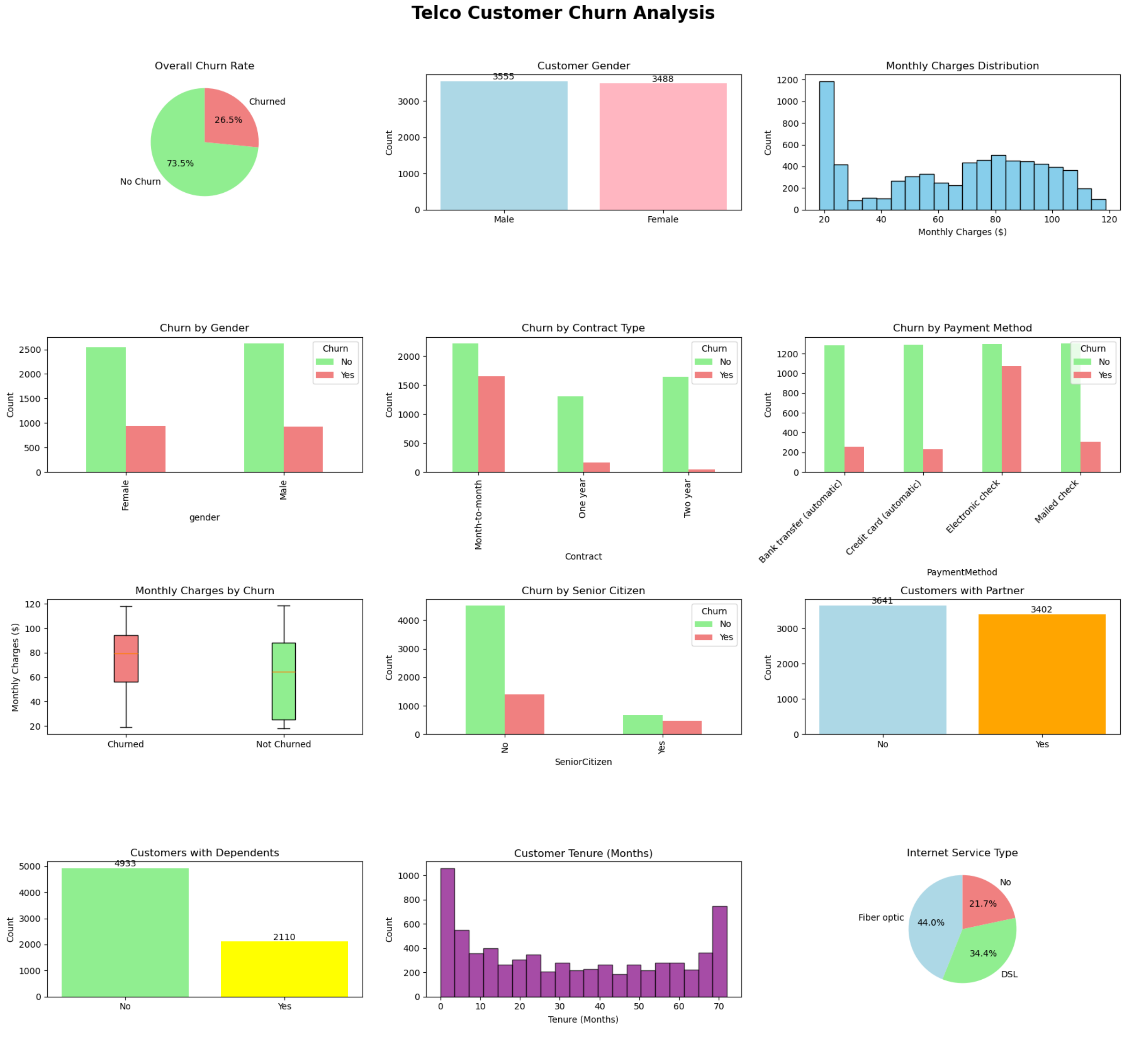


```
==================================================
KEY INSIGHTS
==================================================
1. Overall Churn Rate: 26.5%
2. Highest Churn by Contract: Month-to-month (42.7%)
3. Avg Monthly Charges: Churned pay $13 more
4. Avg Tenure: Churned customers stay 20 months less
5. Highest Churn Payment: Electronic check (45.3%)
6. Highest Churn Internet: Fiber optic (41.9%)
==================================================
```

In [3]:

```
# 1. Setup and Load Data
import pandas as pd
import matplotlib.pyplot as plt

# Load data - FIXED PATH
df = pd.read_csv("C:\\Users\\prems\\Videos\\power bi\\Telco_Cusomer_Churn.csv")

# Convert SeniorCitizen to Yes/No
df['SeniorCitizen'] = df['SeniorCitizen'].map({0: 'No', 1: 'Yes'})

# Convert TotalCharges to numeric and handle missing values
df['TotalCharges'] = pd.to_numeric(df['TotalCharges'], errors='coerce')
df['TotalCharges'] = df['TotalCharges'].fillna(0)

# Create one big figure with all subplots
fig, axes = plt.subplots(4, 3, figsize=(18, 16))
fig.suptitle('Telco Customer Churn Analysis', fontsize=20, fontweight='bold', y=1.02)

# 1. Overall Churn Rate (Top Left)
churn_counts = df['Churn'].value_counts()
axes[0,0].pie(churn_counts.values, labels=['No Churn', 'Churned'],
              autopct='%1.1f%%', colors=['lightgreen', 'lightcoral'],
              startangle=90)
axes[0,0].set_title('Overall Churn Rate')

# 2. Customer Gender Distribution
gender_counts = df['gender'].value_counts()
axes[0,1].bar(gender_counts.index, gender_counts.values, color=['lightblue', 'lightpink'
axes[0,1].set_title('Customer Gender')
axes[0,1].set_ylabel('Count')
# Add numbers on bars
for i, value in enumerate(gender_counts.values):
    axes[0,1].text(i, value + 50, str(value), ha='center')

# 3. Monthly Charges Distribution
axes[0,2].hist(df['MonthlyCharges'], bins=20, color='skyblue', edgecolor='black')
axes[0,2].set_title('Monthly Charges Distribution')
axes[0,2].set_xlabel('Monthly Charges ($)')
axes[0,2].set_ylabel('Count')

# 4. Churn by Gender
churn_gender = pd.crosstab(df['gender'], df['Churn'])
churn_gender.plot(kind='bar', ax=axes[1,0], color=['lightgreen', 'lightcoral'])
axes[1,0].set_title('Churn by Gender')
axes[1,0].set_ylabel('Count')
axes[1,0].legend(['No', 'Yes'], title='Churn')

# 5. Churn by Contract Type
churn_contract = pd.crosstab(df['Contract'], df['Churn'])
churn_contract.plot(kind='bar', ax=axes[1,1], color=['lightgreen', 'lightcoral'])
```

```python
axes[1,1].set_title('Churn by Contract Type')
axes[1,1].set_ylabel('Count')
axes[1,1].legend(['No', 'Yes'], title='Churn')

# 6. Churn by Payment Method
churn_payment = pd.crosstab(df['PaymentMethod'], df['Churn'])
churn_payment.plot(kind='bar', ax=axes[1,2], color=['lightgreen', 'lightcoral'])
axes[1,2].set_title('Churn by Payment Method')
axes[1,2].set_ylabel('Count')
axes[1,2].set_xticklabels(churn_payment.index, rotation=45, ha='right')
axes[1,2].legend(['No', 'Yes'], title='Churn')

# 7. Monthly Charges by Churn (Box Plot) - FIXED VERSION
churned_data = df[df['Churn'] == 'Yes']['MonthlyCharges']
not_churned_data = df[df['Churn'] == 'No']['MonthlyCharges']
box_data = [churned_data, not_churned_data]

# Using boxplot with correct parameter name
bp = axes[2,0].boxplot(box_data, patch_artist=True)
axes[2,0].set_xticklabels(['Churned', 'Not Churned'])
bp['boxes'][0].set_facecolor('lightcoral')
bp['boxes'][1].set_facecolor('lightgreen')
axes[2,0].set_title('Monthly Charges by Churn')
axes[2,0].set_ylabel('Monthly Charges ($)')

# 8. Senior Citizen Churn
churn_senior = pd.crosstab(df['SeniorCitizen'], df['Churn'])
churn_senior.plot(kind='bar', ax=axes[2,1], color=['lightgreen', 'lightcoral'])
axes[2,1].set_title('Churn by Senior Citizen')
axes[2,1].set_ylabel('Count')
axes[2,1].legend(['No', 'Yes'], title='Churn')

# 9. Partner Status Distribution
partner_counts = df['Partner'].value_counts()
axes[2,2].bar(partner_counts.index, partner_counts.values, color=['lightblue', 'orange']
axes[2,2].set_title('Customers with Partner')
axes[2,2].set_ylabel('Count')
# Add numbers on bars
for i, value in enumerate(partner_counts.values):
    axes[2,2].text(i, value + 50, str(value), ha='center')

# 10. Dependents Distribution
dependents_counts = df['Dependents'].value_counts()
axes[3,0].bar(dependents_counts.index, dependents_counts.values, color=['lightgreen', 'y
axes[3,0].set_title('Customers with Dependents')
axes[3,0].set_ylabel('Count')
# Add numbers on bars
for i, value in enumerate(dependents_counts.values):
    axes[3,0].text(i, value + 50, str(value), ha='center')

# 11. Tenure Distribution
axes[3,1].hist(df['tenure'], bins=20, color='purple', alpha=0.7, edgecolor='black')
axes[3,1].set_title('Customer Tenure (Months)')
axes[3,1].set_xlabel('Tenure (Months)')
axes[3,1].set_ylabel('Count')

# 12. Internet Service Type
internet_counts = df['InternetService'].value_counts()
axes[3,2].pie(internet_counts.values, labels=internet_counts.index,
              autopct='%1.1f%%', startangle=90,
              colors=['lightblue', 'lightgreen', 'lightcoral'])
axes[3,2].set_title('Internet Service Type')

# Adjust layout
plt.tight_layout()
plt.show()

# Print summary statistics
print("="*60)
print("SUMMARY STATISTICS")
print("="*60)
print(f"Total Customers: {len(df)}")
print(f"Churn Rate: {(df['Churn'] == 'Yes').mean() * 100:.1f}%")
print(f"Average Monthly Charge: ${df['MonthlyCharges'].mean():.2f}")
print(f"Average Tenure: {df['tenure'].mean():.1f} months")
print(f"Female Customers: {len(df[df['gender'] == 'Female'])} ({(df['gender'] == 'Female
print(f"Male Customers: {len(df[df['gender'] == 'Male'])} ({(df['gender'] == 'Male').mea
print(f"Senior Citizens: {len(df[df['SeniorCitizen'] == 'Yes'])} ({(df['SeniorCitizen']
print("="*60)
```



```
============================================================
SUMMARY STATISTICS
============================================================
Total Customers: 7043
Churn Rate: 26.5%
Average Monthly Charge: $64.76
Average Tenure: 32.4 months
Female Customers: 3488 (49.5%)
Male Customers: 3555 (50.5%)
Senior Citizens: 1142 (16.2%)
============================================================
```

In [2]:

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

# Load the dataset
df = pd.read_csv("C:\\Users\\prems\\Videos\\power bi\\Telco_Cusomer_Churn.csv")

# Data cleaning
df['TotalCharges'] = pd.to_numeric(df['TotalCharges'], errors='coerce')
df['TotalCharges'] = df['TotalCharges'].fillna(0)

# Convert Churn to binary
df['ChurnBinary'] = df['Churn'].map({'Yes': 1, 'No': 0})

# 1. Overall Churn Rate
total_customers = len(df)
churned_customers = df['ChurnBinary'].sum()
churn_rate = (churned_customers / total_customers) * 100

# 2. Tenure Impact Analysis
short_tenure = df[df['tenure'] < 12]
long_tenure = df[df['tenure'] >= 24]
short_tenure_churn_rate = (short_tenure['ChurnBinary'].sum() / len(short_tenure)) * 100
long_tenure_churn_rate = (long_tenure['ChurnBinary'].sum() / len(long_tenure)) * 100
tenure_churn_ratio = short_tenure_churn_rate / long_tenure_churn_rate

# 3. Contract Type Analysis
contract_churn = df.groupby('Contract')['ChurnBinary'].mean() * 100

# 4. Internet Service Analysis
internet_churn = df.groupby('InternetService')['ChurnBinary'].mean() * 100

# 5. Payment Method Analysis
payment_churn = df.groupby('PaymentMethod')['ChurnBinary'].mean() * 100

# 6. Monthly Charges Impact
avg_monthly_charge = df['MonthlyCharges'].mean()
high_charge_churn = df[df['MonthlyCharges'] > avg_monthly_charge]['ChurnBinary'].mean()
low_charge_churn = df[df['MonthlyCharges'] <= avg_monthly_charge]['ChurnBinary'].mean()

# 7. Senior Citizens Analysis
senior_churn = df[df['SeniorCitizen'] == 1]['ChurnBinary'].mean() * 100
non_senior_churn = df[df['SeniorCitizen'] == 0]['ChurnBinary'].mean() * 100

# 8. Service Add-ons Analysis
def analyze_service_feature(feature):
    with_service = df[df[feature] == 'Yes']['ChurnBinary'].mean() * 100
    without_service = df[df[feature] == 'No']['ChurnBinary'].mean() * 100
    return with_service, without_service

tech_support_churn = analyze_service_feature('TechSupport')
online_security_churn = analyze_service_feature('OnlineSecurity')
online_backup_churn = analyze_service_feature('OnlineBackup')

# 9. Paperless Billing Analysis
paperless_churn = df[df['PaperlessBilling'] == 'Yes']['ChurnBinary'].mean() * 100
non_paperless_churn = df[df['PaperlessBilling'] == 'No']['ChurnBinary'].mean() * 100

# 10. Customer Lifetime Value Analysis
avg_total_charges_churned = df[df['Churn'] == 'Yes']['TotalCharges'].mean()
avg_total_charges_retained = df[df['Churn'] == 'No']['TotalCharges'].mean()
revenue_difference_ratio = avg_total_charges_retained / avg_total_charges_churned

# Print Insights
print("="*60)
print("TELCO CUSTOMER CHURN ANALYSIS - KEY INSIGHTS")
print("="*60)

print(f"\n1. Overall Churn Rate: {churn_rate:.1f}%")
print(f"   Total Customers: {total_customers:,}")
print(f"   Churned Customers: {churned_customers:,}")

print(f"\n2. Tenure Impact:")
print(f"   Short-tenure (<12 months) churn: {short_tenure_churn_rate:.1f}%")
print(f"   Long-tenure (24+ months) churn: {long_tenure_churn_rate:.1f}%")
print(f"   Short-tenure customers are {tenure_churn_ratio:.1f}× more likely to churn")

print(f"\n3. Contract Type Analysis:")
for contract, rate in contract_churn.items():
    print(f"   {contract}: {rate:.1f}% churn")

print(f"\n4. Internet Service Analysis:")
for service, rate in internet_churn.items():
    print(f"   {service}: {rate:.1f}% churn")

print(f"\n5. Payment Method Analysis:")
for method, rate in payment_churn.items():
    print(f"   {method}: {rate:.1f}% churn")

print(f"\n6. Monthly Charges Impact:")
print(f"   Average monthly charge: ${avg_monthly_charge:.2f}")
print(f"   Above-average charge churn: {high_charge_churn:.1f}%")
print(f"   Below-average charge churn: {low_charge_churn:.1f}%")

print(f"\n7. Senior Citizen Analysis:")
print(f"   Senior citizens: {senior_churn:.1f}% churn")
print(f"   Non-seniors: {non_senior_churn:.1f}% churn")

print(f"\n8. Service Add-ons Impact:")
print(f"   Tech Support - With: {tech_support_churn[0]:.1f}%, Without: {tech_support_chu
print(f"   Online Security - With: {online_security_churn[0]:.1f}%, Without: {online_sec
print(f"   Online Backup - With: {online_backup_churn[0]:.1f}%, Without: {online_backup_

service_ratio = tech_support_churn[1] / tech_support_churn[0]
print(f"   Customers without tech support are {service_ratio:.1f}× more likely to churn"
```

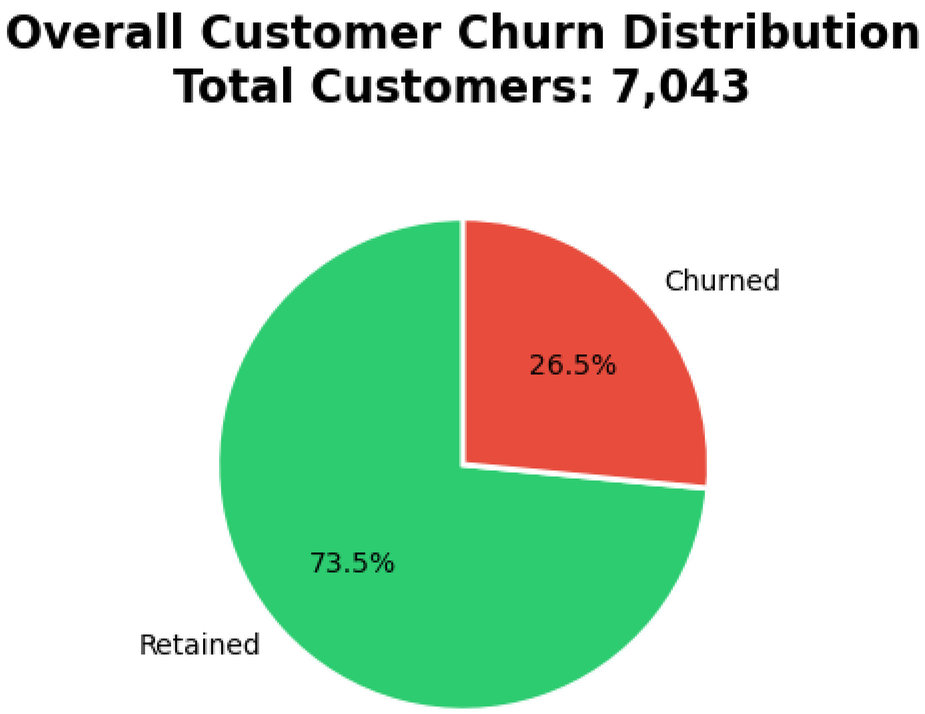
```python
print(f"\n9. Paperless Billing:")
print(f"   With paperless billing: {paperless_churn:.1f}% churn")
print(f"   Without paperless billing: {non_paperless_churn:.1f}% churn")

print(f"\n10. Customer Lifetime Value:")
print(f"    Average total charges (churned): ${avg_total_charges_churned:.2f}")
print(f"    Average total charges (retained): ${avg_total_charges_retained:.2f}")
print(f"    Retained customers generate {revenue_difference_ratio:.1f}× more revenue")

# Additional insights
print(f"\n" + "="*60)
print("ADDITIONAL INSIGHTS")
print("="*60)

# Partner/Dependents analysis
partner_churn = df.groupby('Partner')['ChurnBinary'].mean() * 100
dependents_churn = df.groupby('Dependents')['ChurnBinary'].mean() * 100

print(f"\nPartner Status:")
print(f"   With partner: {partner_churn['Yes']:.1f}% churn")
print(f"   Without partner: {partner_churn['No']:.1f}% churn")

print(f"\nDependents:")
print(f"   With dependents: {dependents_churn['Yes']:.1f}% churn")
print(f"   Without dependents: {dependents_churn['No']:.1f}% churn")

# Gender analysis
gender_churn = df.groupby('gender')['ChurnBinary'].mean() * 100
print(f"\nGender Analysis:")
print(f"   Female: {gender_churn['Female']:.1f}% churn")
print(f"   Male: {gender_churn['Male']:.1f}% churn")

# Streaming services analysis
streaming_tv_churn = analyze_service_feature('StreamingTV')
streaming_movies_churn = analyze_service_feature('StreamingMovies')

print(f"\nStreaming Services:")
print(f"   Streaming TV - With: {streaming_tv_churn[0]:.1f}%, Without: {streaming_tv_chu
print(f"   Streaming Movies - With: {streaming_movies_churn[0]:.1f}%, Without: {streamin
```

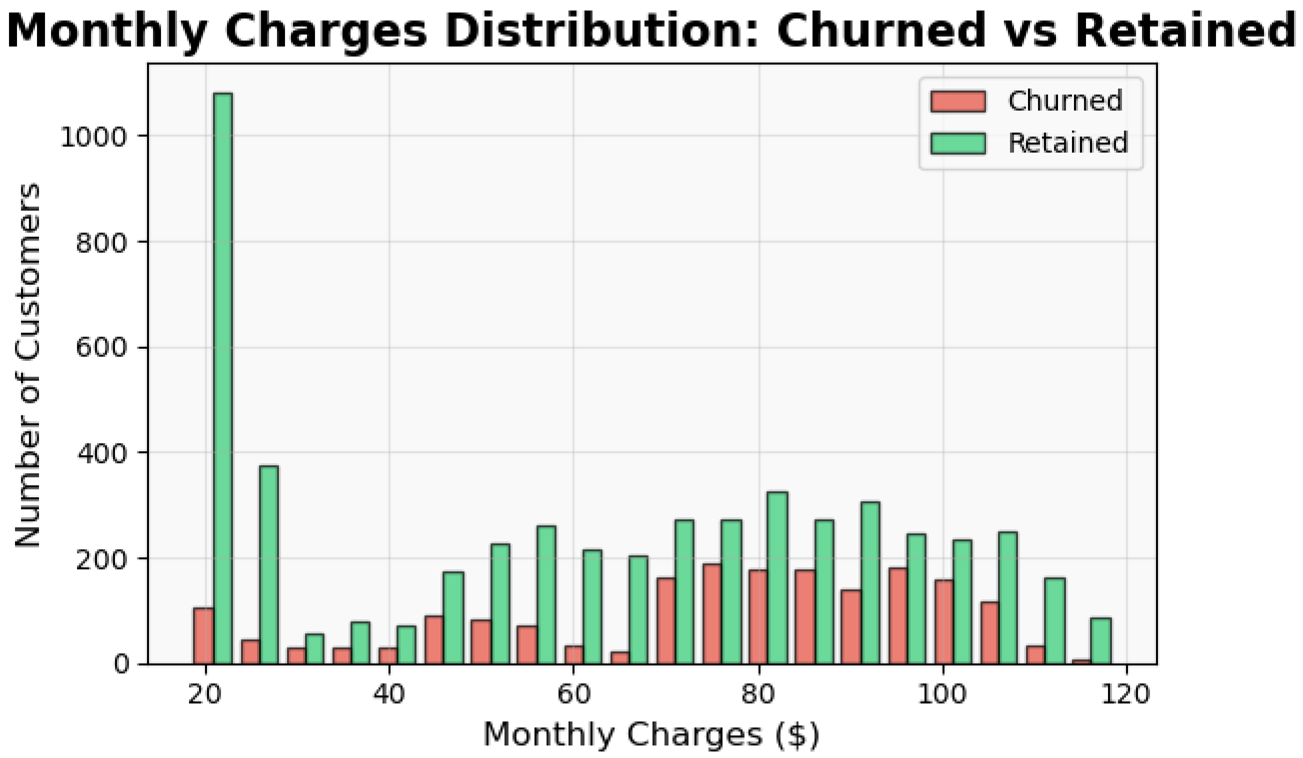
```
============================================================
TELCO CUSTOMER CHURN ANALYSIS - KEY INSIGHTS
============================================================

1. Overall Churn Rate: 26.5%
   Total Customers: 7,043
   Churned Customers: 1,869

2. Tenure Impact:
   Short-tenure (<12 months) churn: 48.3%
   Long-tenure (24+ months) churn: 14.3%
   Short-tenure customers are 3.4× more likely to churn

3. Contract Type Analysis:
   Month-to-month: 42.7% churn
   One year: 11.3% churn
   Two year: 2.8% churn
```

```
4. Internet Service Analysis:
   DSL: 19.0% churn
   Fiber optic: 41.9% churn
   No: 7.4% churn

5. Payment Method Analysis:
   Bank transfer (automatic): 16.7% churn
   Credit card (automatic): 15.2% churn
   Electronic check: 45.3% churn
   Mailed check: 19.1% churn

6. Monthly Charges Impact:
   Average monthly charge: $64.76
   Above-average charge churn: 34.5%
   Below-average charge churn: 16.5%

7. Senior Citizen Analysis:
   Senior citizens: 41.7% churn
   Non-seniors: 23.6% churn

8. Service Add-ons Impact:
   Tech Support - With: 15.2%, Without: 41.6%
   Online Security - With: 14.6%, Without: 41.8%
   Online Backup - With: 21.5%, Without: 39.9%
   Customers without tech support are 2.7× more likely to churn

9. Paperless Billing:
   With paperless billing: 33.6% churn
   Without paperless billing: 16.3% churn

10. Customer Lifetime Value:
    Average total charges (churned): $1531.80
    Average total charges (retained): $2549.91
    Retained customers generate 1.7× more revenue

============================================================
ADDITIONAL INSIGHTS
============================================================

Partner Status:
   With partner: 19.7% churn
   Without partner: 33.0% churn

Dependents:
   With dependents: 15.5% churn
   Without dependents: 31.3% churn

Gender Analysis:
   Female: 26.9% churn
   Male: 26.2% churn

Streaming Services:
   Streaming TV - With: 30.1%, Without: 33.5%
   Streaming Movies - With: 29.9%, Without: 33.7%

============================================================
```

In [32]:

```
# 1. Overall Churn Rate - Pie Chart
plt.figure(figsize=(7, 4))
churn_counts = df['Churn'].value_counts()
colors = ['#2ecc71', '#e74c3c']
plt.pie(churn_counts, labels=['Retained', 'Churned'], autopct='%1.1f%%',
        colors=colors, startangle=90, wedgeprops={'edgecolor': 'white', 'linewidth': 2})
plt.title('Overall Customer Churn Distribution\nTotal Customers: {:,}'.format(total_cust
          fontsize=16, fontweight='bold', pad=20)
plt.show()
# 6. Monthly Charges Analysis - Histogram
plt.figure(figsize=(6, 4))
churned = df[df['Churn'] == 'Yes']['MonthlyCharges']
retained = df[df['Churn'] == 'No']['MonthlyCharges']

plt.hist([churned, retained], bins=20, label=['Churned', 'Retained'],
         color=['#e74c3c', '#2ecc71'], alpha=0.7, edgecolor='black')
plt.title('Monthly Charges Distribution: Churned vs Retained', fontsize=16, fontweight='
plt.xlabel('Monthly Charges ($)', fontsize=12)
plt.ylabel('Number of Customers', fontsize=12)
plt.legend()
plt.grid(alpha=0.3)
plt.tight_layout()
plt.show()
```




Monthly Charges Distribution: Churned vs Retained
Churned
Retained
Number of Customers
0
200
400
600
800
1000
20
40
60
80
100
120
Monthly Charges ($)

In [ ]: