In [1]:

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import re,os,glob,pickle
import datetime
import matplotlib
import warnings
warnings.filterwarnings('ignore')
import yaml,logging
from datetime import timedelta
# importing "math" for mathematical operations
import math

pd.set_option('display.max_rows', 500)
pd.set_option('display.max_columns', 500)
pd.set_option('display.width', 1000)
```

In [3]:

```
exited_emp_list = pd.read_csv('Exited_Employee_Data.csv')
## Calculating Years at Company
exited_emp_list['yearsAtCompany'] = pd.to_datetime(exited_emp_list['Exit Date'])
exited_emp_list['yearsAtCompany'] = (exited_emp_list['yearsAtCompany'] /np.timede



exited_emp_list['Last Promotion Date'] = np.where(exited_emp_list['Last Promotion
exited_emp_list['YearsSinceLastPromotion'] = pd.to_datetime(exited_emp_list['Last
exited_emp_list['YearsSinceLastPromotion'] = (exited_emp_list['YearsSinceLastProm

## Calculating Age from Date of Birth
exited_emp_list['Date of Birth'] = np.where(exited_emp_list['Date of Birth'].notn
exited_emp_list['Age'] = pd.to_datetime('now') - pd.to_datetime(exited_emp_list['
exited_emp_list['Age'] = (exited_emp_list['Age'] /np.timedelta64(1,'Y')).apply(np



active_emp_list = pd.read_csv('Active_Employee_Data.csv')
## Calculating Years at Company for Active Employees
active_emp_list['yearsAtCompany'] = pd.to_datetime('now') - pd.to_datetime(exited
active_emp_list['yearsAtCompany'] = (active_emp_list['yearsAtCompany'] /np.timede

active_emp_list['Last Promotion Date'] = np.where(active_emp_list['Last Promotion
active_emp_list['YearsSinceLastPromotion'] = pd.to_datetime(exited_emp_list['Last
active_emp_list['YearsSinceLastPromotion'] = (active_emp_list['YearsSinceLastProm

## Calculating Age from Date of Birth
active_emp_list.rename(columns={'Date Of Birth':'Date of Birth'},inplace=True)
active_emp_list['Date of Birth'] = np.where(active_emp_list['Date of Birth'].notn
active_emp_list['Age'] = pd.to_datetime('now') - pd.to_datetime(active_emp_list['
active_emp_list['Age'] = (active_emp_list['Age'] /np.timedelta64(1,'Y')).apply(np

## RenameFew Columns:
exited_emp_list.rename(columns={'Experience_In_Months_x':'Experience_In_Months','
                                ,'Previous Relevant Work Experience (Y/N)_x':'Prev

#total_emp_list = pd.read_csv('Total_Employee.csv')
total_emp_list = pd.concat([exited_emp_list, active_emp_list],ignore_index = True



# ## Calculating Age from Date of Birth
# total_emp_list['Date of Birth'] = np.where(total_emp_list['Date of Birth'].notn
# total_emp_list['Age'] = pd.to_datetime('now') - pd.to_datetime(total_emp_list[
# total_emp_list['Age'] = (total_emp_list['Age'] /np.timedelta64(1,'Y')).apply(np
```

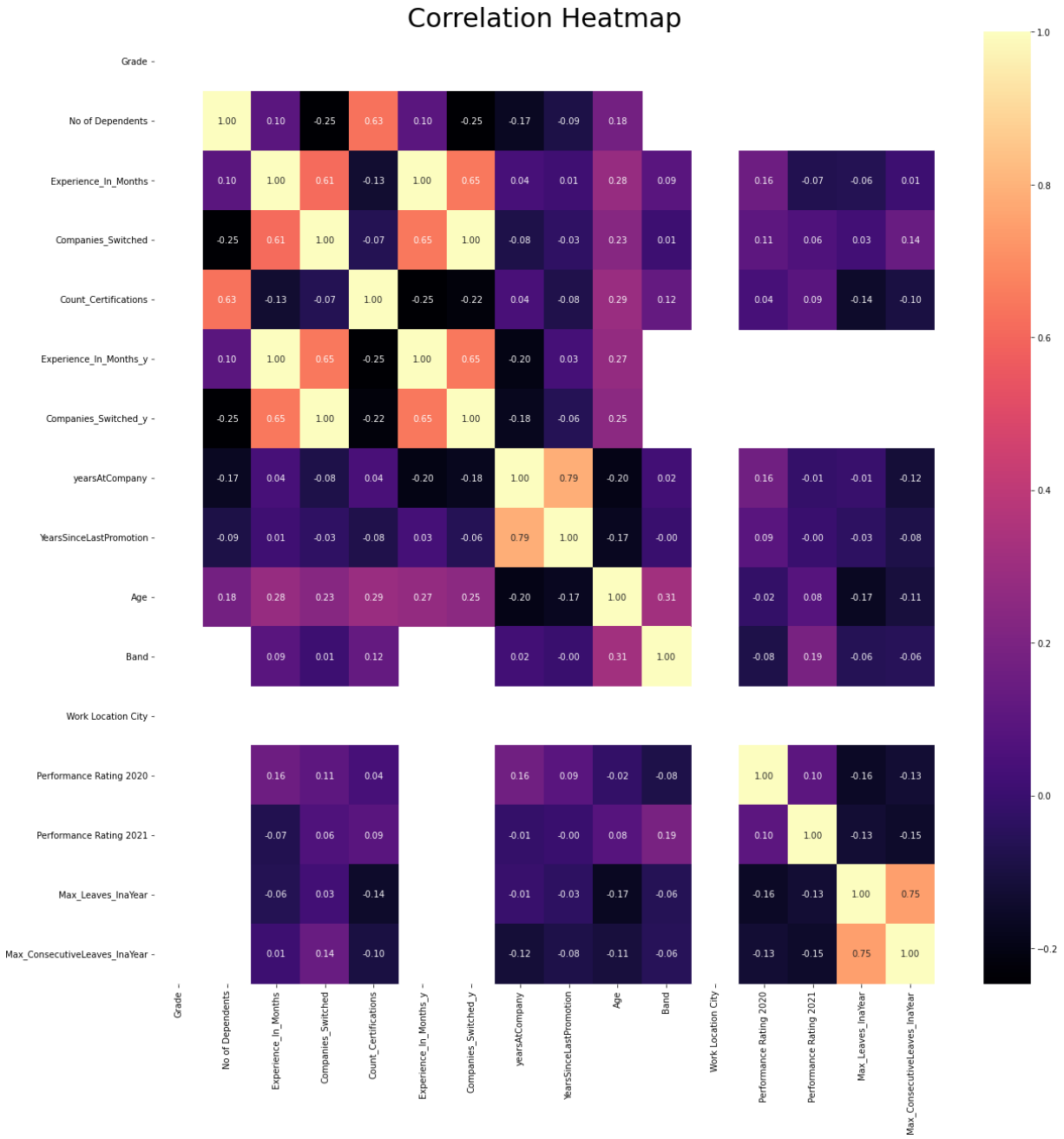
In [4]:
```
total_emp_list
```

Out[4]:

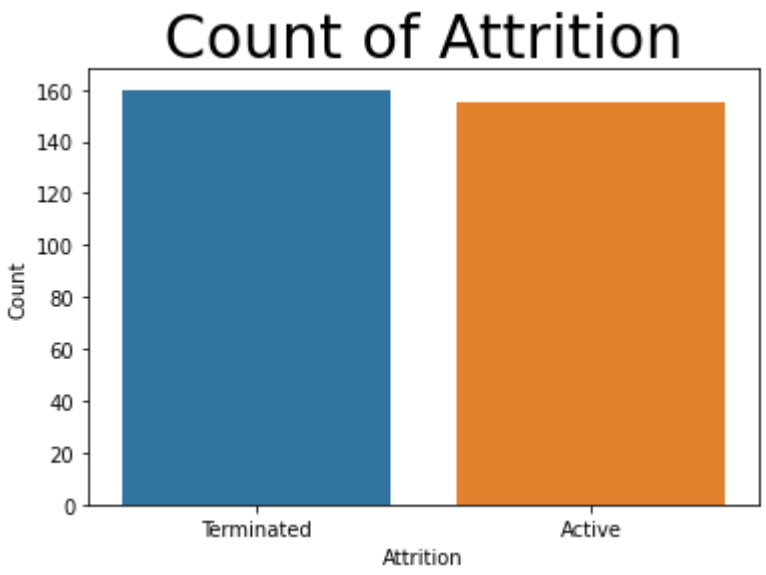
	LPN	Employee Status	Grade	Rank Name	Service Line	Sub Service Line	Region	Country/Region	Even
0	2051968	Terminated	7.0	Manager	Service Line 1	Sub Service Line 1	Region 4	India	Terminatio
1	1603404	Terminated	7.0	Manager	Service Line 1	Sub Service Line 2	Region 5	India	Terminatio
2	1611078	Terminated	7.0	Manager	Service Line 5	Sub Service Line 10	Region 4	India	Terminatio
3	2050252	Terminated	7.0	Manager	Service Line 2	Sub Service Line 3	Region 3	India	Terminatio

In [5]:
```
total_emp_list.to_csv('Total_Employee.csv',index=False)
df = total_emp_list.copy()
df = df[df['Employee Status']!='Unpaid Leave']
```

In [6]:
```
plt.figure(figsize=(20,20))
sns.heatmap(df.corr(), annot=True, fmt='.2f', cmap='magma')
plt.title('Correlation Heatmap', fontsize=30)
plt.show()
```



In [35]:
```
sns.countplot('Employee Status', data=df)
plt.title('Count of Attrition', fontsize=30)
plt.xlabel('Attrition')
plt.ylabel('Count')
plt.show()
```



In [8]:
```
categorical_cols = [feature for feature in df.columns if df[feature].dtypes=='obj
df[categorical_cols].sample(5)
```

Out[8]:

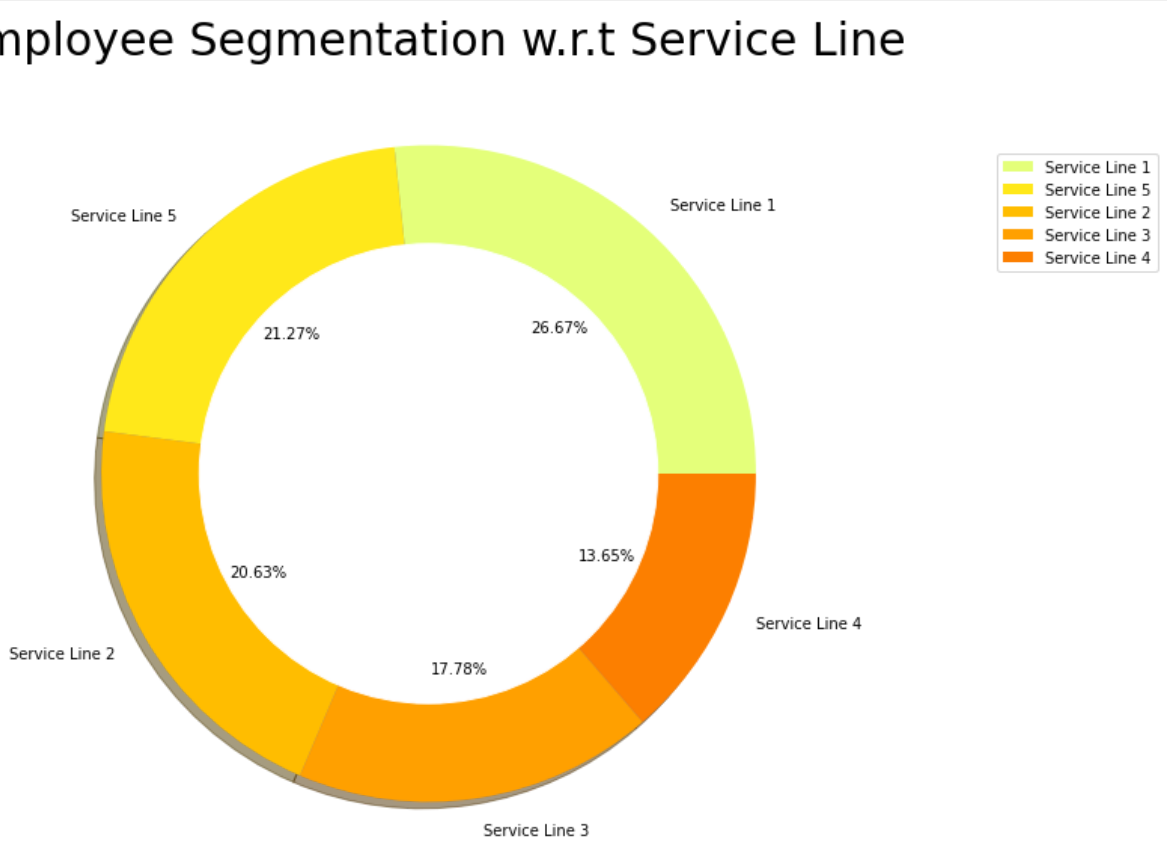
	LPN	Employee Status	Rank Name	Service Line	Sub Service Line	Region	Country/Region	Event	Even Reaso
38	2050823	Terminated	Manager	Service Line 3	Sub Service Line 7	Region 5	India	Termination	Separatic
272	3327630	Active	NaN	Service Line 2	Sub Service Line 5	NaN	India	NaN	Na
32	1610743	Terminated	Manager	Service Line 4	Sub Service Line 8	Region 2	India	Termination	Separatic
278	1324301	Active	NaN	Service Line 5	Sub Service Line 10	NaN	India	NaN	Na
67	323002	Terminated	Manager	Service Line 3	Sub Service Line 7	Region 1	India	Termination	Separatic

In [9]:
```
numerical_cols = [feature for feature in df.columns if df[feature].dtypes!='objec
df[numerical_cols].sample(5)
```

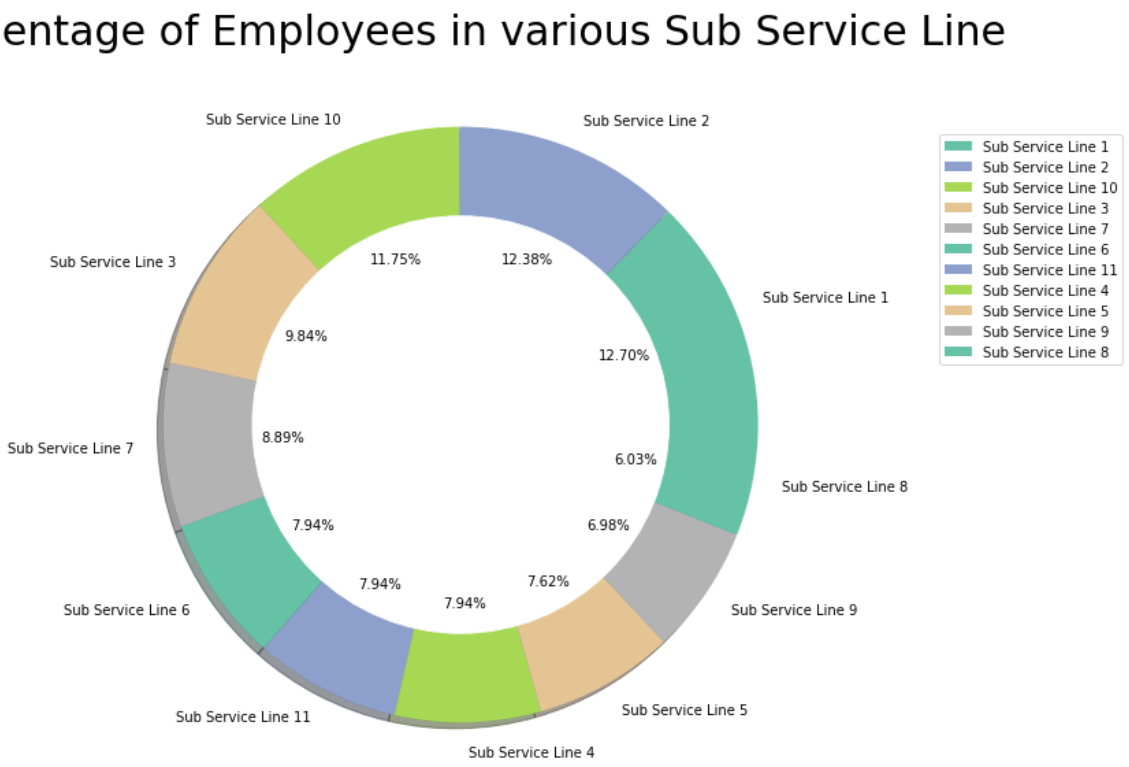
Out[9]:

	Grade	No of Dependents	Experience_In_Months	Companies_Switched	Count_Certifications	Experie
222	NaN	NaN	84.0	1.0	NaN	
267	NaN	NaN	9.0	1.0	NaN	
219	NaN	NaN	75.0	1.0	NaN	
157	7.0	NaN	178.0	3.0	NaN	
26	7.0	NaN	NaN	NaN	1.0	

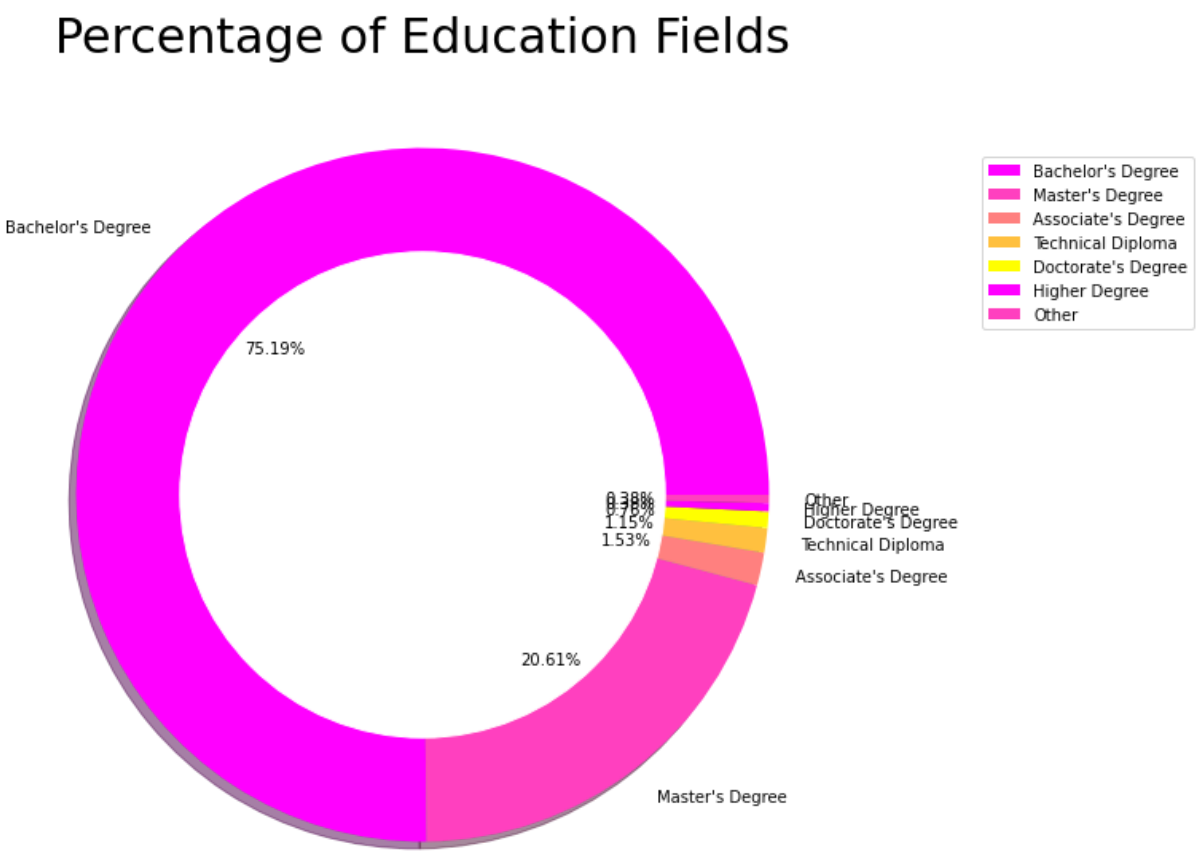
In [10]:
```
size = df['Service Line'].value_counts()
labels = df['Service Line'].unique()
colors = plt.cm.Wistia(np.linspace(0,1,5))

plt.figure(figsize=(10,10))
circle = plt.Circle((0,0), radius=0.7, color='white')
plt.pie(size, colors = colors, labels = labels, shadow = True, autopct = '%.2f%%'
p = plt.gcf()
p.gca().add_artist(circle)
plt.title('Employee Segmentation w.r.t Service Line', fontsize=30)
plt.legend(bbox_to_anchor=(0.5, 0., 0.9, 0.9));
```



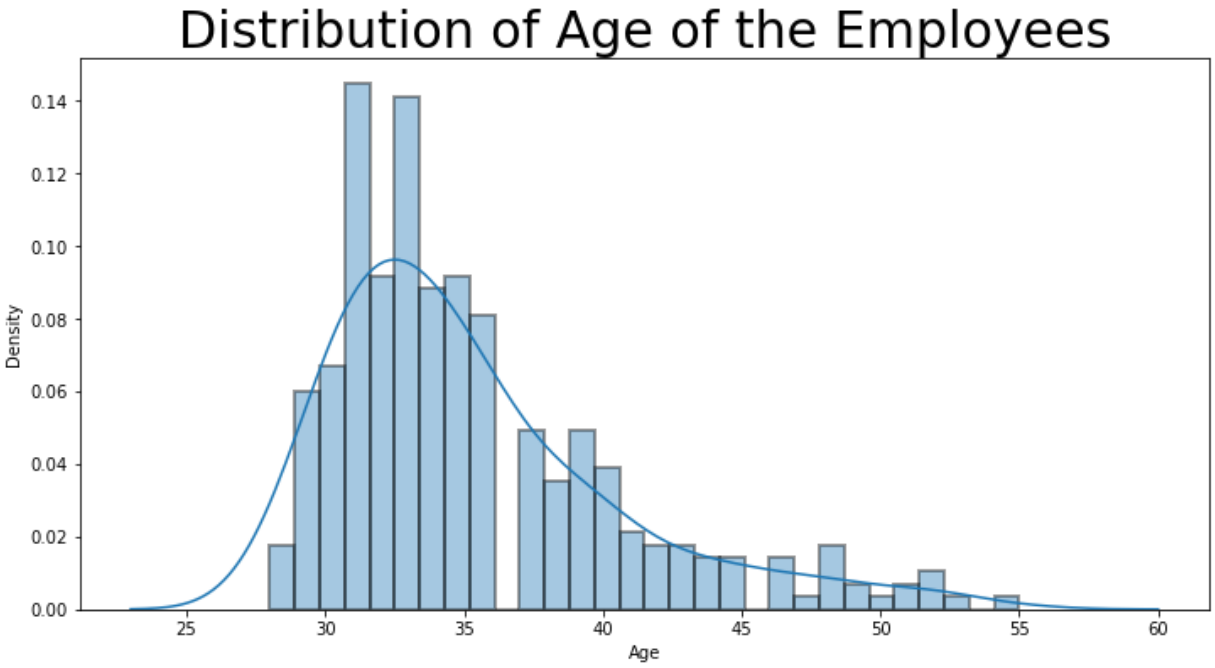
In [11]:
```
size = df['Sub Service Line'].value_counts()
labels = df['Sub Service Line'].unique()
colors = plt.cm.Set2(np.linspace(0,1,5))

plt.figure(figsize=(10,10))
circle = plt.Circle((0,0), radius=0.7, color='white')
plt.pie(size, colors = colors, labels = labels, shadow = True, autopct = '%.2f%%'
p = plt.gcf()
p.gca().add_artist(circle)
plt.title('Percentage of Employees in various Sub Service Line', fontsize=30)
plt.legend(bbox_to_anchor=(0.5, 0., 0.9, 0.9));
```



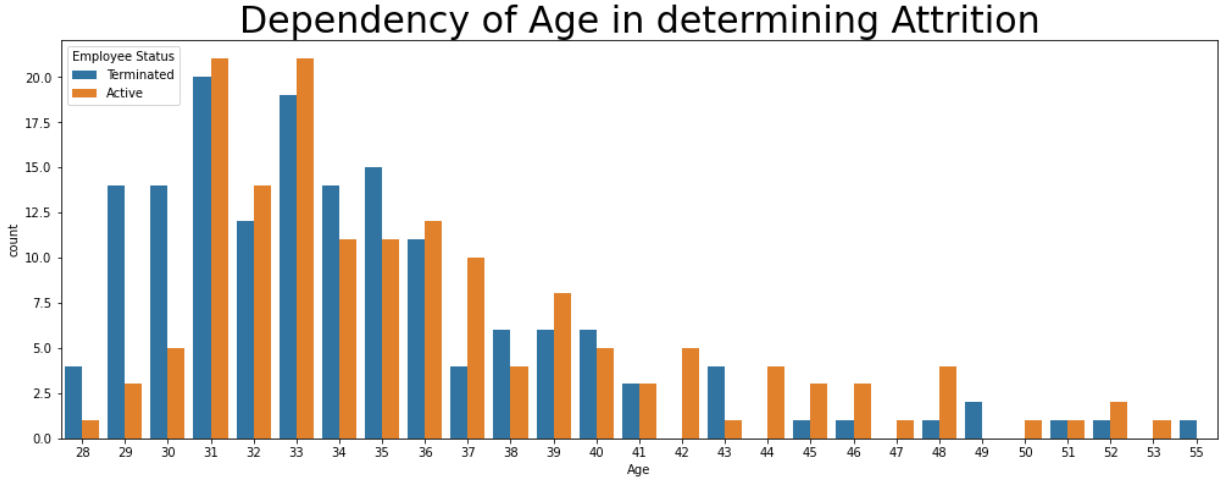
In [12]:
```
size = df['Education'].dropna().value_counts()
labels = df['Education'].dropna().unique()
colors = plt.cm.spring(np.linspace(0,1,5))

plt.figure(figsize=(10,10))
circle = plt.Circle((0,0), radius=0.7, color='white')
plt.pie(size, colors = colors, labels = labels, shadow = True, autopct = '%.2f%%'
p = plt.gcf()
p.gca().add_artist(circle)
plt.title('Percentage of Education Fields', fontsize=30)
plt.legend(bbox_to_anchor=(0.5, 0., 0.9, 0.9));
```



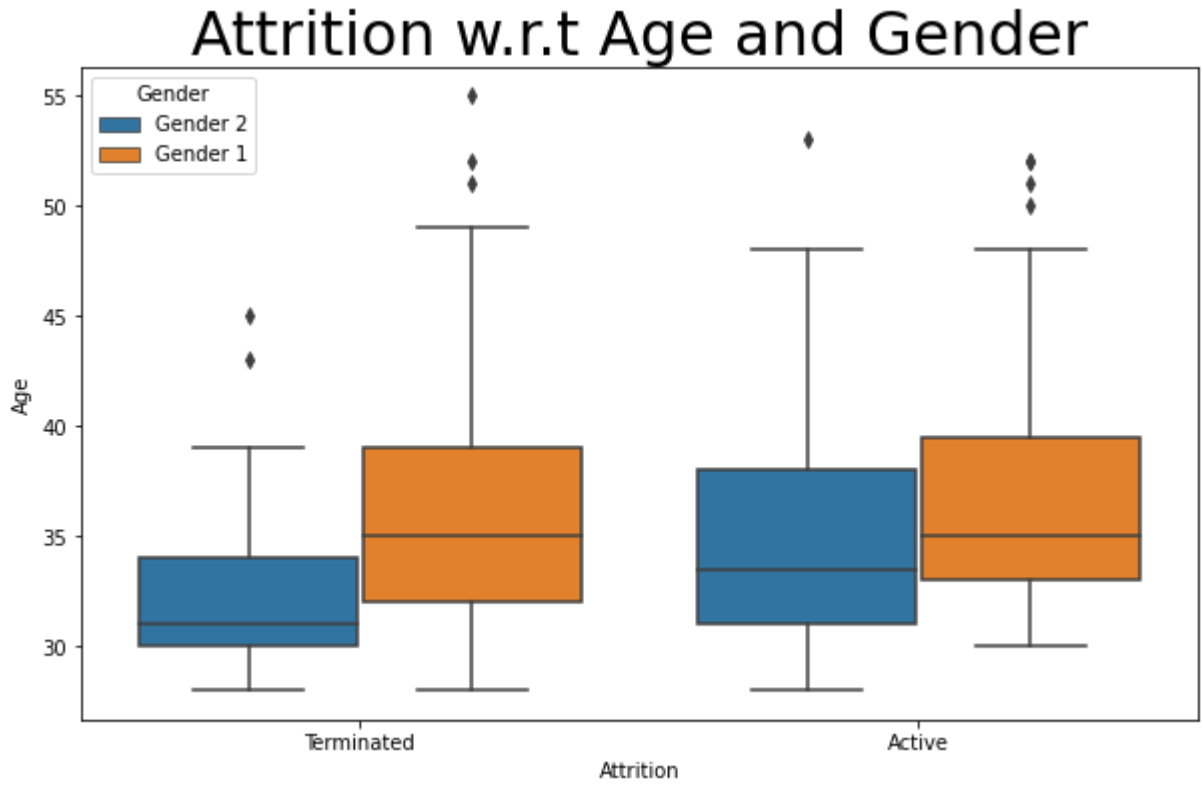
In [13]:
```
## Age =0 is showing missing Values
plt.figure(figsize=(12,6))
sns.distplot(df['Age'], bins=30, hist_kws=dict(edgecolor='black', linewidth=2))
plt.title('Distribution of Age of the Employees', fontsize=30)
plt.xlabel('Age')
plt.show()
```



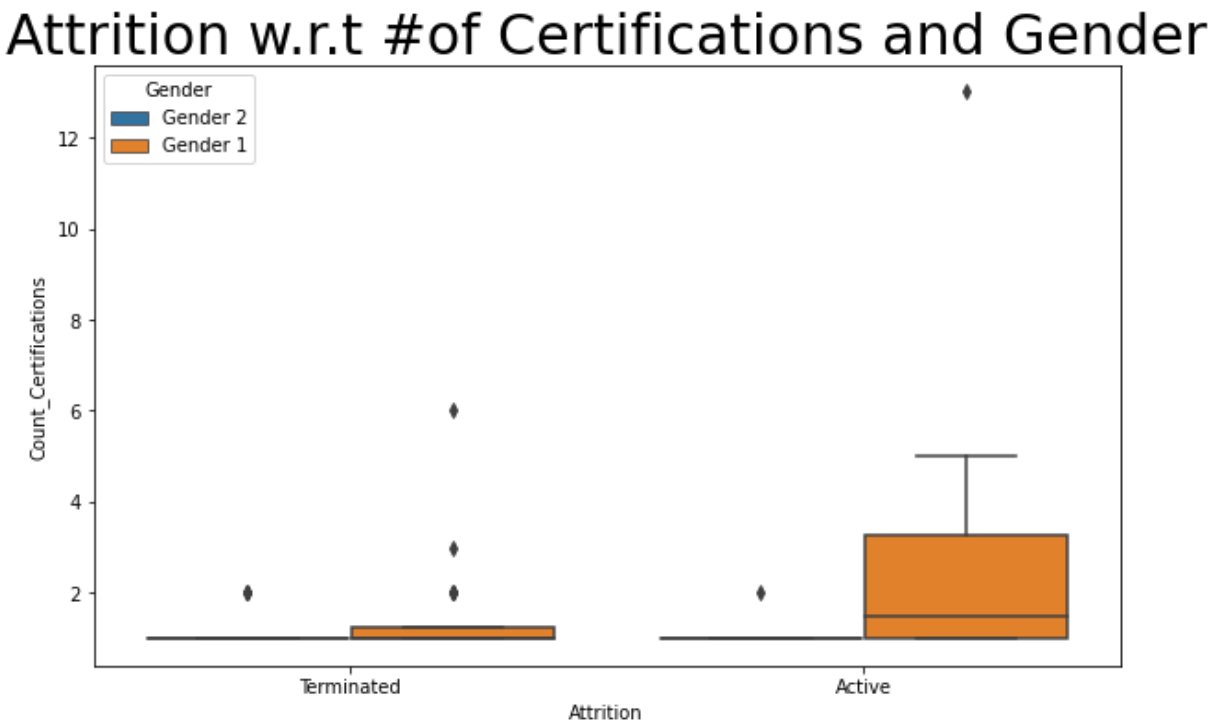
In [14]:
```
plt.figure(figsize=(17,6))
sns.countplot('Age', hue='Employee Status', data=df)
plt.title('Dependency of Age in determining Attrition', fontsize=30)
plt.xlabel('Age')
plt.show()
```



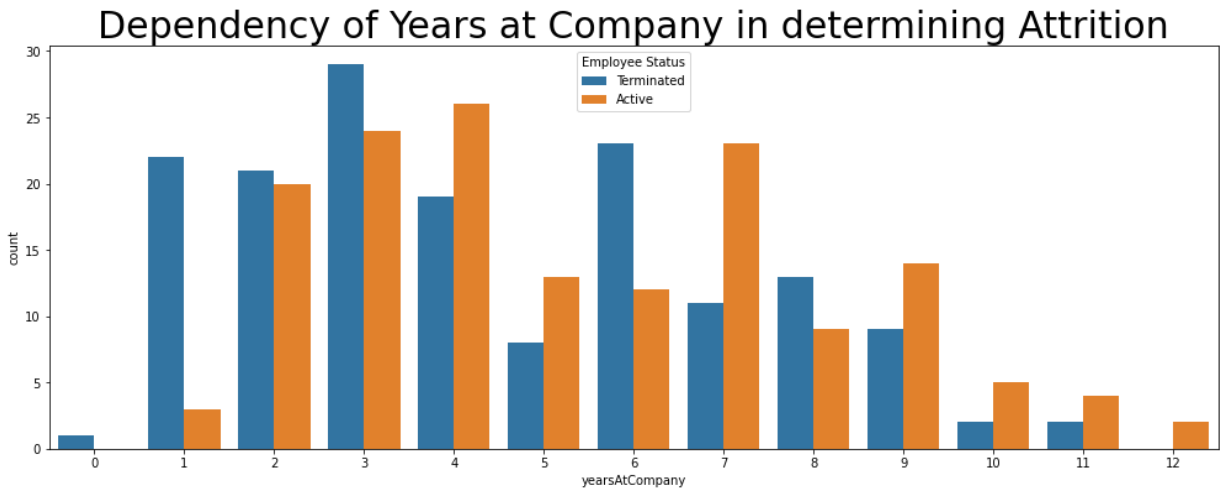
In [15]:
```
plt.figure(figsize=(10,6))
sns.boxplot('Employee Status', 'Age', hue='Gender', data=df)
plt.title('Attrition w.r.t Age and Gender', fontsize=30)
plt.xlabel('Attrition')
plt.show()
```



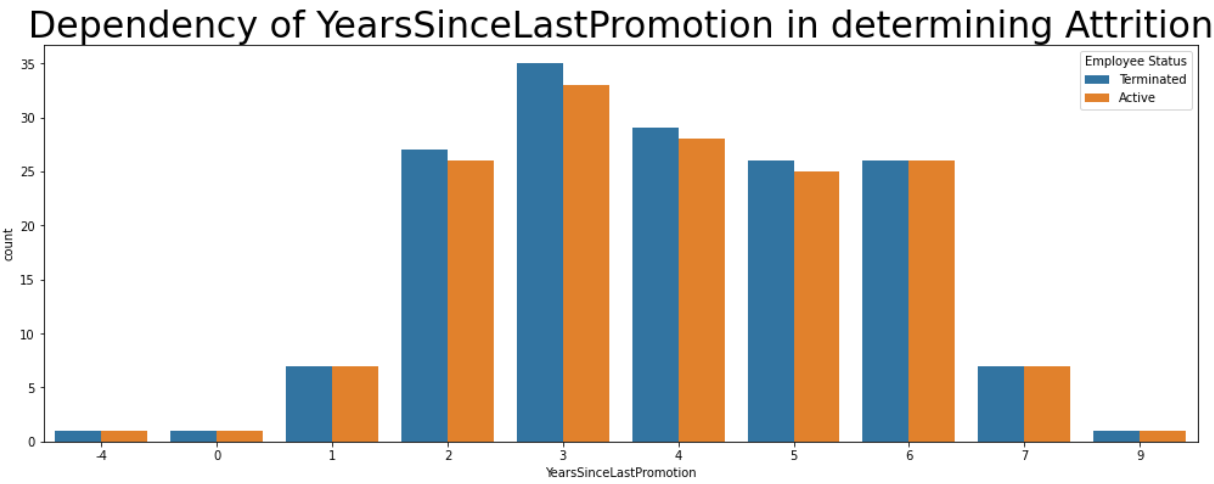
In [16]:
```
plt.figure(figsize=(10,6))
sns.boxplot('Employee Status', 'Count_Certifications', hue='Gender', data=df)
plt.title('Attrition w.r.t #of Certifications and Gender', fontsize=30)
plt.xlabel('Attrition')
plt.show()
```



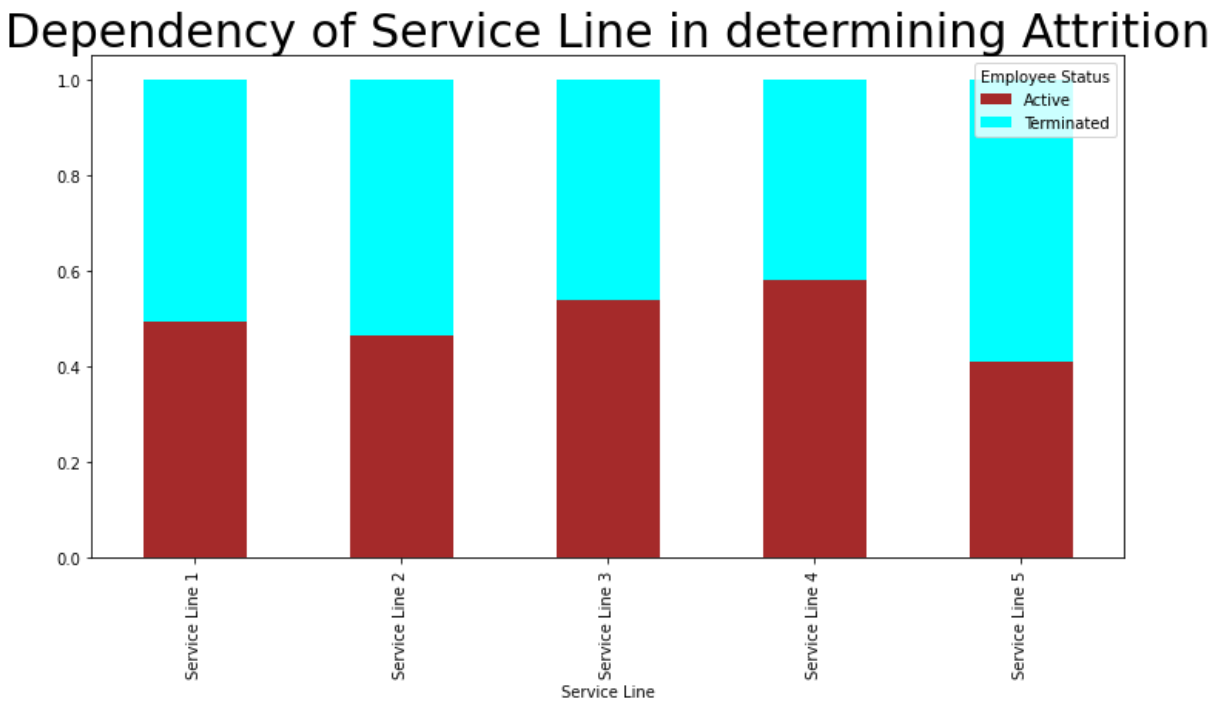
In [17]:
```
plt.figure(figsize=(17,6))
sns.countplot('yearsAtCompany', hue='Employee Status', data=df)
plt.title('Dependency of Years at Company in determining Attrition', fontsize=30)
plt.xlabel('yearsAtCompany')
plt.show()
```



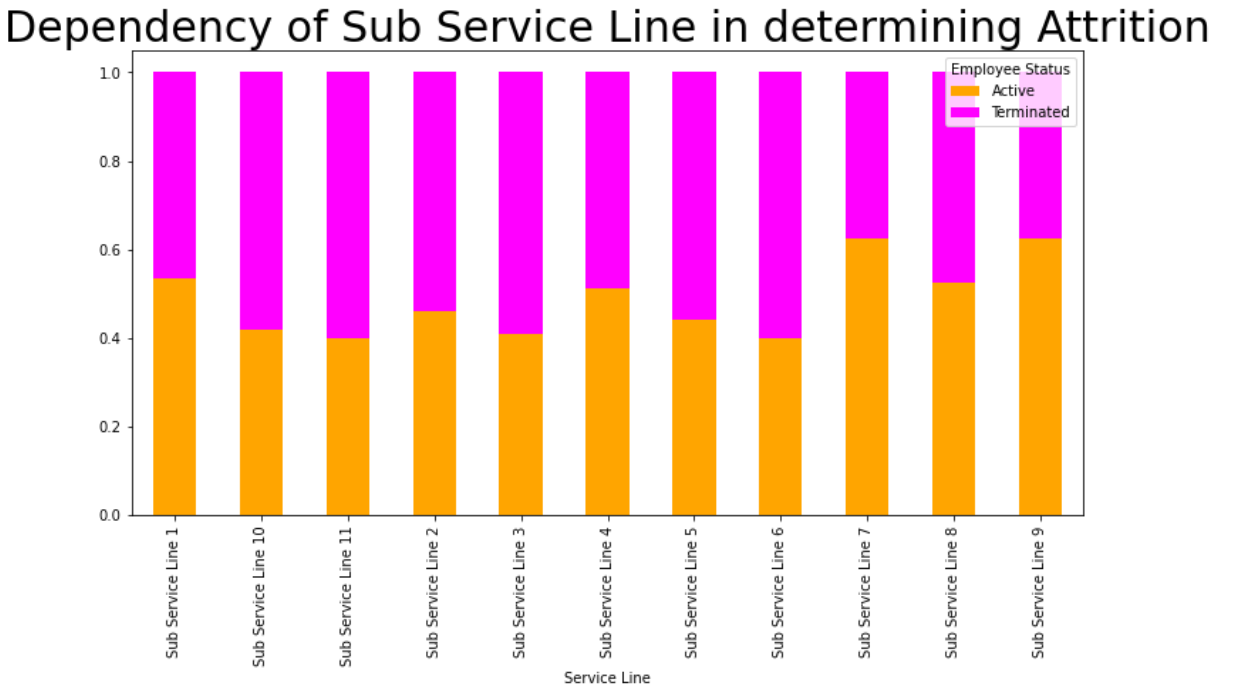
In [18]:
```
plt.figure(figsize=(17,6))
sns.countplot('YearsSinceLastPromotion', hue='Employee Status', data=df)
plt.title('Dependency of YearsSinceLastPromotion in determining Attrition', fonts
plt.xlabel('YearsSinceLastPromotion')
plt.show()
```



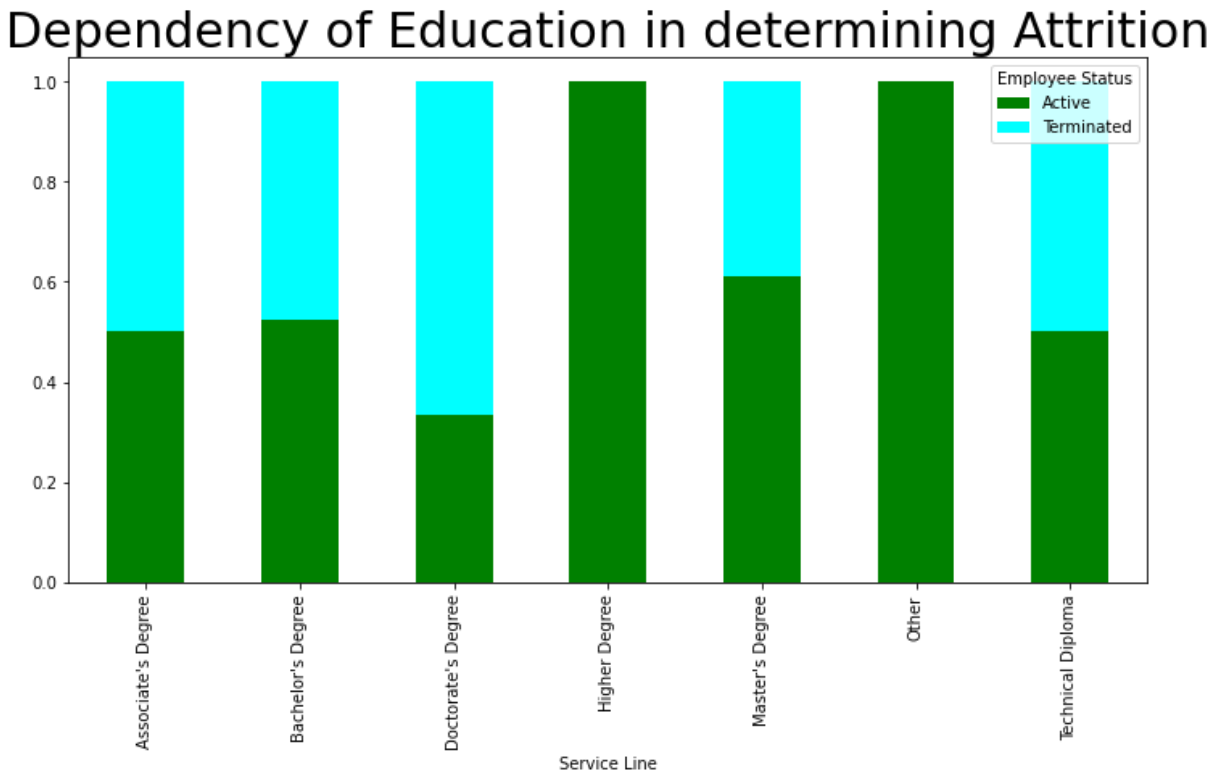
In [19]:
```
data=pd.crosstab(df['Service Line'], df['Employee Status'])
data.div(data.sum(1).astype(float), axis=0).plot(kind='bar', stacked=True, color=
                                                 figsize=(12,6))
plt.title('Dependency of Service Line in determining Attrition', fontsize=30)
plt.xlabel('Service Line')
plt.show()
```



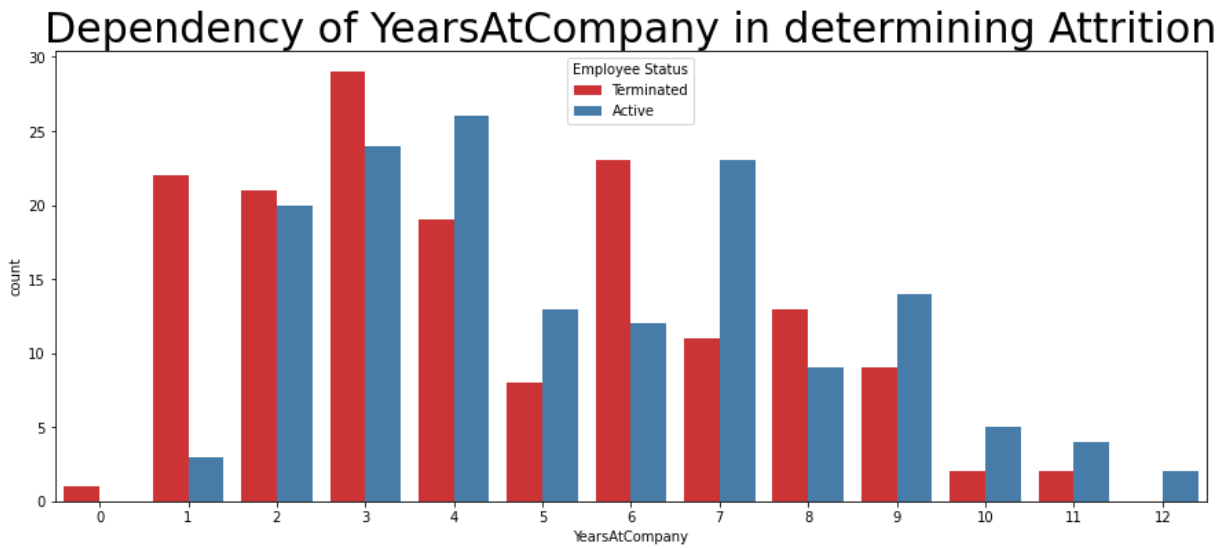
In [20]:
```
data=pd.crosstab(df['Sub Service Line'], df['Employee Status'])
data.div(data.sum(1).astype(float), axis=0).plot(kind='bar', stacked=True, color=
                                                 figsize=(12,6))
plt.title('Dependency of Sub Service Line in determining Attrition', fontsize=30)
plt.xlabel('Service Line')
plt.show()
```



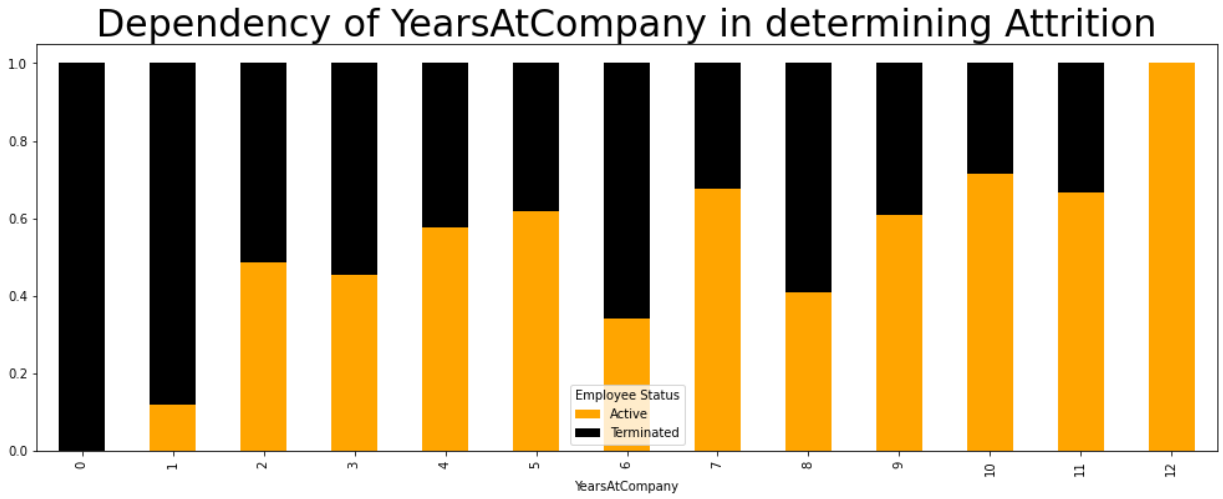
In [21]:
```
data=pd.crosstab(df['Education'], df['Employee Status'])
data.div(data.sum(1).astype(float), axis=0).plot(kind='bar', stacked=True, color=
                                                 figsize=(12,6))
plt.title('Dependency of Education in determining Attrition', fontsize=30)
plt.xlabel('Service Line')
plt.show()
```



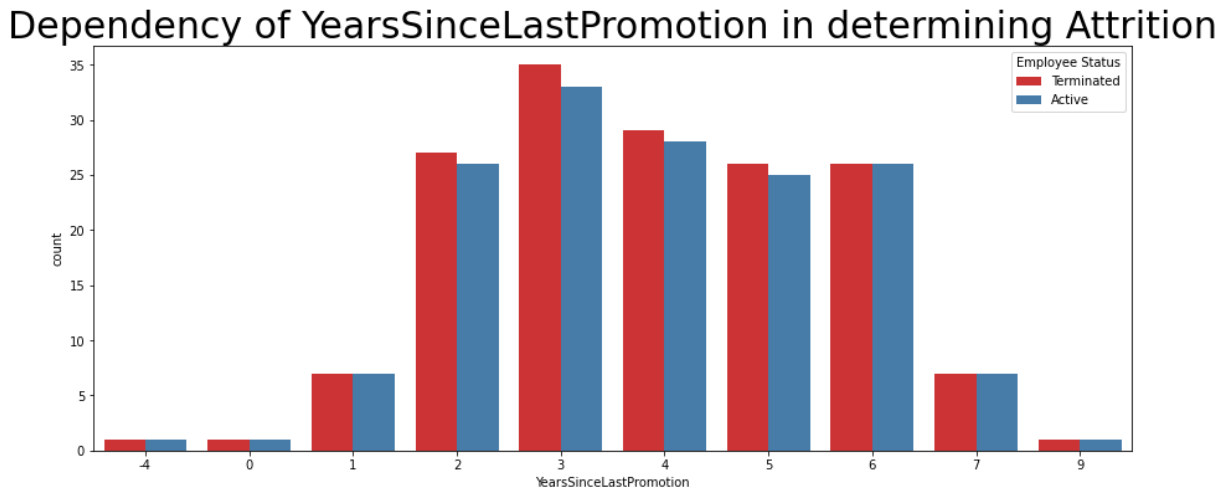
In [23]:
```
plt.figure(figsize=(15,6))
sns.countplot('yearsAtCompany', hue='Employee Status', data=df, palette='Set1')
plt.title('Dependency of YearsAtCompany in determining Attrition', fontsize=30)
plt.xlabel('YearsAtCompany')
plt.show()
```



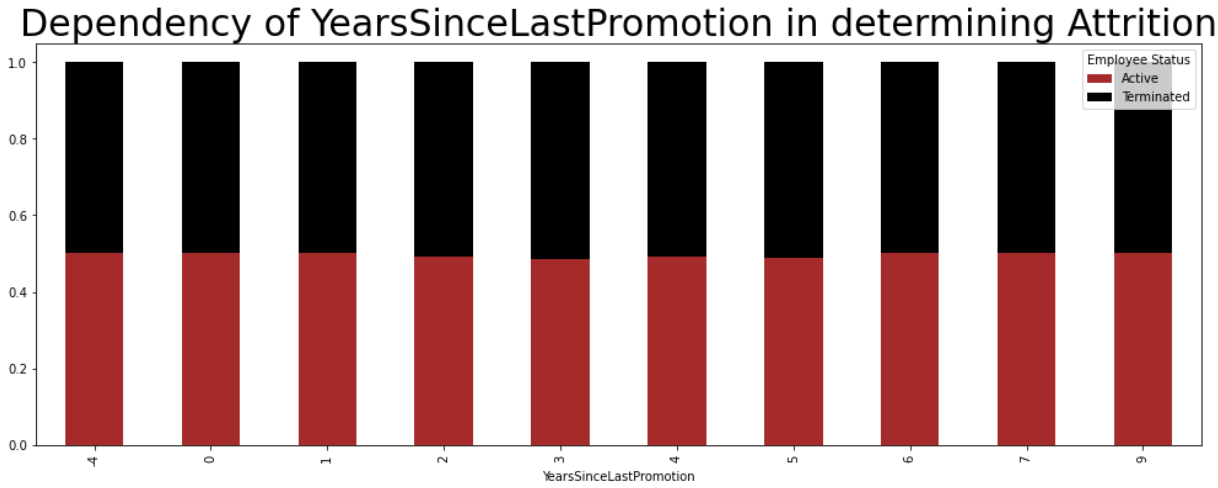
In [24]:
```
data=pd.crosstab(df['yearsAtCompany'], df['Employee Status'])
data.div(data.sum(1).astype(float), axis=0).plot(kind='bar', stacked=True, color=
                                                 figsize=(17,6))
plt.title('Dependency of YearsAtCompany in determining Attrition', fontsize=30)
plt.xlabel('YearsAtCompany')
plt.show()
```



In [26]:
```
plt.figure(figsize=(15,6))
sns.countplot('YearsSinceLastPromotion', hue='Employee Status', data=df, palette=
plt.title('Dependency of YearsSinceLastPromotion in determining Attrition', fonts
plt.xlabel('YearsSinceLastPromotion')
plt.show()
```

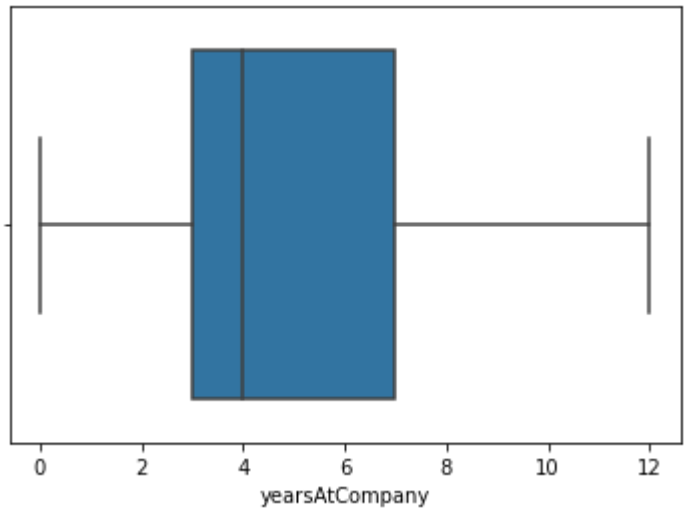


In [27]:
```
data=pd.crosstab(df['YearsSinceLastPromotion'], df['Employee Status'])
data.div(data.sum(1).astype(float), axis=0).plot(kind='bar', stacked=True, color=
                                                 figsize=(17,6))
plt.title('Dependency of YearsSinceLastPromotion in determining Attrition', fonts
plt.xlabel('YearsSinceLastPromotion')
plt.show()
```



In [29]:
```
sns.boxplot(df["yearsAtCompany"])
```

Out[29]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x22d083c91c8>
```

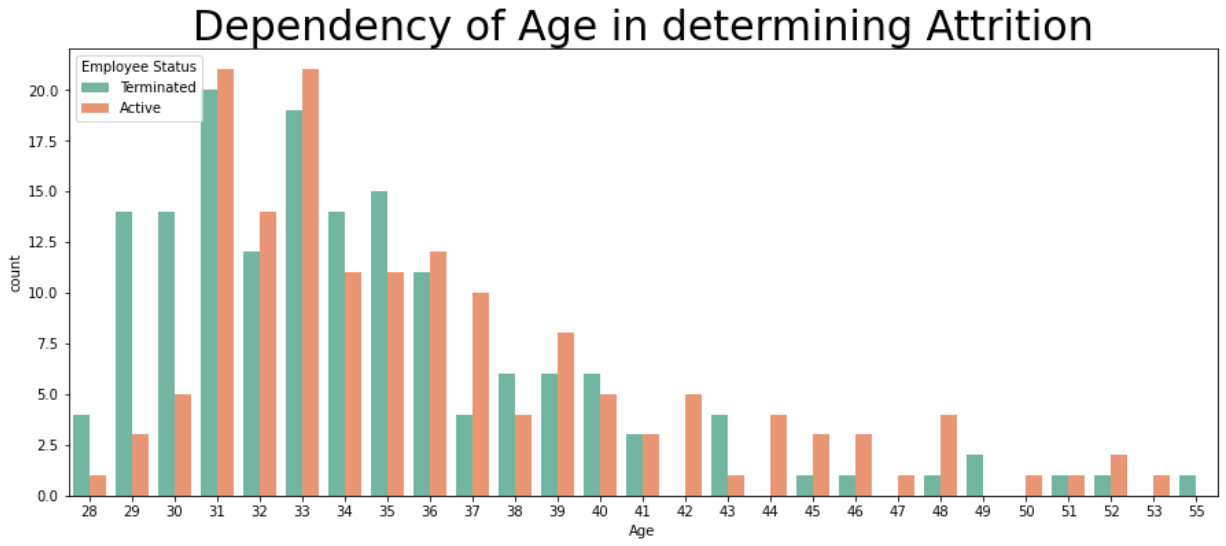


% Attrition

In [31]:
```
percentage = df["Employee Status"].value_counts()
retention = percentage[1]*100/(percentage[1]+percentage[0])
print("The percentage of people leaving organization as per dataset :",round(rete
```

```
The percentage of people leaving organization as per dataset : 49.21 %
```

In [38]:
```
plt.figure(figsize=(15,6))
sns.countplot('Age', hue='Employee Status', data=df, palette='Set2')
plt.title('Dependency of Age in determining Attrition', fontsize=30)
plt.xlabel('Age')
plt.show()
```



In [ ]: