Project Name -

Project Type - EDA/Regression/Classification/Unsupervised

Contribution - Individual: SOMNATH KAYAL

Team Member 1 -

Team Member 2 -

Team Member 3 -

Team Member 4 -

Project Summary -

Project Objective:

To analyze a mental health survey dataset in order to:

Understand the prevalence of mental health issues in the workplace.

Identify geographic, demographic, and workplace predictors of mental health challenges.

Explore attitudes and stigma related to discussing mental health at work.

Evaluate the availability of resources and employer support across different regions and companies.

Key Fields:

Demographics: Gender, Country, State, self_employed, Age

Mental Health Info: treatment, family_history, work_interfere

Workplace Context: no_employees, remote_work, tech_company

Support Resources: benefits, seek_help, care_options, wellness_program, leave, anonymity

Attitudes & Stigma: mental_health_consequence, phys_health_consequence, mental_vs_physical, obs_consequence, coworkers, supervisors, mental_health_interview

Data Cleaning & Preparation:

Open-ended gender entries like "Malr", "Cis Male", "femail", "nonbinary", "androgyne" were normalized using .replace() and .str.lower().strip() to create consistent gender categories:

Male Female Non-Binary/Other

Datetime Parsing:

Converted Timestamp to datetime and extracted Year-Month for temporal analysis

Categorical Binning:

Custom bins for age analysis:

GitHub Link -

Provide your GitHub Link here: https://github.com/SomInd101/Mental_health_Survey/tree/main

Problem Statement

Write Problem Statement Here.

1 - How does the frequency of mental health illness and attitudes towards mental health vary by geographic location?

2 - What are the strongest predictors of mental health illness or certain attitudes towards mental health in the workplace?

Define Your Business Objective?

Business Objective:

1. Identify Risk Factors for Mental Health Issues:

Understand who is most likely to struggle with mental health based on demographics (e.g., gender, age, country), work conditions (e.g., remote work, company size), and personal history (e.g., family history).

Predict which employee segments are more likely to require support or treatment.

2. Measure Attitudes and Stigma Toward Mental Health:

Assess whether employees feel safe discussing mental health with coworkers or supervisors.

Identify regions or job types where stigma around mental illness is still strong.

3. Drive Policy and Program Improvements: Help HR leaders and executives design better mental health policies by highlighting gaps in existing resources and support.

Recommend where to invest in training, awareness, and care options.

Long-Term Impact:

By meeting these objectives, organizations can:

Reduce burnout, absenteeism, and turnover

Improve employee well-being and productivity

Build a culture of mental health support and openness

General Guidelines : -

1. Well-structured, formatted, and commented code is required.
2. Exception Handling, Production Grade Code & Deployment Ready Code will be a plus. Those students will be awarded some additional credits.

   The additional credits will have advantages over other students during Star Student selection.

```
    [ Note: - Deployment Ready Code is defined as, the
whole .ipynb notebook should be executable in one go
               without a single error logged. ]
```

3. Each and every logic should have proper comments.
4. You may add as many number of charts you want. Make Sure for each and every chart the following format should be answered.

```
# Chart visualization code
```

- Why did you pick the specific chart?
- What is/are the insight(s) found from the chart?
- Will the gained insights help creating a positive business impact? Are there any insights that lead to negative growth? Justify with specific reason.

1. You have to create at least 20 logical & meaningful charts having important insights.

[Hints : - Do the Vizualization in a structured way while following "UBM" Rule.

U - Univariate Analysis,

B - Bivariate Analysis (Numerical - Categorical, Numerical - Numerical, Categorical - Categorical)

M - Multivariate Analysis]

Let's Begin !

1. Know Your Data

Import Libraries

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

Dataset Loading

```
# Load Dataset
file_path= r"D:\Data Analytics\Intership\Labmentix\Mental Health
survey"

df= pd.read_csv(fr"{file_path}\survey.csv")
```

Dataset First View

```
# Dataset First Look
df.head()
```

```
             Timestamp  Age  Gender          Country state
self_employed  \
0  2014-08-27 11:29:31   37  Female   United States    IL
NaN
1  2014-08-27 11:29:37   44       M   United States    IN
NaN
2  2014-08-27 11:29:44   32    Male          Canada   NaN
NaN
3  2014-08-27 11:29:46   31    Male  United Kingdom   NaN
NaN
4  2014-08-27 11:30:22   31    Male   United States    TX
NaN

  family_history treatment work_interfere    no_employees  ...  \
0             No       Yes          Often            6-25  ...
1             No        No         Rarely  More than 1000  ...
2             No        No         Rarely           6-25  ...
3            Yes       Yes          Often         26-100  ...
4             No        No          Never        100-500  ...

                 leave mental_health_consequence
phys_health_consequence  \
0       Somewhat easy                         No
No
```

```
1          Don't know                          Maybe
No
2  Somewhat difficult                            No
No
3  Somewhat difficult                           Yes
Yes
4          Don't know                            No
No

      coworkers supervisor mental_health_interview
phys_health_interview  \
0  Some of them        Yes                       No
Maybe
1            No         No                       No
No
2           Yes        Yes                      Yes
Yes
3  Some of them         No                     Maybe
Maybe
4  Some of them        Yes                      Yes
Yes

  mental_vs_physical obs_consequence comments
0                Yes              No      NaN
1         Don't know              No      NaN
2                 No              No      NaN
3                 No             Yes      NaN
4         Don't know              No      NaN

[5 rows x 27 columns]
```

Dataset Rows & Columns count

```
# Dataset Rows & Columns count
df.shape

(1259, 27)
```

Dataset Information

```
# Dataset Info
df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1259 entries, 0 to 1258
Data columns (total 27 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   Timestamp                  1259 non-null   object
 1   Age                        1259 non-null   int64
```

```
 2   Gender                     1259 non-null   object
 3   Country                    1259 non-null   object
 4   state                      744 non-null    object
 5   self_employed              1241 non-null   object
 6   family_history             1259 non-null   object
 7   treatment                  1259 non-null   object
 8   work_interfere             995 non-null    object
 9   no_employees               1259 non-null   object
 10  remote_work                1259 non-null   object
 11  tech_company               1259 non-null   object
 12  benefits                   1259 non-null   object
 13  care_options               1259 non-null   object
 14  wellness_program           1259 non-null   object
 15  seek_help                  1259 non-null   object
 16  anonymity                  1259 non-null   object
 17  leave                      1259 non-null   object
 18  mental_health_consequence  1259 non-null   object
 19  phys_health_consequence    1259 non-null   object
 20  coworkers                  1259 non-null   object
 21  supervisor                 1259 non-null   object
 22  mental_health_interview    1259 non-null   object
 23  phys_health_interview      1259 non-null   object
 24  mental_vs_physical         1259 non-null   object
 25  obs_consequence            1259 non-null   object
 26  comments                   164 non-null    object
dtypes: int64(1), object(26)
memory usage: 265.7+ KB
```

Duplicate Values

```
# Dataset Duplicate Value Count
df.duplicated().sum()
```

```
np.int64(0)
```

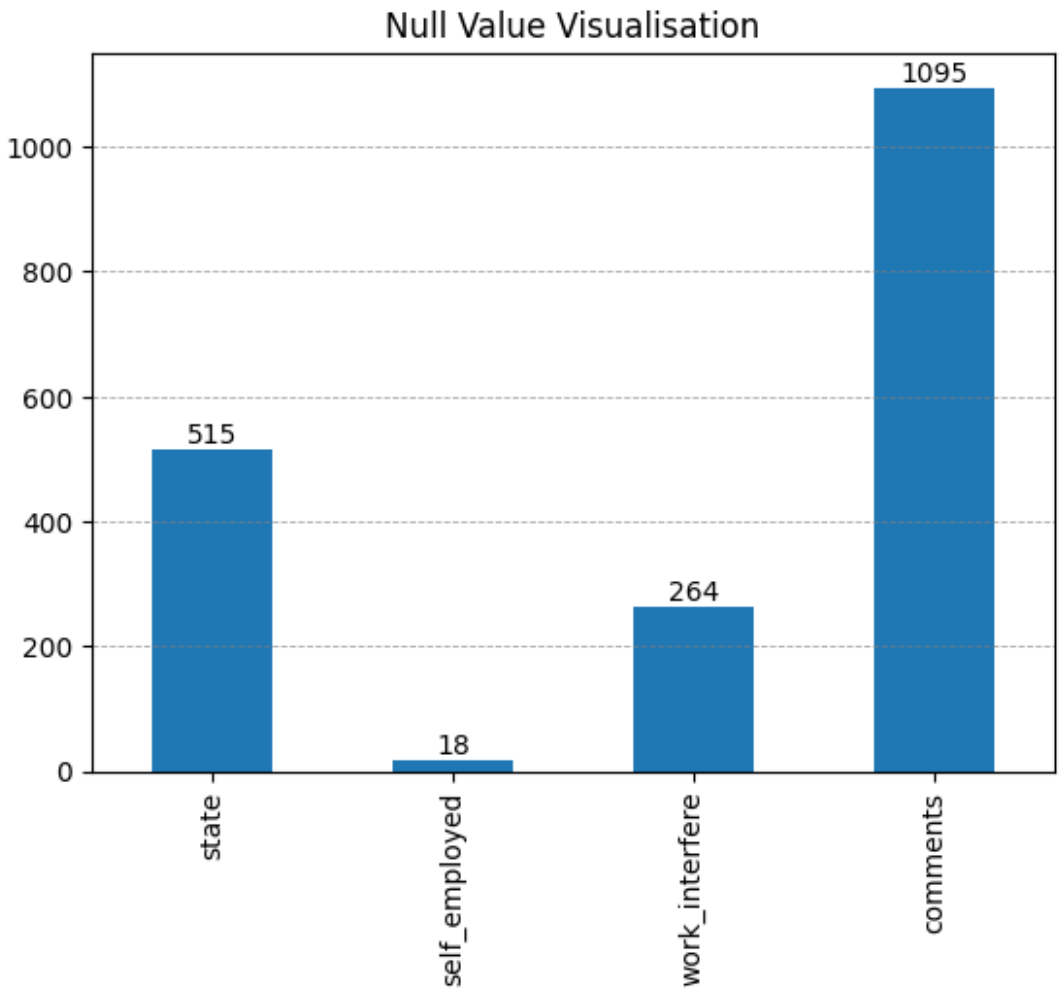
Missing Values/Null Values

```
# Missing Values/Null Values Count
df.isnull().sum()
```

```
Timestamp                      0
Age                            0
Gender                         0
Country                        0
state                        515
self_employed                 18
family_history                 0
treatment                      0
work_interfere               264
no_employees                   0
```

```
remote_work                     0
tech_company                    0
benefits                        0
care_options                    0
wellness_program                0
seek_help                       0
anonymity                       0
leave                           0
mental_health_consequence       0
phys_health_consequence         0
coworkers                       0
supervisor                      0
mental_health_interview         0
phys_health_interview           0
mental_vs_physical              0
obs_consequence                 0
comments                     1095
dtype: int64
```

```
# Visualizing the missing values
df_null = df.isnull().sum()
df_null=df_null[df_null>0]

df_nullV= df_null.plot.bar()
plt.title('Null Value Visualisation')
plt.grid(
    visible = True,
    axis ='y',
    color='gray',
    linestyle='--',
    linewidth=0.6,
    alpha =0.7)

for cols in df_nullV.containers:
    df_nullV.bar_label(cols, label_type ='edge' )
```




```
# Show rows with any null values
df[df.isnull().any(axis =1)]
[['state','self_employed','work_interfere','comments']].head(10)
```

	state	self_employed	work_interfere	comments
0	IL	NaN	Often	NaN
1	IN	NaN	Rarely	NaN
2	NaN	NaN	Rarely	NaN
3	NaN	NaN	Often	NaN
4	TX	NaN	Never	NaN
5	TN	NaN	Sometimes	NaN
6	MI	NaN	Sometimes	NaN
7	NaN	NaN	Never	NaN
8	IL	NaN	Sometimes	NaN
9	NaN	NaN	Never	NaN

```
#
sns.heatmap(df[['state','self_employed','work_interfere','comments']].
```

```
isnull(), cmap='coolwarm', fmt='.2g', yticklabels= False )
# df.isnull().any(axis=1).sum()
```

What did you know about your dataset?

The* "timestamp" columns are in object for need to extract date and time separately and create month/ year wise data column for operation

Only in the US, States are given no other states avilable for rest of the country

1259 rows × 27 columns presented in the data set

Null values in "state : 515" Null values in "work_interfere" 264" Null values in "self_employed" 18 Null values in "comments" 1095

Null values percentage present in the dataset: 5.56%

2. Understanding Your Variables

```
# Dataset Columns
df.columns
```

```
Index(['Timestamp', 'Age', 'Gender', 'Country', 'state',
'self_employed',
       'family_history', 'treatment', 'work_interfere',
'no_employees',
       'remote_work', 'tech_company', 'benefits', 'care_options',
       'wellness_program', 'seek_help', 'anonymity', 'leave',
       'mental_health_consequence', 'phys_health_consequence',
'coworkers',
       'supervisor', 'mental_health_interview',
'phys_health_interview',
       'mental_vs_physical', 'obs_consequence', 'comments'],
      dtype='object')
```

```
# Dataset Describe
df.describe()
```

```
                Age
count  1.259000e+03
mean   7.942815e+07
std    2.818299e+09
min   -1.726000e+03
25%    2.700000e+01
50%    3.100000e+01
75%    3.600000e+01
max    1.000000e+11
```

Variables Description

Timestamp: Date survey was filled

Gender: Gender identity of surveyed person

Country: Country of residence

State: US residents only

self_employed: Are you self-employed?

family history: Do you have a family history of mental illness?

treatment: Have you sought treatment for a mental health condition?

work_interfere: If you have a mental health condition, do you feel that it interferes with your work?

no_employees: How many employees does your company or organization have?

remote_work: Do you work remotely (outside of an office) at least 50% of the time?

tech_company: Is your employer primarily a tech company/organization?

benefits: Does your employer provide mental health benefits?

care_options: Do you know the options for mental health care your employer provides?

wellness_program: Has your employer ever discussed mental health as part of an employee wellness program?

seek help: Does your employer provide resources to learn more about mental health issues and how to seek help?

anonymity: Is your anonymity protected if you choose to take advantage of mental health or substance abuse treatment resources?

leave: How easy is it for you to take medical leave for a mental health condition?

mental_health_consequence: Do you think that discussing a mental health issue with your employer would have negative consequences?

phys_health_consequence: Do you think that discussing a physical health issue with your employer would have negative consequences?

coworkers: Would you be willing to discuss a mental health issue with your coworkers?

supervisors: Would you be willing to discuss a mental health issue with your direct supervisor(s)?

mental_health_interview: Would you bring up a mental health issue with a potential employer in an interview?

mental_vs_physical: Do you feel that your employer takes mental health as seriously as physical health?

obs_consequence: Have you heard of or observed negative consequences for coworkers with mental health conditions in your workplace?

comments: Any additional notes or comments?

Check Unique Values for each variable.

```
# Check Unique Values for each variable.
df['Gender'].unique()
```

```
array(['Female', 'M', 'Male', 'male', 'female', 'm', 'Male-ish',
'maile',
       'Trans-female', 'Cis Female', 'F', 'something kinda male?',
       'Cis Male', 'Woman', 'f', 'Mal', 'Male (CIS)',
'queer/she/they',
       'non-binary', 'Femake', 'woman', 'Make', 'Nah', 'All', 'Enby',
       'fluid', 'Genderqueer', 'Female ', 'Androgyne', 'Agender',
       'cis-female/femme', 'Guy (-ish) ^_^', 'male leaning
androgynous',
       'Male ', 'Man', 'Trans woman', 'msle', 'Neuter', 'Female
(trans)',
       'queer', 'Female (cis)', 'Mail', 'cis male', 'A little about
you',
       'Malr', 'p', 'femail', 'Cis Man',
       'ostensibly male, unsure what that really means'],
dtype=object)
```

3. ***Data Wrangling***

```
def MH_pivote(dfP, ColsName, ColsIndex, ColsValue):
    """
    df, ColsName, ColsIndex, ColsValue
    """
    dfP = dfP.pivot(columns= ColsName, index= ColsIndex, values =
ColsValue).reset_index(drop = False)
    dfP.head()
    return dfP

def bar_function (dfB, colsName1, colsName2, graphKind, wt,ht,
grphTitle, legnTitle, grdAxix):

    """
    Description: Function for implementing bar/barh graph
    dfB: Dataframe Name, VarBar = Varible for assigning the barplot,
    colsName1= x axis column name,
    colsName2= y axis column name (int) | colsName2 = 'col1' or
['col1', 'col2',..]
    graphKind: Which type of graph 'bar/barh/ scatter'
    wt: figsize width (int)- 7,6,4
    ht: figsize height (int)- 4, 5,3
    legnTitle: Legend title = 'My title'
    grdAxix: Grid Axis 'x' or 'y'
```

```
    """
    VarBar = dfB.plot(kind = graphKind, x= colsName1, y= colsName2,
figsize =(wt,ht))
    plt.title(grphTitle)
    plt.legend(title = legnTitle, bbox_to_anchor = (1.05, 1), loc =
'upper left')

    # if isinstance(colsName2, list) and len(colsName2)>2:
    #     plt.legend(title = legnTitle, bbox_to_anchor = (1.05, 1),
loc = 'upper left')
    # else:
    #     plt.legend().remove()

    plt.grid(visible= True, axis= grdAxix , color= 'gray', linestyle=
'--', linewidth= 0.6)

    for cols in VarBar.containers:
        VarBar.bar_label(cols, label_type= 'edge')
        return VarBar

    return dfB
```

Data Wrangling Code

```
# Write your code to make your dataset analysis ready.
#df_WS is our main working dataset
df_WS = df.copy()
```

Data Cleaning:

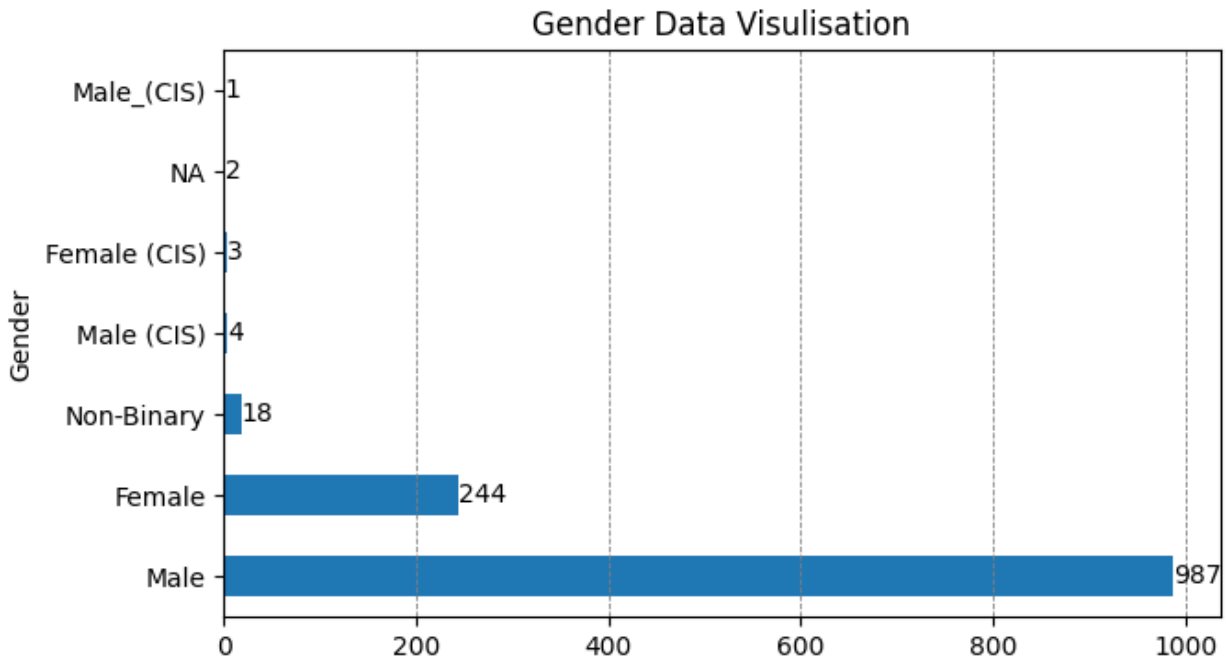
Working on Gender Column

```
dfGenM= {'maile':'Male','Mal':'Male','Man':'Male','M':'Male','Guy (-
ish) ^_^':'Male',
         'Malr':'Male','Male-
ish':'Male','m':'Male','msle':'Male','Make':'Male','Mail':'Male',
'male':'Male', 'Male ':'Male',

'female':'Female','F':'Female','f':'Female','Woman':'Female','woman':'
Female','Female':'Female','Female ':'Female',

'Femake':'Female','queer/she/they':'Female','femail':'Female',
         'Cis Male':'Male (CIS)','Cis Male':'Male (CIS)','Cis
Man':'Male (CIS)','cis male':'Male (CIS)', 'Male (CIS)':'Male_(CIS)',
         'Cis Female':'Female (CIS)','cis-female/femme':'Female
(CIS)','Female (cis)':'Female (CIS)','Female (CIS)':'Female_(CIS)',
        'something kinda male?':'Non-Binary','non-binary':'Non-
Binary','Nah':'Non-Binary','All':'Non-Binary', 'male leaning
androgynous':'Non-Binary',
```

```
         'Genderqueer':'Non-Binary','fluid':'Non-Binary','Enby':'Non-
Binary','Agender':'Non-Binary','Neuter':'Non-Binary','queer':'Non-
Binary',
         'ostensibly male, unsure what that really means':'Non-
Binary','queer/she/they':'Non-Binary','Androgyne':'Non-Binary',
         'A little about you':'NA','p':'NA', 'Female (trans)':'Non-
Binary','Trans-female':'Non-Binary','Trans woman':'Non-Binary'}

df_WS['Gender']= df_WS['Gender'].map(lambda x: dfGenM.get(x,x))

# .applymap(), map(): This method applies a function that accepts and
returns a scalarto every element of a DataFrame.
#We can't apply ".applymap()" for series till now

df_WS['Gender'].value_counts()
```

```
Gender
Male            987
Female          244
Non-Binary       18
Male (CIS)        4
Female (CIS)      3
NA                2
Male_(CIS)        1
Name: count, dtype: int64
```

```
df_WS['Gender']= df_WS['Gender'].astype('str')

df_WSGenV= df_WS['Gender'].value_counts().plot.barh(figsize =(7,4))
plt.title('Gender Data Visulisation')
plt.grid(
    visible = True,
    axis= 'x',
    color ='gray',
    linestyle='--',
    linewidth=0.6
)

for cols in df_WSGenV.containers:
    df_WSGenV.bar_label(cols, label_type = 'edge')
```

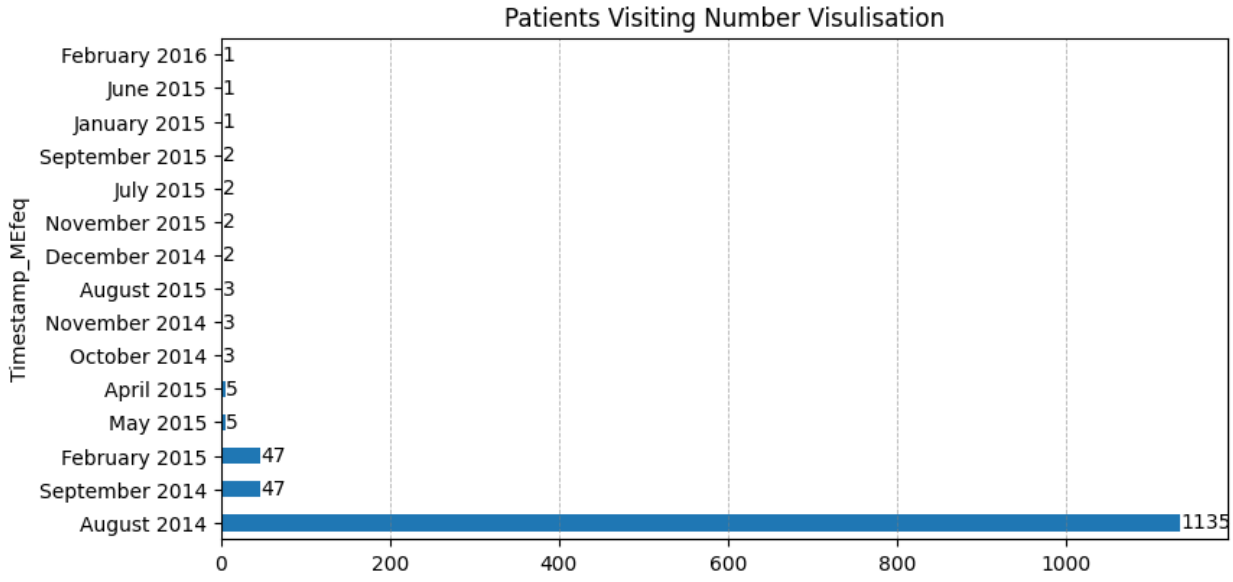


Working on Timestamp Column

```
df_WS['Timestamp']=df_WS['Timestamp'].astype('datetime64[ns]')

df_WS['Timestamp'].max()

Timestamp('2016-02-01 23:04:31')

df_WS['Timestamp'].min()

Timestamp('2014-08-27 11:29:31')

#This code will categorise based on the date range
# df_WS['Timestamp_MEfeq'] =
pd.date_range(start=df_WS['Timestamp'].min(),
#     end=df_WS['Timestamp'].max(),
#     freq='ME')
# df_WS['Timestamp_MEfeq'] = pd.to_datetime(df_WS['Timestamp'],"%Y-%m-
%d %H:%M:%S").dt.strftime('%Y-%m') # Month-Year with numbers - '2014-
08'

df_WS['Timestamp_MEfeq'] = pd.to_datetime(df_WS['Timestamp'],"%Y-%m-%d
%H:%M:%S").dt.strftime('%B %Y') # %B Month-Year with Name -'August
2014'
df_WS['Timestamp_MEfeq'].value_counts()
# df_WS['Timestamp']

Timestamp_MEfeq
August 2014       1135
September 2014      47
February 2015       47
```

```
May 2015              5
April 2015            5
October 2014          3
November 2014         3
August 2015           3
December 2014         2
November 2015         2
July 2015             2
September 2015        2
January 2015          1
June 2015             1
February 2016         1
Name: count, dtype: int64
```

```
df_MEfeq= df_WS['Timestamp_MEfeq'].value_counts().plot.barh(figsize
=(9,4.5))
plt.title('Patients Visiting Number Visulisation')
plt.grid(visible =True, axis ='x', color ='gray', linestyle ='--',
linewidth =0.6, alpha = 0.6)

for cols in df_MEfeq.containers:
    df_MEfeq.bar_label(cols, label_type ='edge')
```



Working on Country & States Column

```
df_WS['Country'].value_counts()
```

```
Country
United States             751
United Kingdom            185
Canada                     72
Germany                    45
```

```
Netherlands                27
Ireland                    27
Australia                  21
France                     13
India                      10
New Zealand                 8
Poland                      7
Italy                       7
Sweden                      7
Switzerland                 7
South Africa                6
Brazil                      6
Belgium                     6
Israel                      5
Singapore                   4
Bulgaria                    4
Russia                      3
Austria                     3
Finland                     3
Mexico                      3
Denmark                     2
Greece                      2
Portugal                    2
Colombia                    2
Croatia                     2
Slovenia                    1
Costa Rica                  1
Latvia                      1
Uruguay                     1
Spain                       1
Romania                     1
Zimbabwe                    1
Japan                       1
Nigeria                     1
Hungary                     1
Bosnia and Herzegovina      1
Thailand                    1
Norway                      1
Bahamas, The                1
Moldova                     1
Georgia                     1
China                       1
Czech Republic              1
Philippines                 1
Name: count, dtype: int64
```

```
df_WS.loc[(df_WS['Country']=='United States') &
(df_WS['state'].isna())] #Very old data set from the 1 month survey
list -August 2014
```

```
                Timestamp  Age  Gender        Country state
self_employed  \
52   2014-08-27 11:45:33   31    Male  United States   NaN
No
294  2014-08-27 14:15:57   56    Male  United States   NaN
No
367  2014-08-27 15:13:33   36    Male  United States   NaN
No
525  2014-08-27 17:32:04   41  Female  United States   NaN
No
574  2014-08-27 20:52:20   50    Male  United States   NaN
No
596  2014-08-27 22:14:23   24  Female  United States   NaN
No
638  2014-08-28 03:13:10   35    Male  United States   NaN
Yes
817  2014-08-28 14:41:47   44    Male  United States   NaN
Yes
854  2014-08-28 17:01:06   31    Male  United States   NaN
No
926  2014-08-28 21:27:19   43    Male  United States   NaN
No
1019 2014-08-29 10:03:24   25    Male  United States   NaN
No

     family_history treatment work_interfere    no_employees  ...  \
52               No        No            NaN         100-500  ...
294              No       Yes          Never  More than 1000  ...
367             Yes       Yes          Often         100-500  ...
525             Yes       Yes         Rarely        500-1000  ...
574              No        No          Never          26-100  ...
596             Yes       Yes      Sometimes         100-500  ...
638              No        No            NaN             1-5  ...
817             Yes       Yes      Sometimes             1-5  ...
854             Yes        No            NaN           6-25  ...
926             Yes        No      Sometimes        500-1000  ...
1019             No        No         Rarely          26-100  ...

     mental_health_consequence phys_health_consequence
coworkers  \
52                       Maybe                   Maybe  Some of them

294                        No                   Maybe           Yes

367                        No                      No  Some of them

525                      Maybe                   Maybe  Some of them

574                        No                      No            No
```

```
596                          Yes                    Maybe            No

638                           No                       No  Some of them

817                          Yes                      Yes  Some of them

854                        Maybe                       No  Some of them

926                        Maybe                       No            No

1019                         Yes                       No  Some of them


        supervisor mental_health_interview phys_health_interview  \
52    Some of them                    Maybe                  Maybe
294   Some of them                       No                  Maybe
367   Some of them                       No                     No
525   Some of them                       No                     No
574             No                       No                  Maybe
596             No                       No                     No
638            Yes                       No                     No
817             No                       No                     No
854   Some of them                       No                     No
926   Some of them                       No                  Maybe
1019  Some of them                       No                    Yes

     mental_vs_physical obs_consequence  \
52           Don't know              No
294          Don't know              No
367          Don't know              No
525                 Yes              No
574                  No              No
596                  No             Yes
638                 Yes              No
817                 Yes              No
854          Don't know              No
926                  No              No
1019         Don't know             Yes

                                                  comments
Timestamp_MEfeq
52                                                     NaN     August
2014
294                                                    NaN     August
2014
367                                                    NaN     August
2014
525                                                    NaN     August
2014
574                                                    NaN     August
```

```
2014
596                                                    NaN     August
2014
638                                                    NaN     August
2014
817                                                    NaN     August
2014
854                                                    NaN     August
2014
926   My employer gives access to basic counseling a...     August
2014
1019  My work is using my brain. I do it incredibly ...     August
2014

[11 rows x 28 columns]
```

```
df_WS.loc[df_WS['Country'] == 'United States', 'state']=
df_WS.loc[df_WS['Country'] == 'United States', 'state'].fillna(value =
'StatesNotFound')
```

```
df_WS.loc[df_WS['Country'] == 'United States', 'state'].value_counts()
```

```
state
CA                138
WA                 70
NY                 56
TN                 45
TX                 44
OH                 30
OR                 29
PA                 29
IL                 28
IN                 27
MI                 22
MN                 21
MA                 20
FL                 15
VA                 14
NC                 14
GA                 12
WI                 12
MO                 12
StatesNotFound     11
UT                 10
CO                  9
AL                  8
AZ                  7
MD                  7
NJ                  6
OK                  6
```

```
KY                    5
SC                    5
CT                    4
IA                    4
DC                    4
NH                    3
SD                    3
VT                    3
KS                    3
NV                    3
NM                    2
WY                    2
NE                    2
WV                    1
ID                    1
MS                    1
RI                    1
LA                    1
ME                    1
Name: count, dtype: int64
```

Problem Question 1:

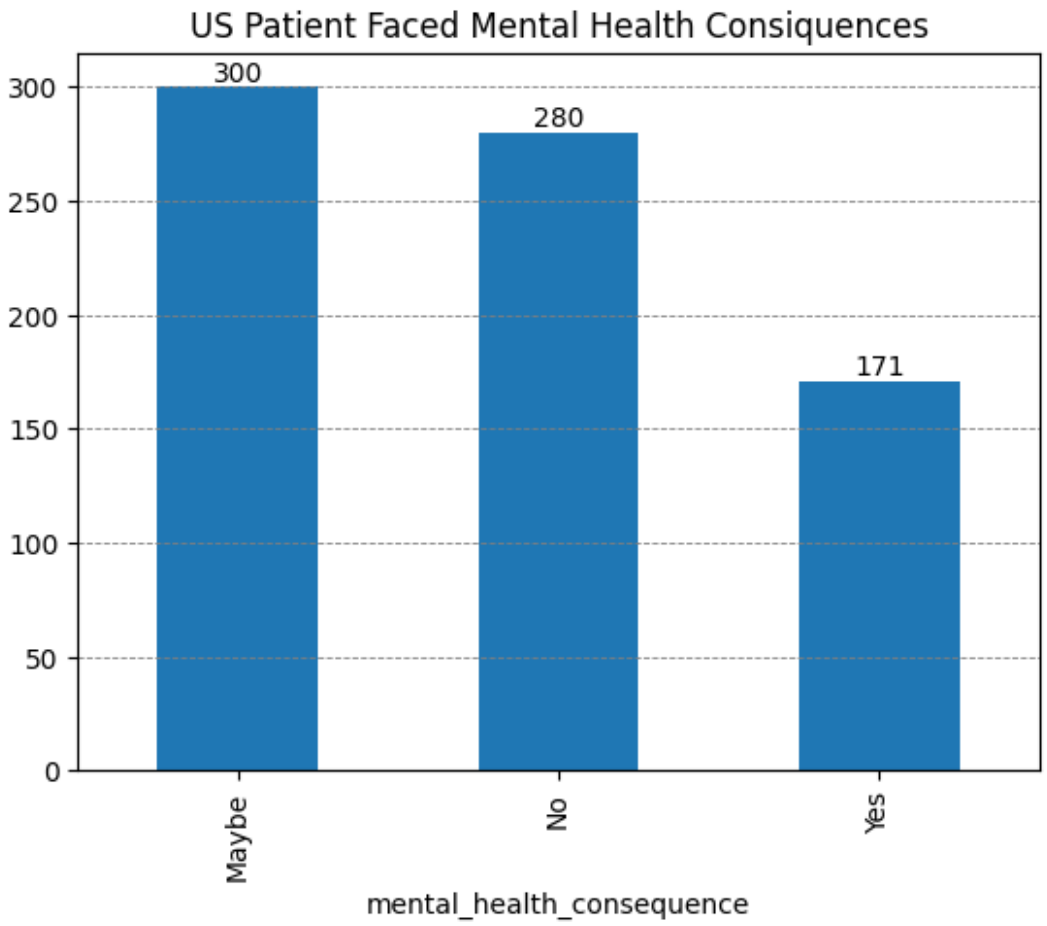
How does the frequency of mental health illness and attitudes towards mental health vary by geographic location?

Within US how statewise mental health illness increases or decreases based on the 'mental_health_consequence' for attitudes towards mental.

based on the family history, patient_count, work interfere, and other factors

Question 1.1 for US only

```
## US People Mental Health Survey Visualisation
df_MHstat = df_WS.loc[df_WS['Country']== 'United States',
'mental_health_consequence'].value_counts().plot.bar()
plt.title('US Patient Faced Mental Health Consiquences')
plt.grid(visible =True, axis = 'y', color ='gray', linestyle ='--',
linewidth=0.6)

for cols in df_MHstat.containers:
    df_MHstat.bar_label(cols, label_type= 'edge')
# df_WS[df_WS['Country']== 'United States']
[['mental_health_consequence','Timestamp_MEfeq']].plot.bar(x='','')
```



###Conclusion:

171 people faced Mental Health Consiquences after discussing with their employer

300 people may faced Mental Health Consiquences after discussing with their employer

280 people doesn't faced Mental Health Consiquences after discussing with their employer

How people think about, feel about, and respond to mental health issues?

Measured by **mental_health_consequence:**

Do people fear negative consequences if they speak up?

Based on the above graph my answer is "may be". Only ~30% people think or they may faced consiquences previously

```
df_MH_sw = df_WS[df_WS['Country']== 'United States']
[['state','mental_health_consequence']].copy()

df_MH_sw= df_MH_sw.groupby(['state',
                            'mental_health_consequence'],
observed=False)
```

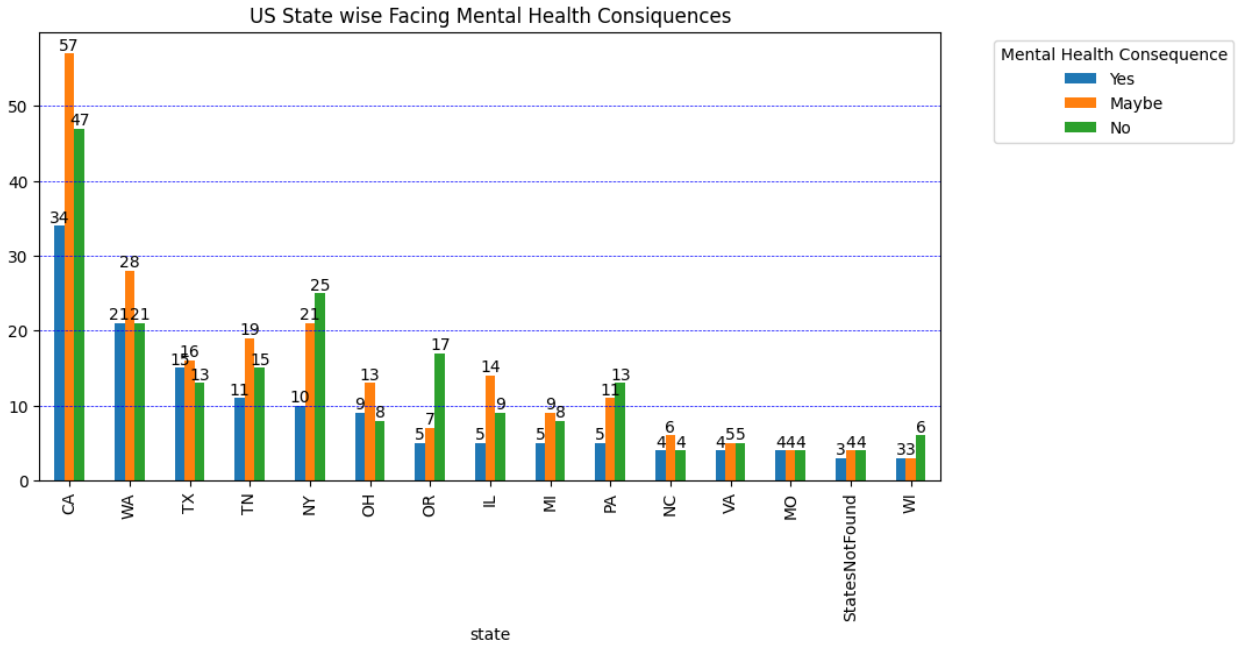
```
['mental_health_consequence'].count().reset_index(name='Patient_count'
)

df_MH_sw.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 114 entries, 0 to 113
Data columns (total 3 columns):
 #   Column                      Non-Null Count  Dtype 
---  ------                      --------------  ----- 
 0   state                       114 non-null    object
 1   mental_health_consequence  114 non-null    object
 2   Patient_count               114 non-null    int64 
dtypes: int64(1), object(2)
memory usage: 2.8+ KB
```

```
df_MH_sw=df_MH_sw.pivot(columns='mental_health_consequence',
index='state',
values='Patient_count').reset_index(drop =False)

df_MH_sw.head()
```

mental_health_consequence	state	Maybe	No	Yes
0	AL	4.0	3.0	1.0
1	AZ	5.0	1.0	1.0
2	CA	57.0	47.0	34.0
3	CO	5.0	4.0	NaN
4	CT	NaN	2.0	2.0

```
## US People Mental Health Survey Visualisation based on the states
df_MH_sw_V= df_MH_sw.sort_values(by ='Yes',ascending =
False).head(15).plot.bar(x= 'state',y=['Yes','Maybe','No'], figsize
=(10,5))
plt.title('US State wise Facing Mental Health Consiquences')
plt.legend(title='Mental Health Consequence', bbox_to_anchor=
(1.05,1), loc='upper left')
plt.grid(visible =True, axis ='y', color ='blue', linestyle ='--',
linewidth = 0.5)

for cols in df_MH_sw_V.containers:
    df_MH_sw_V.bar_label(cols, label_type ='edge')
```



Conclusion:

In "CA" :

34(state wise highest) people think/faced face mental health consiquences after discussing with their employer

57(state wise highest) people may faced or think they will face mental health consiquences after discussing with their employer

Frequency of Mental Health Illness:

How common is mental health illness in different locations?

Measured by:

treatment: Whether people sought treatment

family_history: Whether there's a family history of mental illness

work_interfere: Whether mental illness interferes with work

```
dfMH_common = df_WS[df_WS['Country']== 'United States']
[['state','family_history','work_interfere','treatment']].copy()

dfMH_common.head()
```

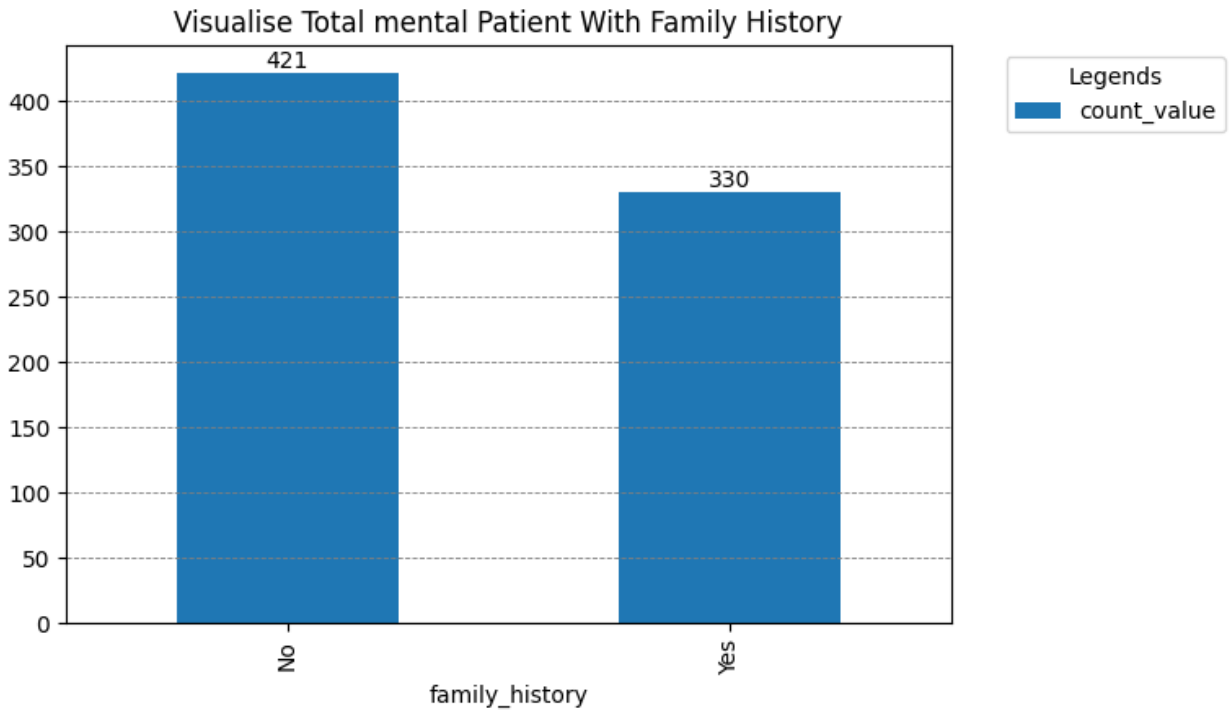
	state	family_history	work_interfere	treatment
0	IL	No	Often	Yes
1	IN	No	Rarely	No
4	TX	No	Never	No

```
5    TN            Yes      Sometimes        No
6    MI            Yes      Sometimes       Yes
```

```
## Visualise mental health patient with family_history background

dfMhY_fh= dfMH_common.groupby(['state',
                               'family_history'])
['family_history'].count().reset_index(name = 'patient_count')
dfMhY_fh.head()
```

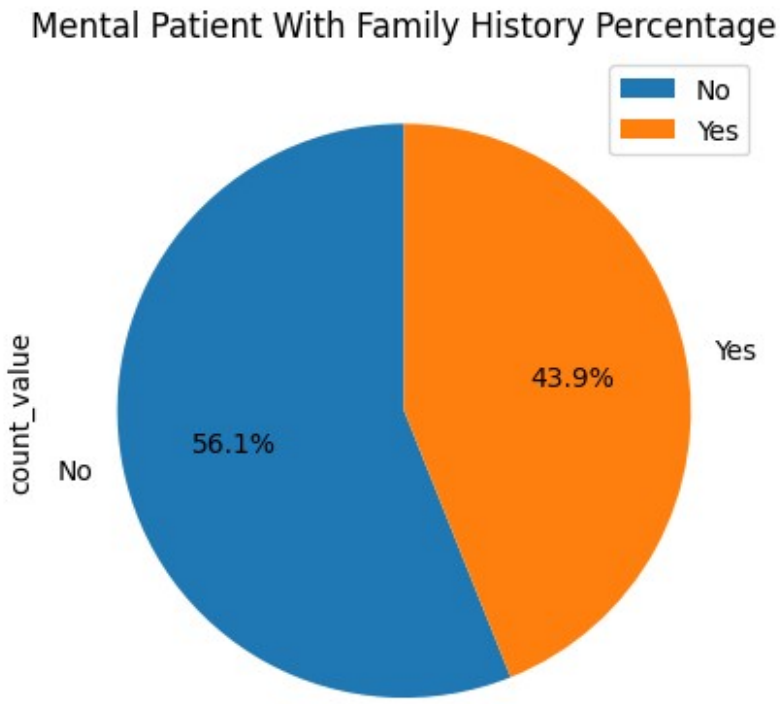
```
  state family_history  patient_count
0    AL             No              3
1    AL            Yes              5
2    AZ             No              4
3    AZ            Yes              3
4    CA             No             70
```

```
# dfMhY_Tfh: Hold the total number of mental Patient values with
Family background
dfMhY_Tfh = (dfMhY_fh[['family_history',
                       'patient_count']].groupby('family_history')
['patient_count'].sum().reset_index(name =
'count_value')).plot.bar(x='family_history'

,y='count_value',

figsize=(7,4.6))
plt.title('Visualise Total mental Patient With Family History')
plt.legend(title = 'Legends',
           bbox_to_anchor =(1.05,1),
           loc = 'upper left')
plt.grid(visible = True,
         axis = 'y',
         color= 'gray',
         linestyle= '--',
         linewidth= 0.6 )

for cols in dfMhY_Tfh.containers:
    dfMhY_Tfh.bar_label(cols, label_type ='edge')
```



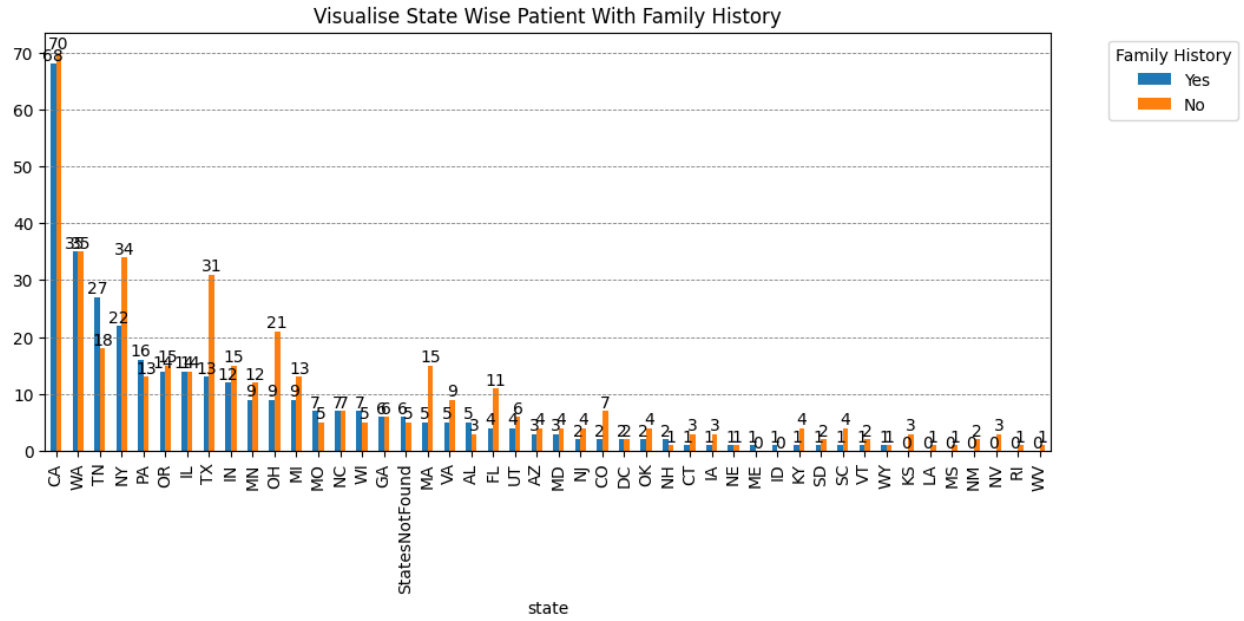

```
(dfMhY_fh[['family_history',
                        'patient_count']].groupby('family_history')
['patient_count'].sum().reset_index(name =
'count_value')).plot.pie(y='count_value',

labels= dfMhY_fh['family_history'],

autopct='%1.1f%%',

startangle = 90,

title= 'Mental Patient With Family History Percentage')
```

```
<Axes: title={'center': 'Mental Patient With Family History
Percentage'}, ylabel='count_value'>
```



In US 56.1% people doesn't have any mental patient from their family history.

```
dfMhY_fh =
MH_pivote(dfMhY_fh,'family_history','state','patient_count')

dfMhY_fhV= dfMhY_fh.sort_values(by='Yes', ascending
=False).plot.bar(x='state',y=['Yes','No'], figsize=(11,4.6))
plt.title('Visualise State Wise Patient With Family History')
plt.legend(title = 'Family History', bbox_to_anchor =(1.05,1), loc =
'upper left')
plt.grid(visible = True,
         axis = 'y',
         color= 'gray',
         linestyle= '--',
         linewidth= 0.6 )

for cols in dfMhY_fhV.containers:
    dfMhY_fhV.bar_label(cols, label_type ='edge')
```



In CA 49.27% people have mental patient from their family members

```
## Whether mental illness interferes with work

dfMH_WorkStat=
dfMH_common[['state','work_interfere']].groupby(['state','work_interfe
re'],
                                                              observed=
False)['work_interfere'].count().reset_index(name ='Value_count')
dfMH_WorkStat.head()
```

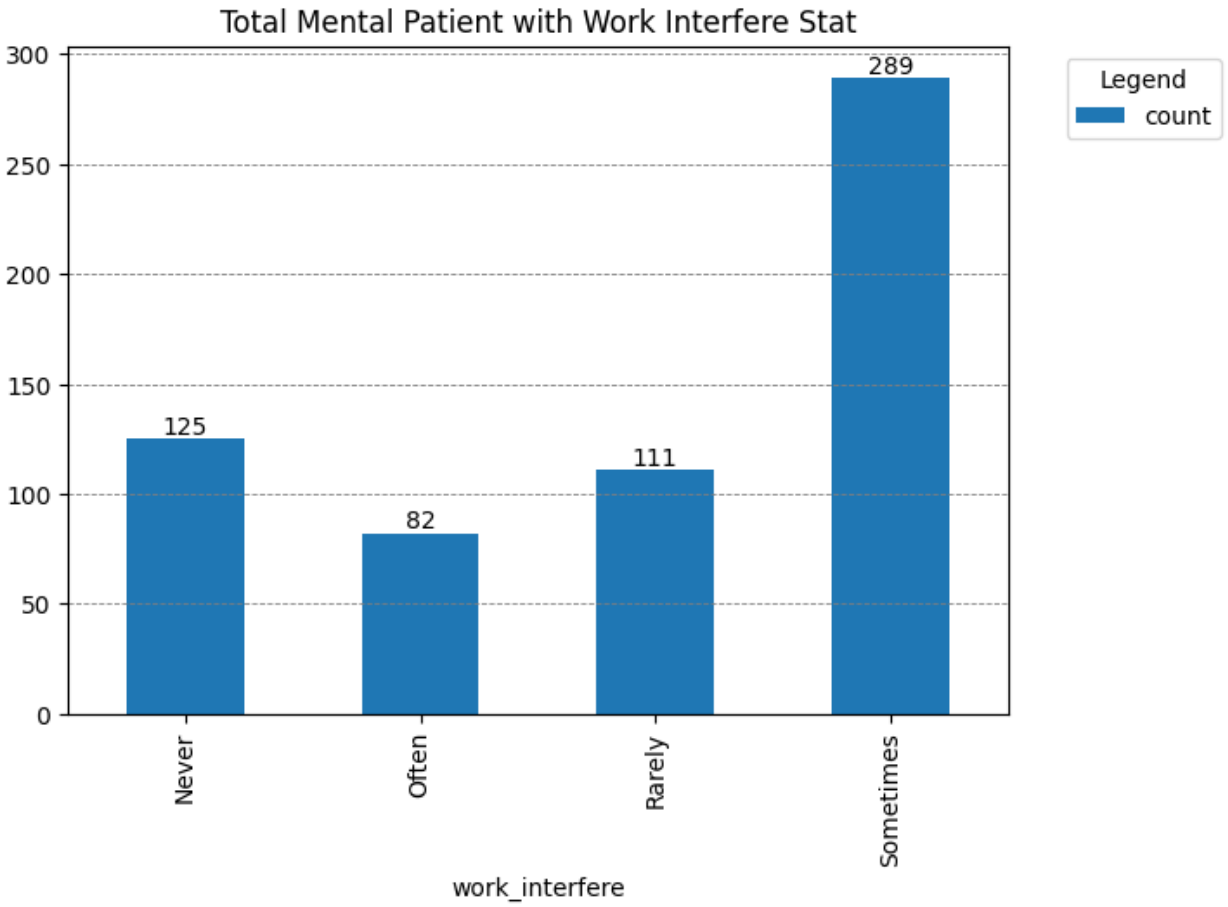
	state	work_interfere	Value_count
0	AL	Never	1
1	AL	Often	1
2	AL	Sometimes	6
3	AZ	Often	1
4	AZ	Rarely	1

```
#"dfMH_WT_Stat" Total Mental Patient with Work Interfere Stat
dfMH_WT_Stat=
dfMH_WorkStat[['work_interfere','Value_count']].groupby('work_interfer
e')['Value_count'].sum().reset_index(name = 'count')
dfMH_WT_Stat.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4 entries, 0 to 3
Data columns (total 2 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   work_interfere  4 non-null      object
 1   count           4 non-null      int64
```

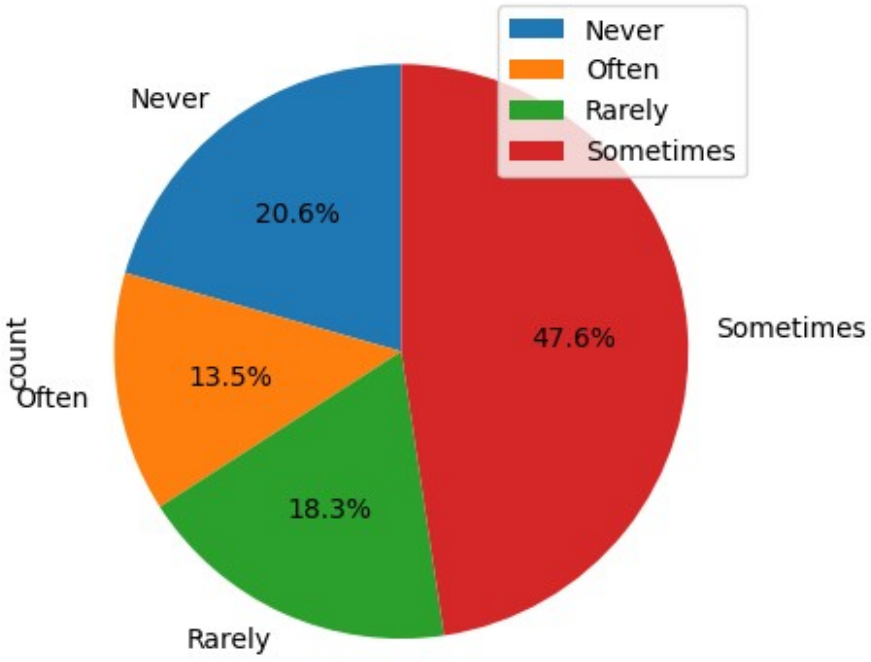
```
dtypes: int64(1), object(1)
memory usage: 196.0+ bytes
```

```
# Visuisation Total Mental Patient with Work Interfere Stat
dfMH_WT_StatV= dfMH_WT_Stat.copy()

dfMH_WT_StatV = bar_function(
    dfB= dfMH_WT_StatV ,
    colsName1= 'work_interfere' ,
    colsName2 = 'count',
    graphKind = 'bar',
    wt = 7,
    ht= 5,
    grphTitle = 'Total Mental Patient with Work Interfere Stat',
    legnTitle = 'Legend',
    grdAxix = 'y')

# dfMH_WT_Stat.plot.bar(x='work_interfere', y='count', figsize =
(7 ,4.5))
# plt.title('Total Mental Patient with Work Interfere Stat')
# plt.legend(title='Legend', bbox_to_anchor = (1.05,1), loc= 'upper
left')
# plt.grid(visible =True, axis= 'y', linestyle='--', linewidth ='0.6')

# for cols in dfMH_WT_StatV.containers:
#     dfMH_WT_StatV.bar_label(cols, label_type = 'edge')
```




```
dfMH_WT_Stat.plot.pie(y='count',
                      labels = dfMH_WT_Stat['work_interfere'],
                      autopct= '%1.1f%%',
                      startangle=90)
```

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

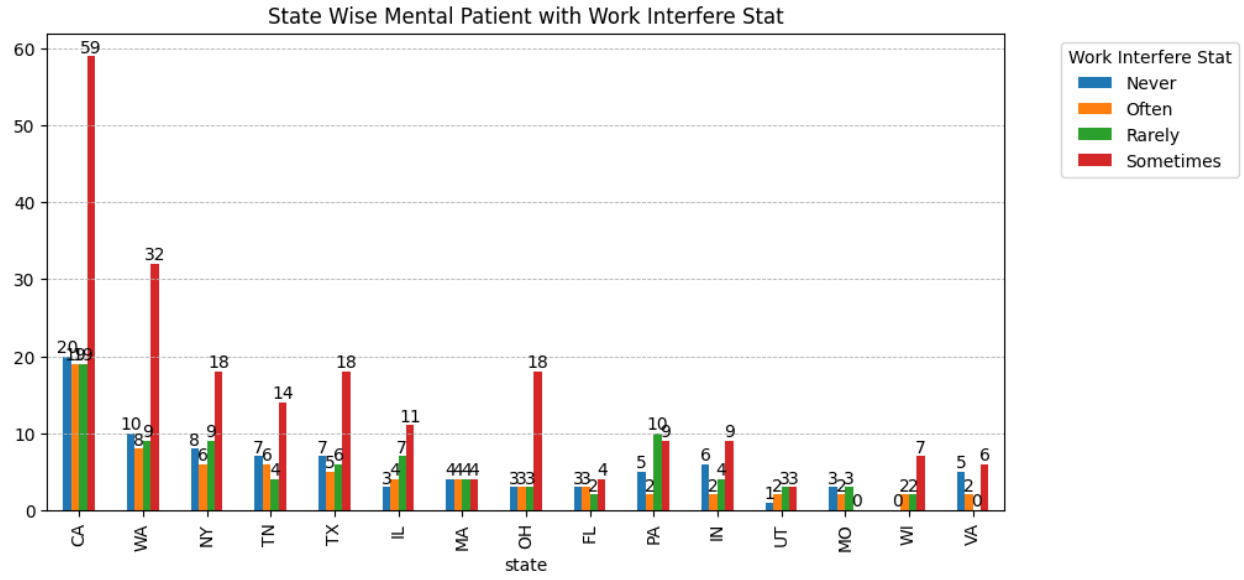


Conclusion:

47.6% people work interfere sometime happened or they think will sometime happen if they speak it to their employer

```
#Pivoting "dfMH_WorkStat" based on "work_interfere" data
dfMH_WorkStat=
MH_pivote(dfMH_WorkStat,'work_interfere','state','Value_count')

dfMH_WorkStat.info() #No null value present here

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 44 entries, 0 to 43
Data columns (total 5 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   state      44 non-null     object
 1   Never      35 non-null     float64
 2   Often      27 non-null     float64
 3   Rarely     27 non-null     float64
 4   Sometimes  38 non-null     float64
dtypes: float64(4), object(1)
memory usage: 1.8+ KB

dfMH_WorkStat[['Never','Often','Rarely','Sometimes']] =
dfMH_WorkStat[['Never','Often','Rarely','Sometimes']].fillna(value =
0)
#Replacing "'Never','Often','Rarely','Sometimes'" columns null values
```

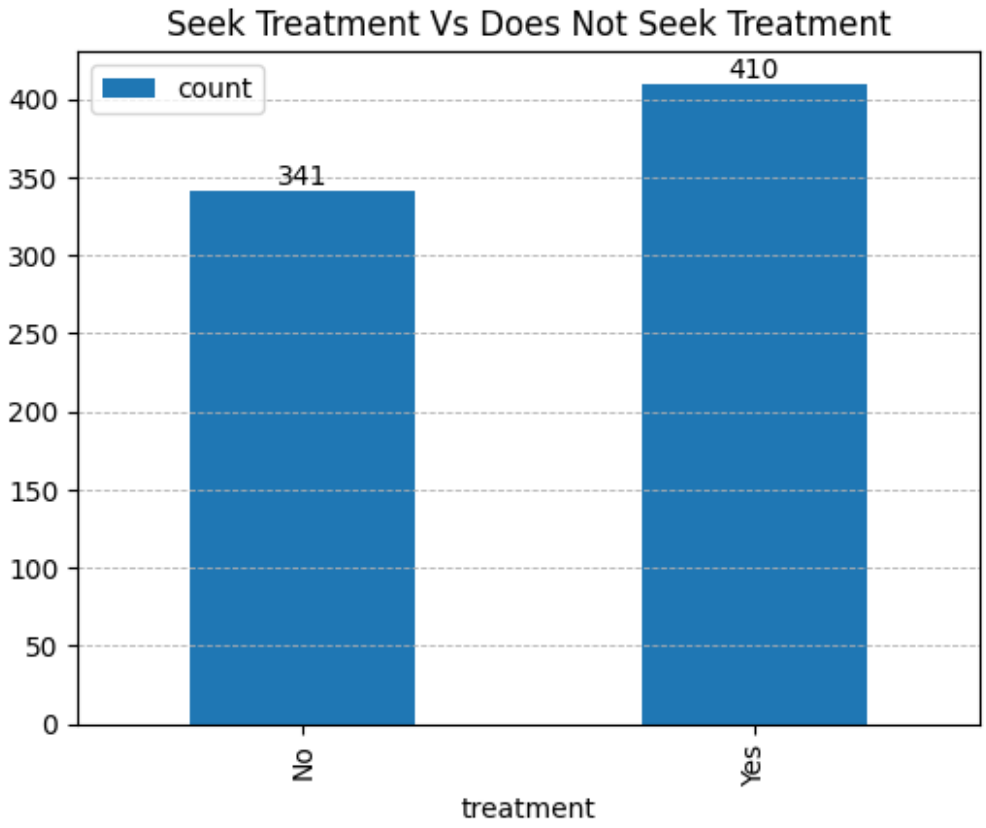
```
with 0

dfMH_WorkStat[['Never','Often','Rarely','Sometimes']] =
dfMH_WorkStat[['Never','Often','Rarely','Sometimes']].astype('int64')
#Changing "'Never','Often','Rarely','Sometimes'" columns data types to
int64

# plt.figure(figsize=(8, 7))
# sns.heatmap(data=dfMH_WorkStat[['Often','Rarely','Sometimes']],
cmap='',fmt='.2g',yticklabels='auto')

# dfMH_WorkStat.plot(kind ='')

## State Wise Top 15 Mental Patient with Work Interfere Stat

dfMH_WorkStatV= dfMH_WorkStat.sort_values(by= ['Often',
'Rarely','Sometimes'],
                              ascending =
False).head(15).plot.bar(x='state',
y=['Never','Often','Rarely','Sometimes'], figsize =(10,5))
plt.title('State Wise Mental Patient with Work Interfere Stat')
plt.legend(title='Work Interfere Stat', bbox_to_anchor = (1.05,1),
loc= 'upper left')
plt.grid(visible =True, axis= 'y', linestyle='--', linewidth ='0.6')

for cols in dfMH_WorkStatV.containers:
    dfMH_WorkStatV.bar_label(cols, label_type='edge')
```



Conclusion:

In CA high percentage of people suffer from work interfere or they think will suffer

```
## How Many mental patient seek medical treatment

#Visualise total mental patient those who seek medical treatment and
who doesn't seek medical treatment

dfMH_treatTol= dfMH_common.groupby('treatment', observed= False)
['treatment'].size().reset_index(name = 'count').plot.bar(x=
'treatment',

y= 'count', figsize =(6, 4.5))
plt.title('Seek Treatment Vs Does Not Seek Treatment')
plt.grid(visible =True, axis= 'y', linestyle='--', linewidth ='0.6')

for cols in dfMH_treatTol.containers:
    dfMH_treatTol.bar_label(cols, label_type='edge')

# dfMH_treatTol
```



Conclusion:

410 people seek medical treatment

faced mental consiquences = Yes Stat finding

```
## "dfMH_TreatSW": Variable holding data for those who have maced
mental consiquences how many of them seek for/ doesn't treatment state
wise

dfMH_TreatSW= df_WS[(df_WS['mental_health_consequence']== 'Yes') &
(df_WS['Country']== 'United States')][['state',

'family_history',

'work_interfere',

'treatment']]

dfMH_TreatSW2 =dfMH_TreatSW[['state',
                             'treatment']].groupby(['state',
                                                    'treatment'],
observed= False)['treatment'].size().reset_index(name = 'count')
dfMH_TreatSW2.head()

# dfMH_TreatSW= dfMH_common[dfMH_common['mental_health_consequence']==
'Yes'][['state',
#
'treatment']].groupby(['state',
#
'treatment'], observed= False)['treatment'].size().reset_index(name =
'count')
# dfMH_TreatSW.head()
```

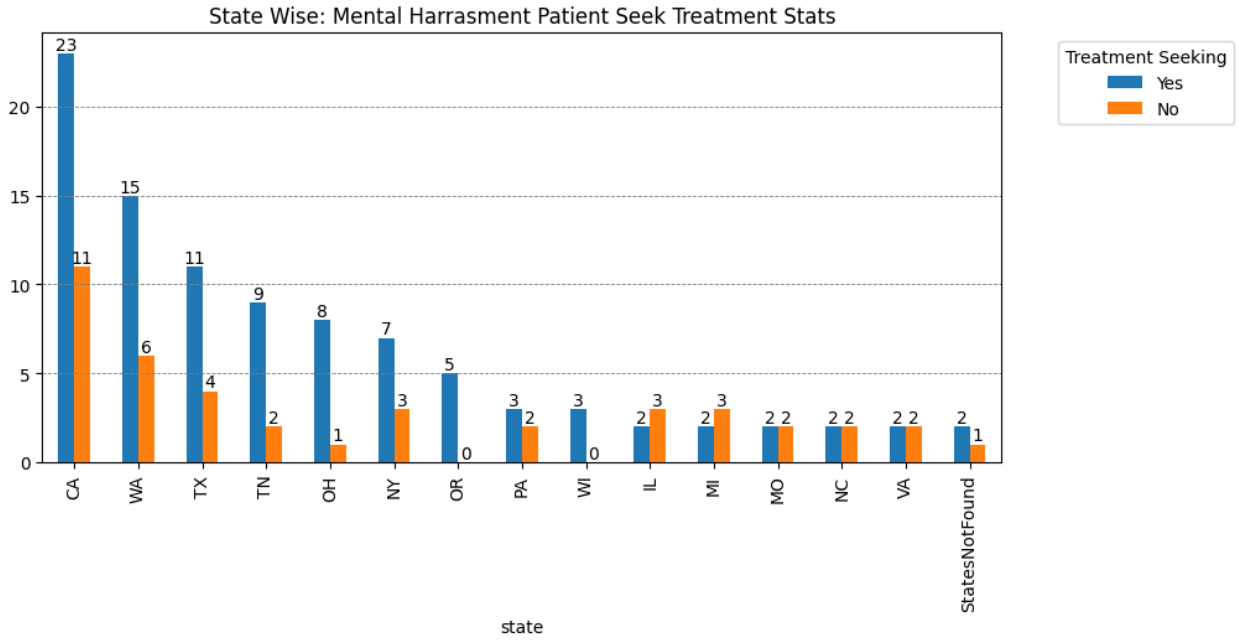
	state	treatment	count
0	AL	Yes	1
1	AZ	Yes	1
2	CA	No	11
3	CA	Yes	23
4	CT	No	1

```
dfMH_TreatSW2 =MH_pivote(dfMH_TreatSW2,'treatment','state','count')
dfMH_TreatSW2.head()
```

treatment	state	No	Yes
0	AL	NaN	1.0
1	AZ	NaN	1.0
2	CA	11.0	23.0
3	CT	1.0	1.0
4	FL	NaN	1.0

```
#Replacing "['No','Yes']" columns null values with 0
dfMH_TreatSW2[['No','Yes']] = dfMH_TreatSW2[['No','Yes']].fillna(value
=0)

#Changing " ['No','Yes'] " columns data types to int64
```

```
dfMH_TreatSW2[['No','Yes']] =
dfMH_TreatSW2[['No','Yes']].astype('int64')

## Visualise State Wise: Mental Patient Treatment Seeking Stats

dfMH_TreatSW2V= dfMH_TreatSW2.sort_values(by= ['Yes','No'], ascending
= False).head(15).plot.bar(x='state', y =['Yes','No'], figsize
=(10,4.5))
plt.title('State Wise: Mental Harrasment Patient Seek Treatment
Stats')
plt.legend(title = 'Treatment Seeking', bbox_to_anchor =(1.05,1), loc
= 'upper left')
plt.grid(visible= True, axis = 'y', color = 'gray', linestyle = "--",
linewidth= 0.6)

for cols in dfMH_TreatSW2V.containers:
    dfMH_TreatSW2V.bar_label(cols, label_type = 'edge')
```



Conclusion:

IN CA 23 people seek medical treatment (Highest among states)

IN WA 15 (71%) people doesn't seek medical treatment

```
## Finding Stats those are mental patient family background "yes",
seeking treatment

## "dfMH_TreatSW": Variable holding data for those who have faced
mental consiquences how many of them seek for/ doesn't treatment state
```

```
wise
dfMH_TreatSW.head()
```

	state	family_history	work_interfere	treatment
12	CA	Yes	Sometimes	Yes
25	TN	Yes	Sometimes	Yes
31	PA	Yes	Rarely	No
41	MI	No	NaN	No
60	IA	Yes	Sometimes	Yes

```
## "dfMH_Ot": Variable holding data for those who have past mental
patient from their family and faced work_interfere type with treatment
stat.

dfMH_Ot = dfMH_TreatSW[dfMH_TreatSW['family_history']=='Yes']
[['work_interfere','treatment']].copy()
dfMH_Ot.head(3)
# "dfMH_Ot" variable represent dataset of "work_interfere","treatment"
for ['mental_health_consequence']=='Yes' & ['family_history']=='Yes'
```

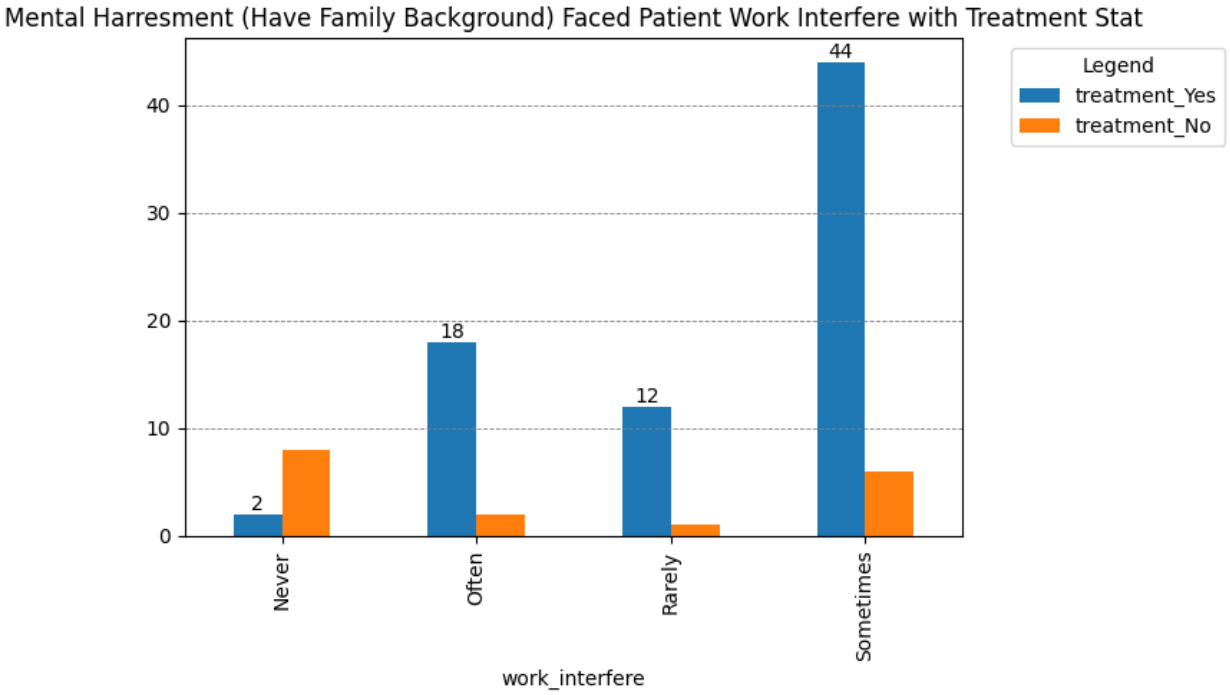
	work_interfere	treatment
12	Sometimes	Yes
25	Sometimes	Yes
31	Rarely	No

```
dfMH_OtPv= dfMH_Ot.groupby(['work_interfere','treatment'])
['treatment'].count().reset_index(name ='count')

dfMH_OtPv= MH_pivote(dfMH_OtPv,'treatment','work_interfere','count')

dfMH_OtPv= dfMH_OtPv.rename(columns=
{'No':'treatment_No','Yes':'treatment_Yes'})
dfMH_OtPv
# "dfMH_OtPv" Stats represent how "work_interfere" happens with the
mental patient those --
# -- who have previous family background "yes", seeking treatment
yes/No
```

treatment	work_interfere	treatment_No	treatment_Yes
0	Never	8	2
1	Often	2	18
2	Rarely	1	12
3	Sometimes	6	44

```
## Visualise Work Interfere Pattern

dfMH_OtPvShow = dfMH_OtPv.copy()

dfMH_OtPvShow =bar_function(
    dfB = dfMH_OtPvShow,
    colsName1 = 'work_interfere',
```

```
    colsName2 = ['treatment_Yes','treatment_No'],
    graphKind = 'bar',
    wt = 7,
    ht = 4.5,
    grphTitle = 'Mental Harresment (Have Family Background) Faced
Patient Work Interfere with Treatment Stat',
    legnTitle = 'Legend',
    grdAxix = 'y')


# dfMH_OtPvShow=  = dfMH_OtPv.plot.bar(x='work_interfere',
y=['treatment_Yes','treatment_No'], figsize = (7,5))
# plt.title('Work Interfere Pattern')
# plt.legend(title = 'Legend', bbox_to_anchor =(1.05,1), loc = 'upper
left')
# plt.grid(visible= True, axis = 'y', color = 'gray', linestyle =
"--", linewidth= 0.6)

# for cols in dfMH_OtPvShow.containers:
#     dfMH_OtPvShow.bar_label(cols, label_type = 'edge')
```



Conclusion:

mental Harresment faced patient those who have previous mental patient from their family background --

-
  - Seeking treatment people get most of the work interfere

-
  - Doesn't seeking treatment people get less work interfere

```
"""
Finding Stats those are mental harres patient with "family_history"
background "No", but "mental_health_consequence"= Yes
seeking treatment work interfere patteren

"""
```

```
'\nFinding Stats those are mental harres patient with "family_history"
background "No", but "mental_health_consequence"= Yes\nseeking
treatment work interfere patteren\n\n'
```

```
dfMH_ON= dfMH_TreatSW[ dfMH_TreatSW['family_history']=='No']
[['work_interfere','treatment']].copy()
dfMH_ON.head()
# "dfMH_ON" variable represent dataset of "work_interfere","treatment"
for ['mental_health_consequence']=='Yes' & ['family_history']=='No'
```

```
    work_interfere treatment
41             NaN        No
110      Sometimes       Yes
120          Never        No
127         Rarely       Yes
142          Never        No
```

```
dfMH_ON.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 77 entries, 41 to 1257
Data columns (total 2 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   work_interfere  61 non-null     object
 1   treatment       77 non-null     object
dtypes: object(2)
memory usage: 1.8+ KB
```

```
dfMH_ON['work_interfere'] = dfMH_ON['work_interfere'].fillna(value =
'No Data Avilable')
dfMH_ON.head()
```

```
       work_interfere treatment
41   No Data Avilable        No
110         Sometimes       Yes
120             Never        No
127            Rarely       Yes
142             Never        No
```

```
dfMH_ONpiV= dfMH_ON.groupby(['work_interfere','treatment'],observed=
False)['treatment'].count().reset_index(name = 'count')
```

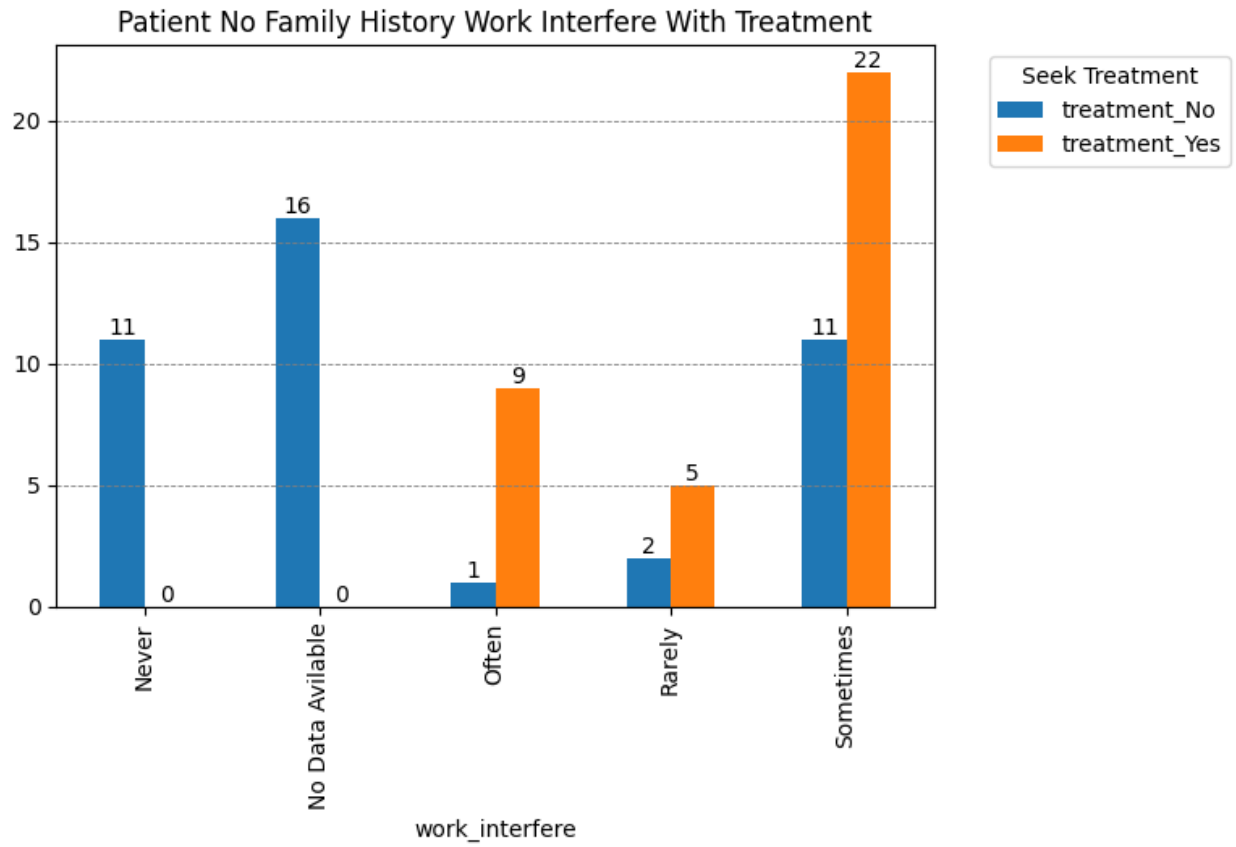
```
dfMH_ONpiV.head()
# "dfMH_ONpiV" is Pivote representation 'dfMH_ON'
```

```
     work_interfere treatment  count
0             Never        No     11
1  No Data Avilable        No     16
2             Often        No      1
3             Often       Yes      9
4            Rarely        No      2
```

```
dfMH_ONpiV =MH_pivote(dfMH_ONpiV,
'treatment','work_interfere','count')

dfMH_ONpiV = dfMH_ONpiV.rename(columns
={'No':'treatment_No','Yes':'treatment_Yes'}) #Changing the column
name
dfMH_ONpiV
```

```
treatment    work_interfere  treatment_No  treatment_Yes
0                     Never          11.0            NaN
1          No Data Avilable          16.0            NaN
2                     Often           1.0            9.0
3                    Rarely           2.0            5.0
4                 Sometimes          11.0           22.0
```

```
dfMH_ONpiV[['treatment_No','treatment_Yes']] =
dfMH_ONpiV[['treatment_No','treatment_Yes']].fillna(value = 0)
#replacing Null values with 0

dfMH_ONpiV[['treatment_No','treatment_Yes']]=
dfMH_ONpiV[['treatment_No','treatment_Yes']].astype('int64')

dfMH_onVis= dfMH_ONpiV.plot.bar(x='work_interfere', y=
['treatment_No','treatment_Yes'], figsize =(7,4.5))
plt.title('Patient No Family History Work Interfere With Treatment')
plt.legend(title = 'Seek Treatment', bbox_to_anchor =(1.05,1), loc =
'upper left')
plt.grid(visible= True, axis = 'y', color = 'gray', linestyle = "--",
linewidth= 0.6)

for cols in dfMH_onVis.containers:
    dfMH_onVis.bar_label(cols, label_type = 'edge')
```



Conclusion:

Mental Harresment faced patient those who :**does not have mental patient:** previously from their family background --

- – Seeking treatment people get most of the work interfere
- – Doesn't seeking treatment people get less work interfere
- – 16 people data is unablilable

Maybe/ Probaly Mental harrasment Faced Patient Data:

```
dfMayBe_tot = df_WS[(df_WS['mental_health_consequence']=='Maybe') &
(df_WS['Country']== 'United States')][['state',

'family_history',

'treatment', 'work_interfere']].copy()

dfMayBe_tot.head()
# variable holds data for those patient those who Maybe Probaly faced
mental harrasment:*
```

```
   state family_history treatment work_interfere
1     IN             No        No         Rarely
6     MI            Yes       Yes      Sometimes
8     IL            Yes       Yes      Sometimes
17    TN             No       Yes      Sometimes
20    NY            Yes       Yes      Sometimes
```

How many Probaly Mental harrasment faced people took Treatment based on the family history

```
dfMhMb_Tfh= dfMayBe_tot[['family_history',
                         'treatment']].groupby(['family_history',
                                                'treatment'])
['treatment'].count().reset_index(name= 'Treatment Stat')
dfMhMb_Tfh
# ['treatment'].count().reset_index(name= 'MbPatient_count')
```

```
  family_history treatment  Treatment Stat
0             No        No             101
1             No       Yes              67
2            Yes        No              27
3            Yes       Yes             105
```

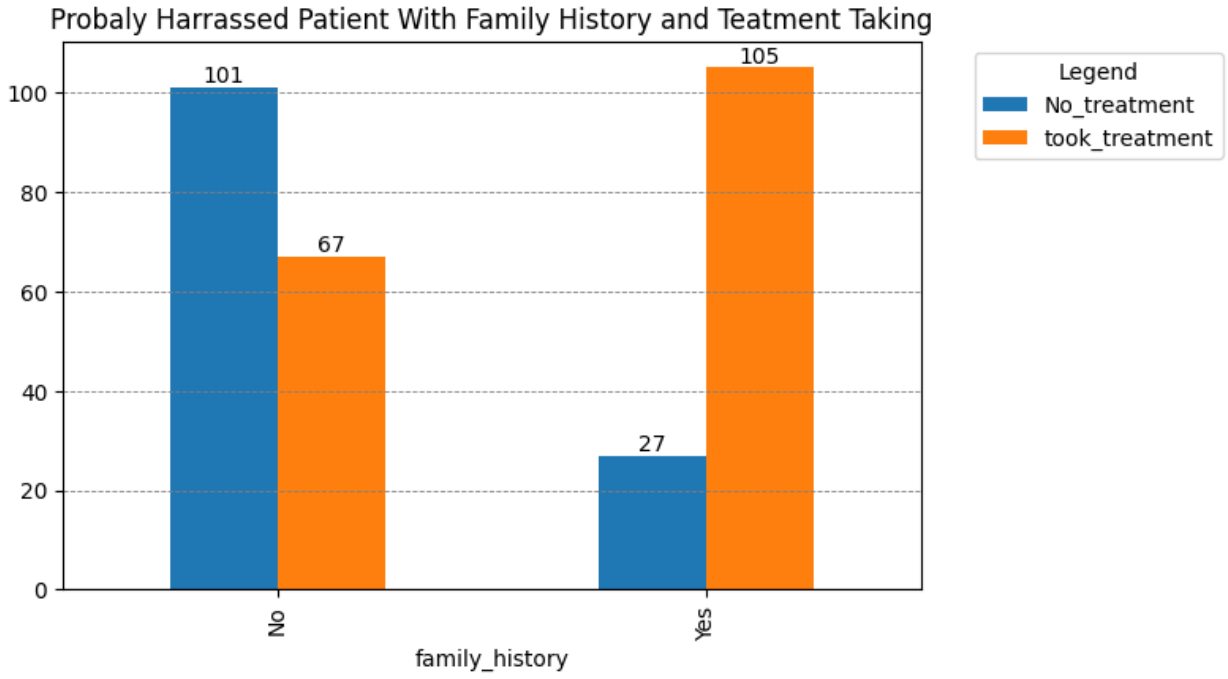
```
dfMhMb_Tfh =
MH_pivote(dfMhMb_Tfh,'treatment','family_history','Treatment Stat')
dfMhMb_Tfh
```

```
treatment family_history   No  Yes
0                    No  101   67
1                   Yes   27  105
```

```
dfMhMb_Tfh= dfMhMb_Tfh.rename(columns
={'No':'No_treatment','Yes':'took_treatment'})
dfMhMb_Tfh
```

```
treatment family_history  No_treatment  took_treatment
0                    No           101              67
1                   Yes            27             105
```

```
dfMhMb_TfhV= dfMhMb_Tfh.plot.bar(x='family_history', figsize =(7,4.5))
plt.title('Probaly Harrassed Patient With Family History and Teatment
Taking')
plt.legend(title = 'Legend', bbox_to_anchor =(1.05,1), loc = 'upper
left')
plt.grid(visible= True, axis = 'y', color = 'gray', linestyle = "--",
linewidth= 0.6)

for cols in dfMhMb_TfhV.containers:
    dfMhMb_TfhV.bar_label(cols, label_type = 'edge')
```



Conclusion:

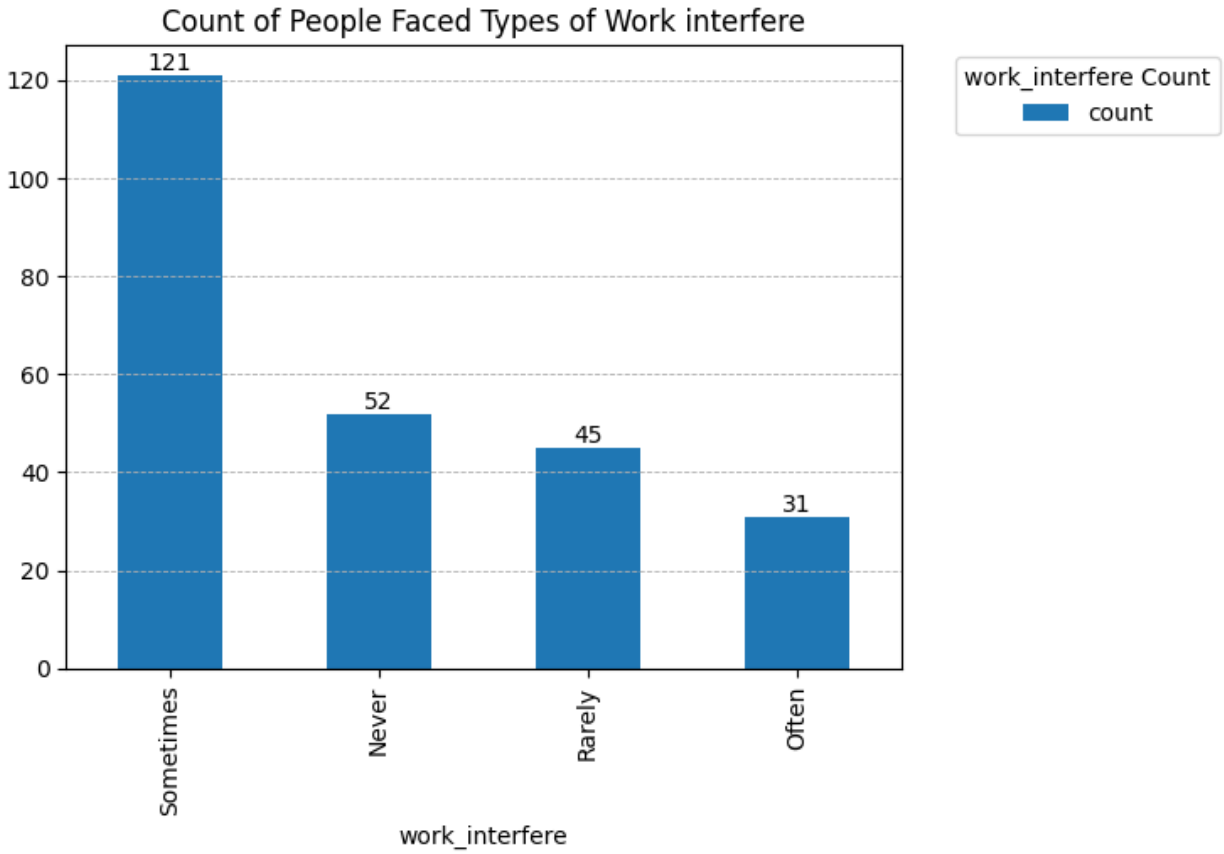
Mental Harresment faced people those who have patient from their family background they took highest number of treatment

How many Probaly Mental harrasment faced people also faced types of work interfere

```
dfMayBe_WI= dfMayBe_tot['work_interfere'].value_counts()
dfMayBe_WI
```

```
work_interfere
Sometimes    121
Never         52
Rarely        45
Often         31
Name: count, dtype: int64
```

```
dfMayBe_WIV= dfMayBe_WI.plot.bar()
plt.title('Count of People Faced Types of Work interfere')
plt.legend(title='work_interfere Count', bbox_to_anchor = (1.05,1),
loc= 'upper left')
plt.grid(visible =True, axis= 'y', linestyle='--', linewidth ='0.6')

for cols in dfMayBe_WIV.containers:
    dfMayBe_WIV.bar_label(cols, label_type= 'edge')
```



Conclusion:

Among the probaly mental harrasment faced people:

-
  - Highest number of people (121) sometime faced work interfere followed by never (52 people).

```
# "dfMayBe_StateTeatCount" variable holds the data for state wise
"work_interfere" types count --
## -- (those who may/probably faced mental health consiquences in
their life)

dfMayBe_StateTeatCount= dfMayBe_tot.groupby(['state',
                                             'work_interfere'],
observed = False)['work_interfere'].count().reset_index(name
='Interfere_TypeCount')

dfMayBe_StateTeatCount.head()
```

```
  state work_interfere  Interfere_TypeCount
0    AL          Never                    1
1    AL      Sometimes                    3
2    AZ      Sometimes                    3
```

```
3    CA          Never                    12
4    CA          Often                     9
```

```
dfMayBe_StateTeatCount =
MH_pivote(dfMayBe_StateTeatCount,'work_interfere','state','Interfere_T
ypeCount')
dfMayBe_StateTeatCount.head()
```

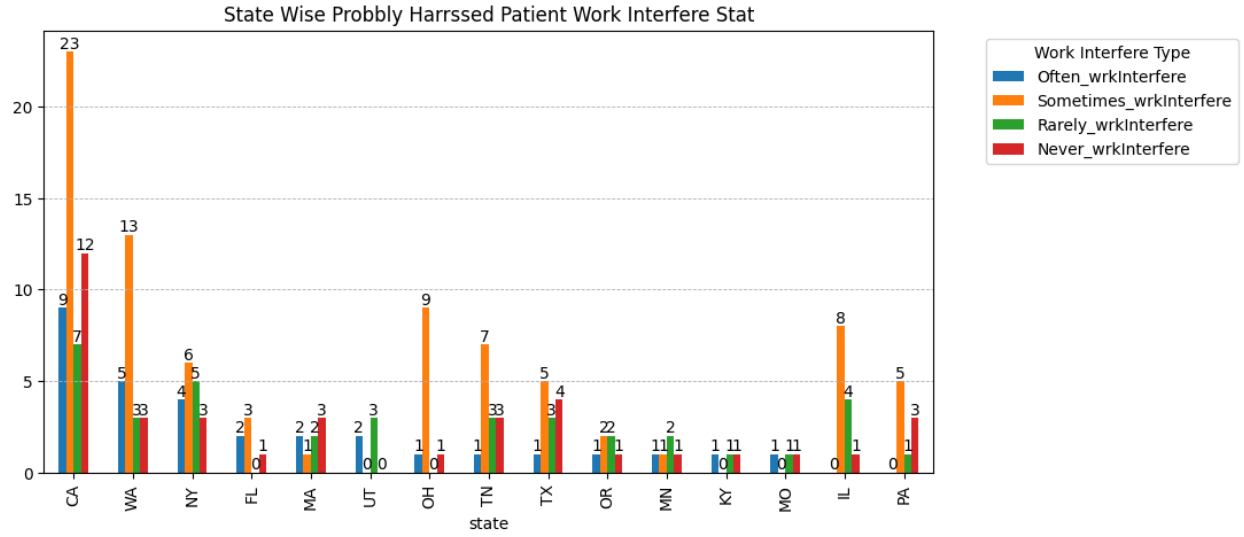
```
work_interfere state  Never  Often  Rarely  Sometimes
0                 AL    1.0    NaN     NaN        3.0
1                 AZ    NaN    NaN     NaN        3.0
2                 CA   12.0    9.0     7.0       23.0
3                 CO    3.0    NaN     1.0        1.0
4                 DC    1.0    NaN     NaN        1.0
```

```
dfMayBe_StateTeatCount= dfMayBe_StateTeatCount.fillna(value= 0)
#filling Null values with Zero

dfMayBe_StateTeatCount = dfMayBe_StateTeatCount.rename(columns=
{'Never':'Never_wrkInterfere',

'Often': 'Often_wrkInterfere',

'Rarely':'Rarely_wrkInterfere',

'Sometimes': 'Sometimes_wrkInterfere'})
dfMayBe_StateTeatCount.head(3)
```

```
work_interfere state  Never_wrkInterfere  Often_wrkInterfere  \
0                 AL                 1.0                 0.0
1                 AZ                 0.0                 0.0
2                 CA                12.0                 9.0

work_interfere  Rarely_wrkInterfere  Sometimes_wrkInterfere
0                              0.0                     3.0
1                              0.0                     3.0
2                              7.0                    23.0
```

```
dfMhMb_fhV=
dfMayBe_StateTeatCount.sort_values(by=['Often_wrkInterfere','Sometimes
_wrkInterfere','Rarely_wrkInterfere']
                                                , ascending=
False).head(15).plot.bar(x='state', y= ['Often_wrkInterfere',

'Sometimes_wrkInterfere',

'Rarely_wrkInterfere',

'Never_wrkInterfere'], figsize=(10,5))
plt.title('State Wise Probbly Harrssed Patient Work Interfere Stat')
plt.legend(title='Work Interfere Type', bbox_to_anchor = (1.05,1),
```

```
loc= 'upper left')
plt.grid(visible =True, axis= 'y', linestyle='--', linewidth ='0.6')

for cols in dfMhMb_fhV.containers:
    dfMhMb_fhV.bar_label(cols, label_type='edge')
```



Conclusion:

Among the probaly mental harrasment faced people:

In CA highest number of people often or sometime faced Work Interfere.

FINDING STAT FOR:

- How many Probaly Mental harrasment faced Patient also has "Family Background = Yes/No" -

```
dfMayBe_StatefamHis= dfMayBe_tot.groupby(['state',
                                          'family_history',],
observed = False)['family_history'].count().reset_index(name
='Interfere_TypeCount')

dfMayBe_StatefamHis.head()
```

	state	family_history	Interfere_TypeCount
0	AL	No	1
1	AL	Yes	3
2	AZ	No	3
3	AZ	Yes	2
4	CA	No	32

```
dfMayBe_StatefamHisPiv = dfMayBe_StatefamHis.copy()
dfMayBe_StatefamHisPiv =MH_pivote(dfMayBe_StatefamHisPiv,
```

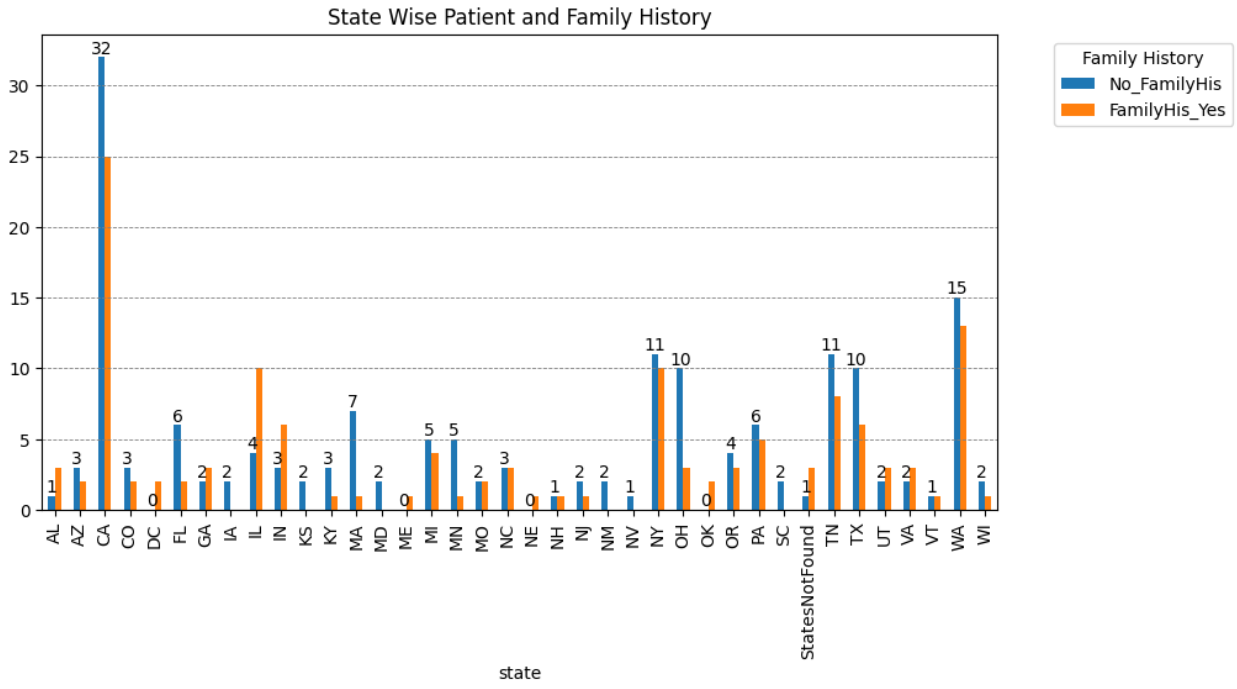
```
'family_history','state','Interfere_TypeCount')
dfMayBe_StatefamHisPiv.head()
```

family_history	state	No	Yes
0	AL	1.0	3.0
1	AZ	3.0	2.0
2	CA	32.0	25.0
3	CO	3.0	2.0
4	DC	NaN	2.0

```
dfMayBe_StatefamHisPiv= dfMayBe_StatefamHisPiv.fillna(value= 0)
#filling Null values with Zero

dfMayBe_StatefamHisPiv = dfMayBe_StatefamHisPiv.rename(columns=
{'No':'No_FamilyHis',

'Yes': 'FamilyHis_Yes'})

dfMayBe_StatefamHisPiv.head(3)
```

family_history	state	No_FamilyHis	FamilyHis_Yes
0	AL	1.0	3.0
1	AZ	3.0	2.0
2	CA	32.0	25.0

```
dfMayBe_StFmHisVis = bar_function(
    dfB = dfMayBe_StatefamHisPiv,
    colsName1 = 'state',
    colsName2 = ['No_FamilyHis','FamilyHis_Yes'],
    graphKind = 'bar',
    wt = 10,
    ht = 5,
    grphTitle = 'State Wise Patient and Family History',
    legnTitle = 'Family History',
    grdAxix = 'y',
)
```



Conclusion:

Among the probaly mental harrasment faced people:

In CA highest number of people :**doesn't have patient:** from their family background.

FINDING STAT FOR:

- How many Probaly Mental harrasment faced Patient with "Family Background = Yes" took treatment and work interfere happens-

```
dfMayBeOStat = dfMayBe_tot[(dfMayBe_tot['family_history']== 'Yes') &
(dfMayBe_tot['treatment']== 'Yes') &
(dfMayBe_tot['work_interfere'].isin(['Often',

'Sometimes','Rarely']))][['state',

'work_interfere']].copy()

dfMayBeOStatPiv= dfMayBeOStat.groupby(['state','work_interfere'],
observed = False)['work_interfere'].count().reset_index(name =
'count')

dfMayBeOStatPiv =
MH_pivote(dfMayBeOStatPiv,'work_interfere','state','count')
dfMayBeOStatPiv.head()
```

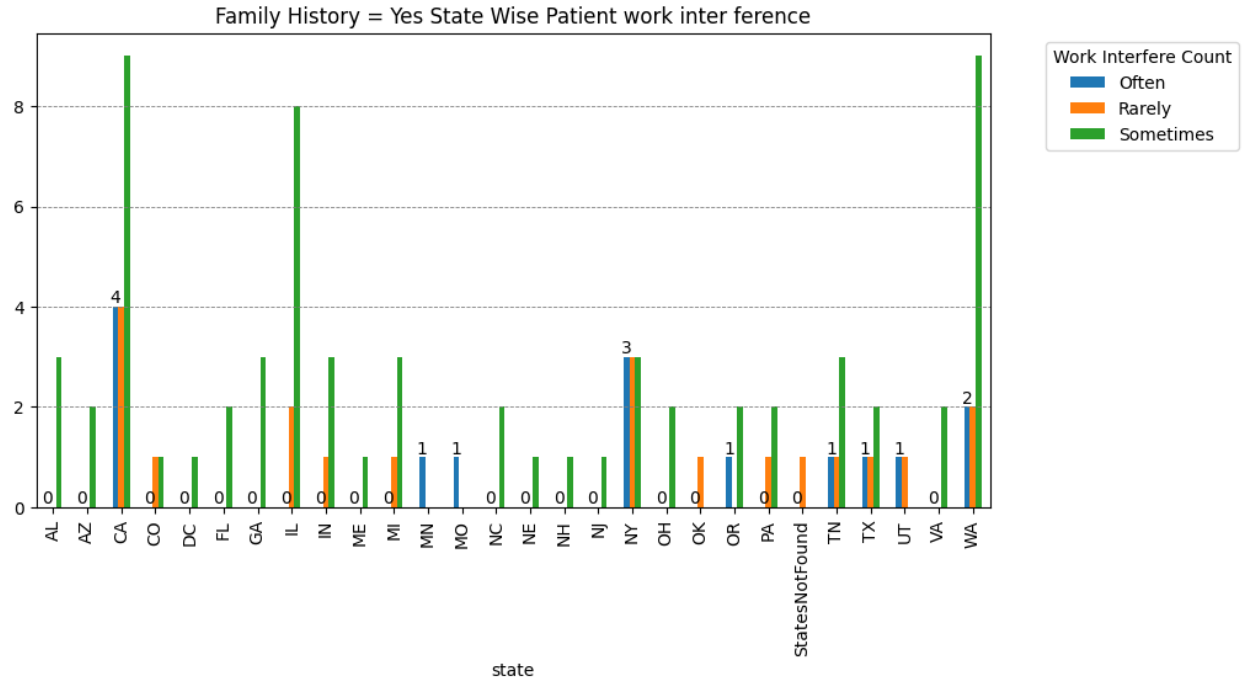
work_interfere	state	Often	Rarely	Sometimes
0	AL	NaN	NaN	3.0
1	AZ	NaN	NaN	2.0

```
2                  CA    4.0     4.0         9.0
3                  CO    NaN     1.0         1.0
4                  DC    NaN     NaN         1.0
```

```
dfMayBeOStatPiv = dfMayBeOStatPiv.fillna(value = 0)

dfMayBeOStatPiv.head()
```

```
work_interfere state  Often  Rarely  Sometimes
0                  AL    0.0     0.0         3.0
1                  AZ    0.0     0.0         2.0
2                  CA    4.0     4.0         9.0
3                  CO    0.0     1.0         1.0
4                  DC    0.0     0.0         1.0
```

```
dfMayBeOStatPivVis =bar_function(
    dfB = dfMayBeOStatPiv,
    colsName1 = 'state',
    colsName2 = ['Often','Rarely','Sometimes'],
    graphKind = 'bar',
    wt = 10,
    ht = 5,
    grphTitle = 'Family History = Yes State Wise Patient work inter
ference',
    legnTitle = 'Work Interfere Count',
    grdAxix = 'y'
)
```



Conclusion:

Among the probaly mental harrasment faced people "Family Background = Yes" took treatment and work interfere happens-**

In CA,WA,IL people **those have patient** from their family background:

-
  - took treatment and work interfere happens **"sometimes".**

In CA,NY people **those have patient** from their family background:

-
  - took treatment and work interfere happens **"Often" followed by "sometimes".**

```
print("Note: Data Showing For Family Background = YES and May faced
negetive consequences from the employer:")
print()
print(f"Number of people *Often* faced Work Interfere
{dfMayBeOStatPiv['Often'].value_counts().sum()}")
print(f"Max Number of people *Often* faced Work Interfere
{dfMayBeOStatPiv['Often'].max()} at
{dfMayBeOStatPiv[dfMayBeOStatPiv['Often']==
dfMayBeOStatPiv['Often'].max()]['state']}")
print()
print(f"Number of people *Sometimes* faced Work Interfere
{dfMayBeOStatPiv['Sometimes'].value_counts().sum()}")
print(f"Max Number of people *Sometimes* faced Work Interfere
{dfMayBeOStatPiv['Sometimes'].max()} at
{dfMayBeOStatPiv[dfMayBeOStatPiv['Sometimes']==
dfMayBeOStatPiv['Sometimes'].max()]['state']}")
print()
print(f"Number of people *Rarely* faced Work Interfere
{dfMayBeOStatPiv['Rarely'].value_counts().sum()}")
print(f"Max Number of people *Rarely* faced Work Interfere
{dfMayBeOStatPiv['Rarely'].max()} at
{dfMayBeOStatPiv[dfMayBeOStatPiv['Rarely']==
dfMayBeOStatPiv['Rarely'].max()]['state']}")
```

```
Note: Data Showing For Family Background = YES and May faced negetive
consequences from the employer:

Number of people *Often* faced Work Interfere 28
Max Number of people *Often* faced Work Interfere 4.0 at 2    CA
Name: state, dtype: object

Number of people *Sometimes* faced Work Interfere 28
Max Number of people *Sometimes* faced Work Interfere 9.0 at 2     CA
27    WA
Name: state, dtype: object

Number of people *Rarely* faced Work Interfere 28
```

```
Max Number of people *Rarely* faced Work Interfere 4.0 at 2    CA
Name: state, dtype: object
```

- How many Probaly Mental harrasment faced Patient with "Family Background = NO" took treatment and work interfere happens-

```
dfMayBeOStN = dfMayBe_tot[(dfMayBe_tot['family_history']== 'No') &
(dfMayBe_tot['treatment']== 'Yes') &
(dfMayBe_tot['work_interfere'].isin(['Often',

'Sometimes',

'Rarely']))][['state',

'work_interfere']].copy()
dfMayBeOStN.head()
```

	state	work_interfere
17	TN	Sometimes
22	MA	Often
34	WI	Sometimes
83	NY	Often
88	FL	Sometimes

```
dfMayBeNOStPiv= dfMayBeOStN.groupby(['state','work_interfere'],
observed = False)['work_interfere'].count().reset_index(name =
'count')

dfMayBeNOStPiv.head()
```

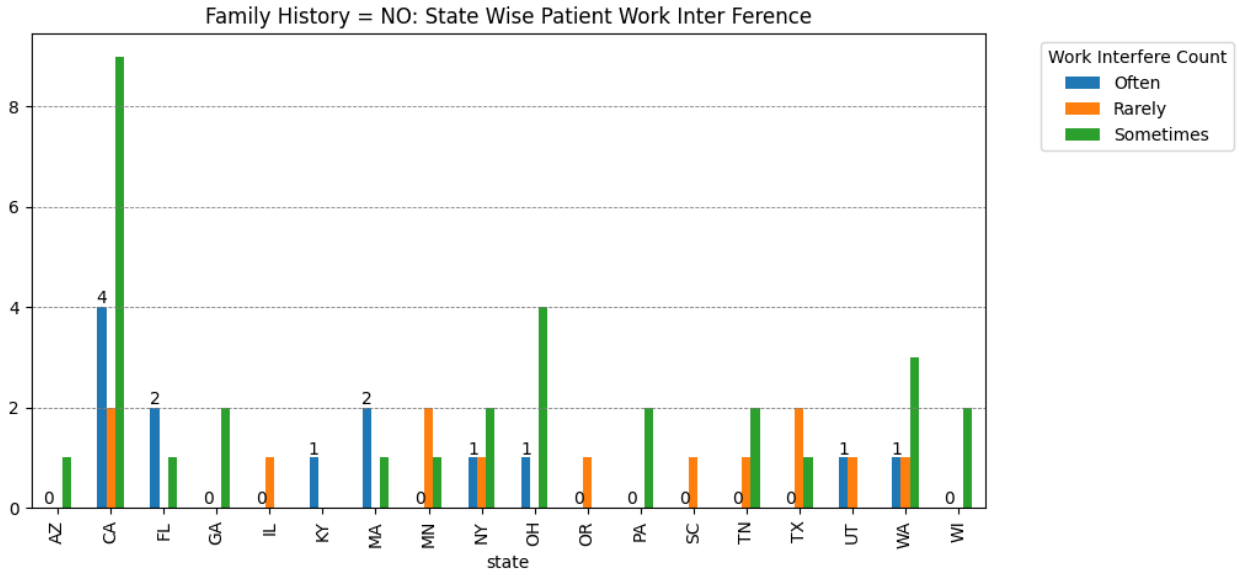
	state	work_interfere	count
0	AZ	Sometimes	1
1	CA	Often	4
2	CA	Rarely	2
3	CA	Sometimes	9
4	FL	Often	2

```
dfMayBeNOStPiv =
MH_pivote(dfMayBeNOStPiv,'work_interfere','state','count')
dfMayBeNOStPiv.head()
```

work_interfere	state	Often	Rarely	Sometimes
0	AZ	NaN	NaN	1.0
1	CA	4.0	2.0	9.0
2	FL	2.0	NaN	1.0
3	GA	NaN	NaN	2.0
4	IL	NaN	1.0	NaN

```
dfMayBeNOStPvVis =bar_function(
    dfB = dfMayBeNOStPiv,
```

```
    colsName1 = 'state',
    colsName2 = ['Often','Rarely','Sometimes'],
    graphKind = 'bar',
    wt = 10,
    ht = 5,
    grphTitle = 'Family History = NO: State Wise Patient Work Inter
Ference',
    legnTitle = 'Work Interfere Count',
    grdAxix = 'y'
)
```



Conclusion:

Among the probaly mental harrasment faced people "Family Background = No" took treatment and work interfere happens-**

In CA,OH,WA people **those does not have patient** from their family background:

- – took treatment and work interfere happens **"sometimes".**

In CA,MA people **those does not have patient** from their family background:

- – took treatment and work interfere happens **"Often" followed by "sometimes".**

```
print("Note: Data Showing For Family Background = NO and May faced
negetive consequences from the employer:")
print()
print(f"Number of people *Often* faced Work Interfere
{dfMayBeNOStPiv['Often'].value_counts().sum()}")
print(f"Max Number of people *Often* faced Work Interfere
{dfMayBeNOStPiv['Often'].max()} at
```

```
{dfMayBeNOStPiv[dfMayBeNOStPiv['Often']==
dfMayBeNOStPiv['Often'].max()]['state']}")
print()

print(f"Number of people *Sometimes* faced Work Interfere
{dfMayBeNOStPiv['Sometimes'].value_counts().sum()}")
print(f"Max Number of people *Sometimes* faced Work Interfere
{dfMayBeNOStPiv['Sometimes'].max()} at
{dfMayBeNOStPiv[dfMayBeNOStPiv['Sometimes']==
dfMayBeNOStPiv['Sometimes'].max()]['state']}")
print()

print(f"Number of people *Rarely* faced Work Interfere
{dfMayBeNOStPiv['Rarely'].value_counts().sum()}")
print(f"Max Number of people *Rarely* faced Work Interfere
{dfMayBeNOStPiv['Rarely'].max()} at
{dfMayBeNOStPiv[dfMayBeNOStPiv['Rarely']==
dfMayBeNOStPiv['Rarely'].max()]['state']}")
```

```
Note: Data Showing For Family Background = NO and May faced negetive
consequences from the employer:

Number of people *Often* faced Work Interfere 8
Max Number of people *Often* faced Work Interfere 4.0 at 1    CA
Name: state, dtype: object

Number of people *Sometimes* faced Work Interfere 13
Max Number of people *Sometimes* faced Work Interfere 9.0 at 1    CA
Name: state, dtype: object

Number of people *Rarely* faced Work Interfere 10
Max Number of people *Rarely* faced Work Interfere 2.0 at 1     CA
7     MN
14    TX
Name: state, dtype: object
```

```
df_WS.columns
```

```
Index(['Timestamp', 'Age', 'Gender', 'Country', 'state',
'self_employed',
       'family_history', 'treatment', 'work_interfere',
'no_employees',
       'remote_work', 'tech_company', 'benefits', 'care_options',
       'wellness_program', 'seek_help', 'anonymity', 'leave',
       'mental_health_consequence', 'phys_health_consequence',
'coworkers',
       'supervisor', 'mental_health_interview',
'phys_health_interview',
       'mental_vs_physical', 'obs_consequence', 'comments',
```

```
'Timestamp_MEfeq'],
      dtype='object')

dfMH_OtPv

treatment work_interfere  treatment_No  treatment_Yes
0                  Never             8              2
1                  Often             2             18
2                 Rarely             1             12
3              Sometimes             6             44

## Conclution:
"""
So suspectd people those are considered as
"probly mental patient" most of them don't have "mental patient"
family member in their family """

' \nSo suspectd people those are considered as \n"probly mental
patient" most of them don\'t have "mental patient" family member in
their family '
```

Finding Probably Mental Patient Work Interfere Stats with ***Family history "Yes":***

```
# df_FNoPiv = bar_function(
#     dfB= df_FNoPiv,
#     colsName1= 'work_interfere',
#     colsName2= ['Yes','No'],
#     graphKind= 'bar',
#     wt= 9,
#     ht= 5,
#     grphTitle= 'Patient Work Interfere With Treatment Stat',
#     legnTitle= 'Treatment Stat',
#     grdAxix= 'y')
```

Attitudes Toward Mental Health:

How people think about, feel about, and respond to mental health issues?

Measured by columns like:

mental_health_consequence: Do people fear negative consequences if they speak up?

coworkers, supervisors: Are they willing to discuss mental health at work?

mental_vs_physical: Do they think mental health is taken as seriously as physical?

benefits, seek_help, anonymity: Do companies support mental health?

```
df_WS.columns
```

```
Index(['Timestamp', 'Age', 'Gender', 'Country', 'state',
'self_employed',
       'family_history', 'treatment', 'work_interfere',
'no_employees',
       'remote_work', 'tech_company', 'benefits', 'care_options',
       'wellness_program', 'seek_help', 'anonymity', 'leave',
       'mental_health_consequence', 'phys_health_consequence',
'coworkers',
       'supervisor', 'mental_health_interview',
'phys_health_interview',
       'mental_vs_physical', 'obs_consequence', 'comments',
'Timestamp_MEfeq'],
      dtype='object')
```

```
df_MentAtiAnalys =df_WS[df_WS['Country'] == 'United States']
[['mental_health_consequence','state','coworkers',

'supervisor','seek_help','benefits','anonymity','Gender','Age']]
df_MentAtiAnalys.head()
```

```
  mental_health_consequence state     coworkers supervisor   seek_help
\
0                        No    IL  Some of them        Yes         Yes

1                     Maybe    IN            No         No  Don't know

4                        No    TX  Some of them        Yes  Don't know

5                        No    TN           Yes        Yes  Don't know

6                     Maybe    MI  Some of them         No          No


     benefits   anonymity  Gender  Age
0         Yes         Yes  Female   37
1  Don't know  Don't know    Male   44
4         Yes  Don't know    Male   31
5         Yes  Don't know    Male   33
6          No          No  Female   35
```

```
df_MentAtiAnalys['Gender'] = df_MentAtiAnalys['Gender'].replace('Male
(CIS)','Male_(CIS)')
```

How people think about, feel about, and respond to mental health issues?

Measured by columns like:

mental_health_consequence: Do people fear negative consequences if they speak up?

This question is previously solved

Now check based on the gender "mental_health_consequence" thinking

**Among the Males maximum people think or they may faced negative consequences.

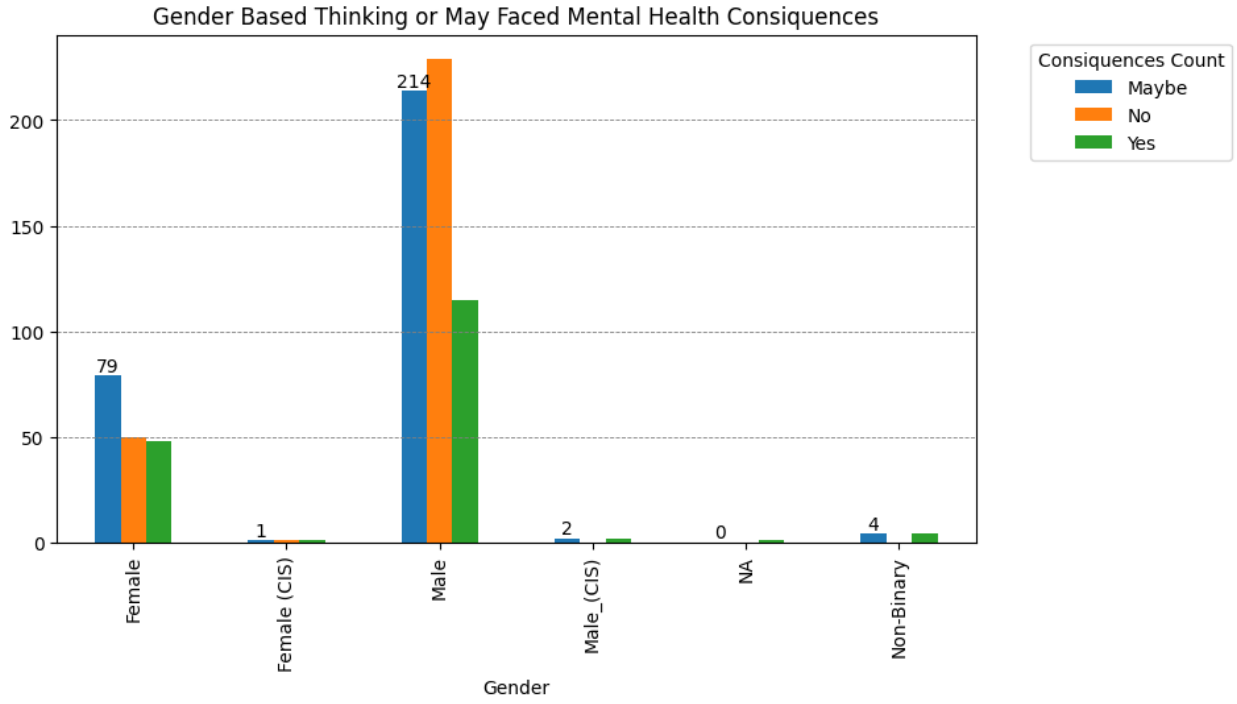
```
df_MntAnalysPiv=
df_MentAtiAnalys.groupby(['Gender','mental_health_consequence'],
observed= False)['mental_health_consequence'].count().reset_index(name
='IndexCount')
df_MntAnalysPiv.head()
```

	Gender	mental_health_consequence	IndexCount
0	Female	Maybe	79
1	Female	No	50
2	Female	Yes	48
3	Female (CIS)	Maybe	1
4	Female (CIS)	No	1

```
df_MntAnalysPiv=MH_pivote(df_MntAnalysPiv,'mental_health_consequence',
'Gender','IndexCount')

df_MntAnalysPiv.head()
```

mental_health_consequence	Gender	Maybe	No	Yes
0	Female	79.0	50.0	48.0
1	Female (CIS)	1.0	1.0	1.0
2	Male	214.0	229.0	115.0
3	Male_(CIS)	2.0	NaN	2.0
4	NA	NaN	NaN	1.0

```
df_MntAnalysPivVis= bar_function(
    dfB = df_MntAnalysPiv,
    colsName1 = 'Gender',
    colsName2 = ['Maybe','No','Yes'],
    graphKind = 'bar',
    wt = 9,
    ht = 5,
    grphTitle = 'Gender Based Thinking or May Faced Mental Health
Consiquences',
    legnTitle = 'Consiquences Count',
    grdAxix = 'y',
)
```

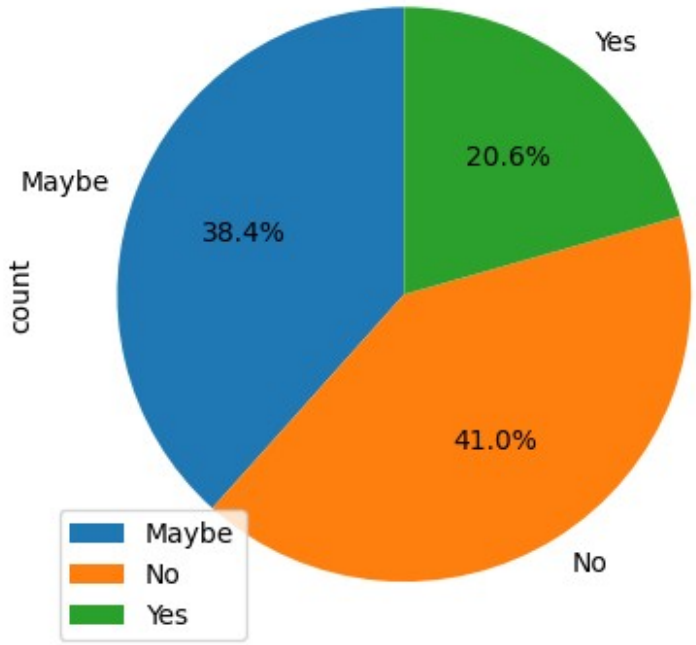



```
#Among Male stat
a=
df_MentAtiAnalys[df_MentAtiAnalys['Gender']=="Male"].groupby('mental_h
ealth_consequence').size().reset_index(name= 'count')
a
```

	mental_health_consequence	count
0	Maybe	214
1	No	229
2	Yes	115

```
a.plot.pie(y = 'count', labels = a['mental_health_consequence'],
           autopct ='%1.1f%%',
           startangle=90)
```

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




```
df_MenAgeAnaly= df_MentAtiAnalys[df_MentAtiAnalys['Gender']=='Male']
[['Age','mental_health_consequence']].copy()
df_MenAgeAnaly.head()
# groupby(['Age','mental_health_consequence'], observed= False)
['mental_health_consequence'].count().reset_index(name ='IndexCount')
# df_MntAnalysPiv.head()
```

	Age	mental_health_consequence
1	44	Maybe
4	31	No
5	33	No
10	31	No
13	36	No

```
df_MenAgeAnaly['Age'].max()
```

```
np.int64(329)
```

```
df_MenAgeAnaly['Age'].min()
```

```
np.int64(-29)
```

```
# Some of the age column values are holding negetive values that's why
lets change it to positive
df_MenAgeAnaly['Age'] =abs(df_MenAgeAnaly['Age'])

#Finding the outliers:
Q1=df_MenAgeAnaly['Age'].quantile(0.25)
```

```
Q3=df_MenAgeAnaly['Age'].quantile(0.75)

IQR = Q3- Q1

lower_bound = Q1- 1.5*IQR
upper_bound = Q3 + 2*IQR

print(lower_bound)
print(upper_bound)

outliers= df_MenAgeAnaly[(df_MenAgeAnaly['Age'] < lower_bound) |
(df_MenAgeAnaly['Age'] > upper_bound)]
outliers
```

```
13.0
58.0
```

	Age	mental_health_consequence
297	60	No
364	329	Maybe
520	62	Maybe
560	65	Maybe
734	5	No
1090	11	No
1236	60	Maybe

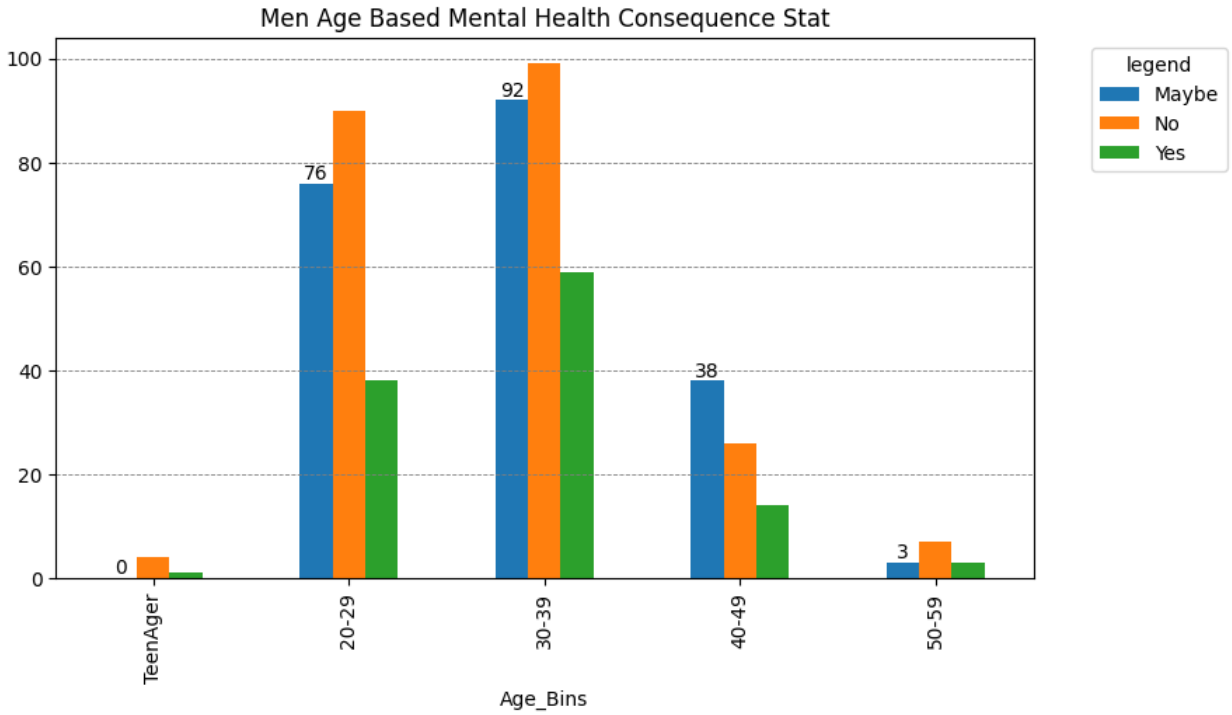
```
#Filtered out the outliers:
df_MenAgeAnalyFil= df_MenAgeAnaly[(df_MenAgeAnaly['Age'] >
lower_bound) & (df_MenAgeAnaly['Age'] < upper_bound)]

#Oldcode:
#In Case labels needs to be based on biological names of the male
# df_MenAgeAnalyFil.loc[:,'Age_Bins']= pd.cut(
#     df_MenAgeAnalyFil['Age'],
#     bins=[15, 20, 30, 40, 50, 60 ],
#     labels=['TeenAgers', 'vicenarian', 'Tricenarian',
'quadragenarian', 'quinquagenarian'])


df_MenAgeAnalyFil.loc[:,'Age_Bins']= pd.cut(
    df_MenAgeAnalyFil['Age'],
    bins=[15, 20, 30, 40, 50, 60 ],
    labels=['TeenAger', '20-29', '30-39', '40-49', '50-59'])
```

```
C:\Users\user\AppData\Local\Temp\ipykernel_7572\1766872679.py:9:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
```

```
returning-a-view-versus-a-copy
  df_MenAgeAnalyFil.loc[:,'Age_Bins']= pd.cut(

df_MenAgeAnalyFilPiv = df_MenAgeAnalyFil.groupby(['Age_Bins',
'mental_health_consequence'], observed = False)
['mental_health_consequence'].count().reset_index(name = 'count')

df_MenAgeAnalyFilPiv
=MH_pivote(df_MenAgeAnalyFilPiv,'mental_health_consequence',
'Age_Bins','count')

df_MenAgeAnalyFilPiv.head(7)
```

mental_health_consequence	Age_Bins	Maybe	No	Yes
0	TeenAger	0	4	1
1	20-29	76	90	38
2	30-39	92	99	59
3	40-49	38	26	14
4	50-59	3	7	3

```
df_MenAgeAnalyFilPivH= bar_function(
    dfB= df_MenAgeAnalyFilPiv,
    colsName1 = 'Age_Bins',
    colsName2 = ['Maybe','No','Yes'],
    graphKind = 'bar',
    wt = 9,
    ht = 5,
    grphTitle = 'Men Age Based Mental Health Consequence Stat',
    legnTitle = 'legend',
    grdAxix = 'y',)
```



Conclusion:

The most younger men (20-39) think or may faced Mental Health Consequence from employer.

Within this (20-39) age group people think will or might be face mental health consequence from employer.

People think or faced from coworkers, supervisors about discussing on mental health at work

```
df_MentAtiAnalys.head()
```

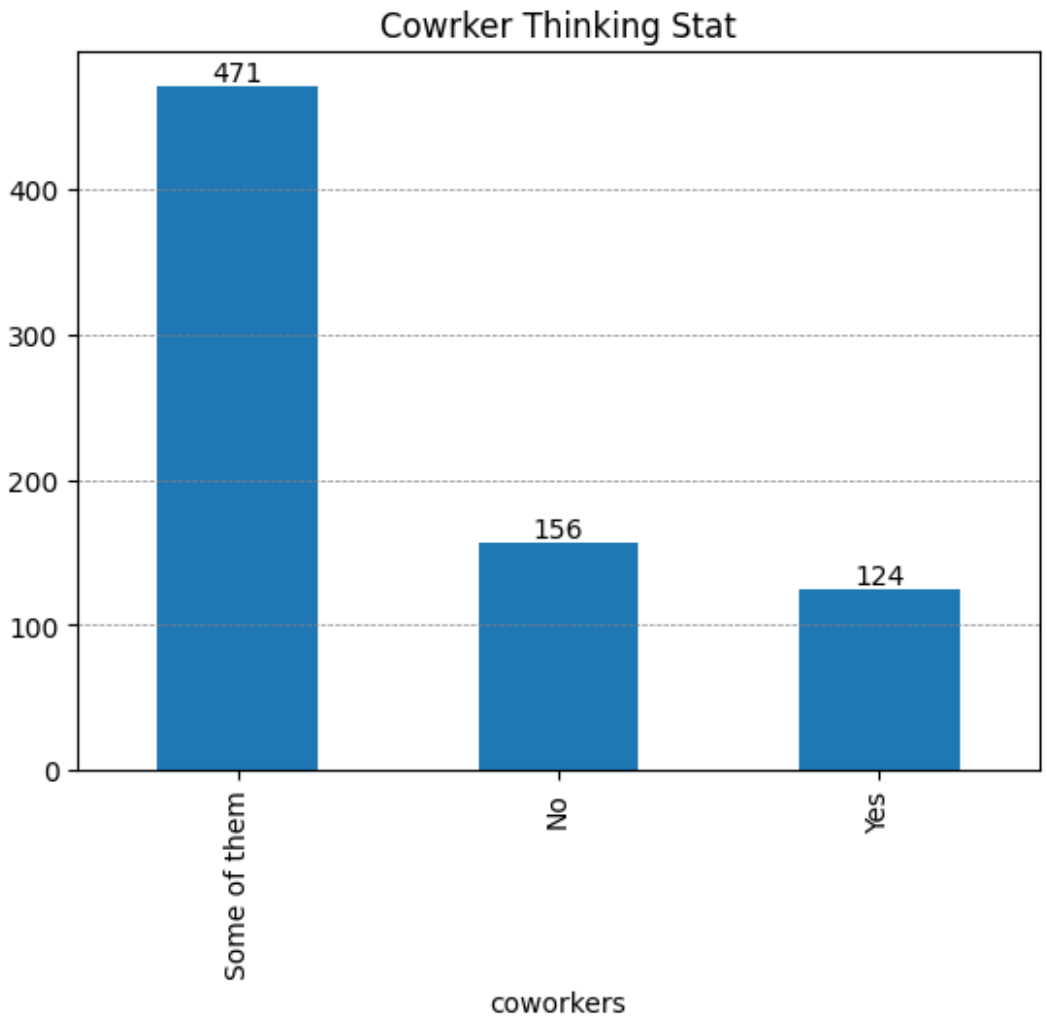
```
  mental_health_consequence state     coworkers supervisor   seek_help
\
0                        No    IL  Some of them        Yes          Yes

1                     Maybe    IN            No         No  Don't know

4                        No    TX  Some of them        Yes  Don't know

5                        No    TN           Yes        Yes  Don't know

6                     Maybe    MI  Some of them         No           No


     benefits   anonymity  Gender  Age
0         Yes         Yes  Female   37
1  Don't know  Don't know    Male   44
4         Yes  Don't know    Male   31
```

```
5         Yes  Don't know    Male   33
6          No          No  Female   35
```

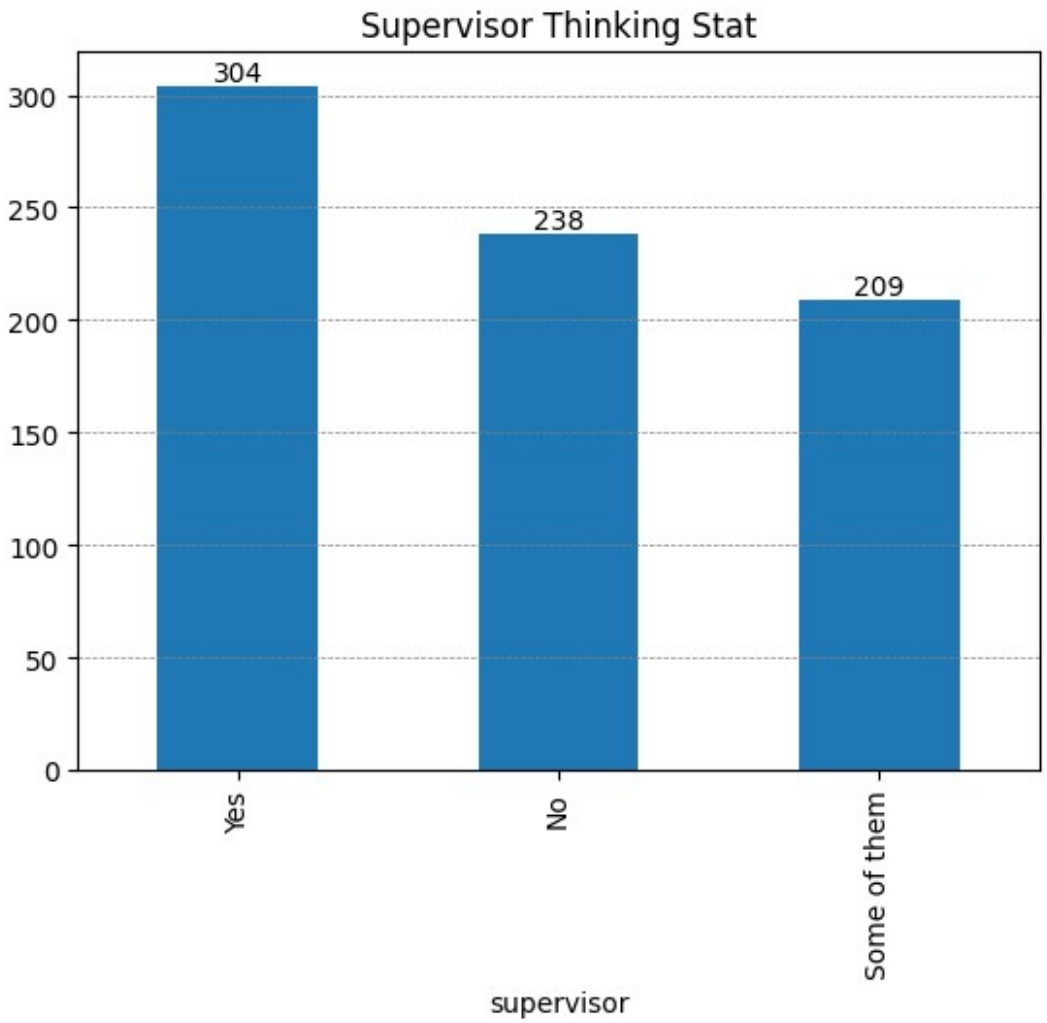
```
df_coWrkSup=
df_MentAtiAnalys[['Gender','coworkers','supervisor','state']].copy()
df_coWrkSup.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 751 entries, 0 to 1258
Data columns (total 4 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   Gender      751 non-null    object
 1   coworkers   751 non-null    object
 2   supervisor  751 non-null    object
 3   state       751 non-null    object
dtypes: object(4)
memory usage: 29.3+ KB
```

```
df_coWrkVis= df_coWrkSup['coworkers'].value_counts().plot.bar()
plt.title('Cowrker Thinking Stat')
plt.grid(visible= True, axis = 'y', color ='gray', linestyle ='--',
linewidth = 0.5)

for cols in df_coWrkVis.containers:
    df_coWrkVis.bar_label( cols, label_type= 'edge')
```




```
df_SupWrkVis= df_coWrkSup['supervisor'].value_counts().plot.bar()
plt.title('Supervisor Thinking Stat')
plt.grid(visible= True, axis = 'y', color ='gray', linestyle ='--',
linewidth = 0.5)

for cols in df_SupWrkVis.containers:
    df_SupWrkVis.bar_label( cols, label_type= 'edge')
```



Conclusion:

From the above two graph the it clearly visible supervisors (304) are more prone towards discussing about the employees mental health condition on work places.

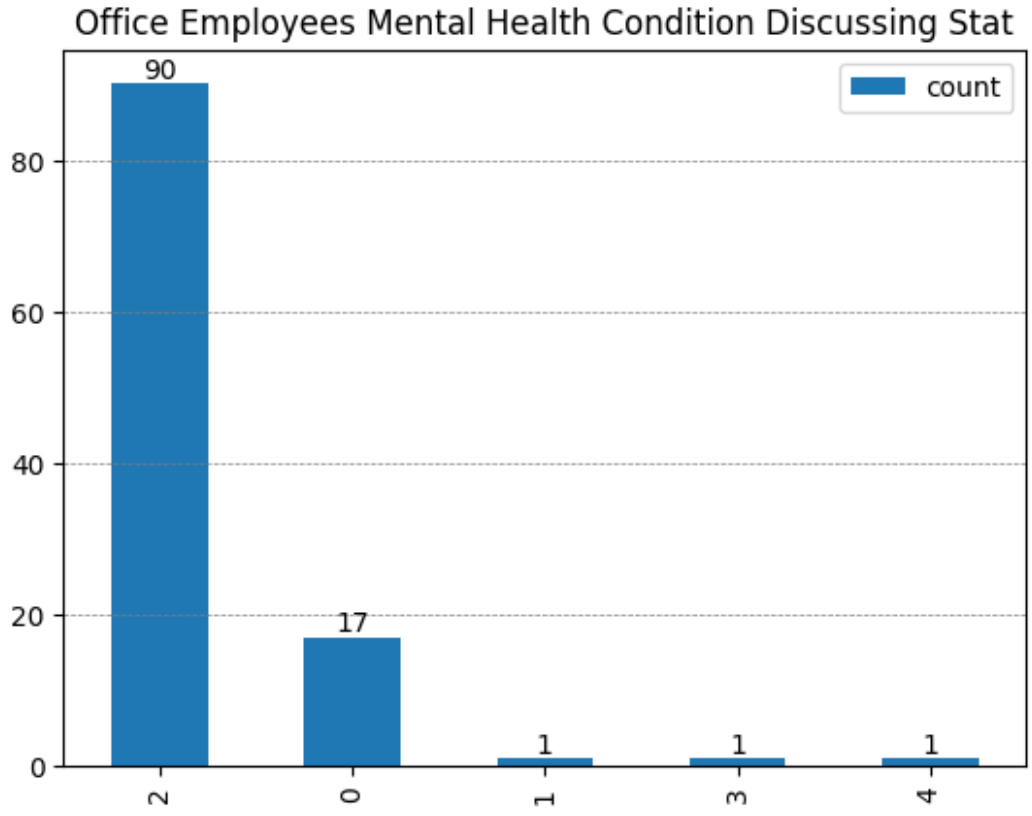
In how many cases supervisor and coworker both are discussing about the employees mental health condition on work places? Ans : 110

df_coWrkSup[(df_coWrkSup['coworkers'] =='Yes') & (df_coWrkSup['supervisor'] =='Yes')] = 110

```
# Based on the geneder:
### Cases supervisor and coworker both are discussing about the
employees mental health condition on work places

df_coWkSpVs= df_coWrkSup[(df_coWrkSup['coworkers'] =='Yes') &
(df_coWrkSup['supervisor']
=='Yes')].groupby(['Gender']).size().reset_index(name = 'count')
df_coWkSpVs.sort_values(by ='count', ascending = False)
```

```
          Gender  count
2           Male     90
0         Female     17
1  Female (CIS)      1
3             NA      1
4     Non-Binary      1
```

```
df_coWkSpVsBr= df_coWkSpVs.sort_values(by ='count', ascending =
False).plot.bar()
plt.title('Office Employees Mental Health Condition Discussing Stat')
plt.grid(visible= True, axis = 'y', color ='gray', linestyle ='--',
linewidth = 0.5)

for cols in df_coWkSpVsBr.containers:
    df_coWkSpVsBr.bar_label( cols, label_type= 'edge')
```



Do companies support mental health? benefits, seek_help, anonymity:

```
#Finding How many employer provides benefits for mental health patient
df_MentAtiAnalys['benefits'].value_counts()
```

```
benefits
Yes           398
Don't know    236
```

```
No            117
Name: count, dtype: int64
```

```
print(f'Employer provides benefits:
{df_MentAtiAnalys.loc[df_MentAtiAnalys['benefits']== 'Yes' ,
['benefits']].value_counts().sum()}')
print(f'Employer not provided any benefits:
{df_MentAtiAnalys.loc[df_MentAtiAnalys['benefits']== 'No' ,
['benefits']].value_counts().sum()}')
print(f'Don\'t know Employer provides benefits or not:
{df_MentAtiAnalys.loc[df_MentAtiAnalys['benefits']== 'Don\'t know' ,
['benefits']].value_counts().sum()}')
```

```
Employer provides benefits: 398
Employer not provided any benefits: 117
Don't know Employer provides benefits or not: 236
```

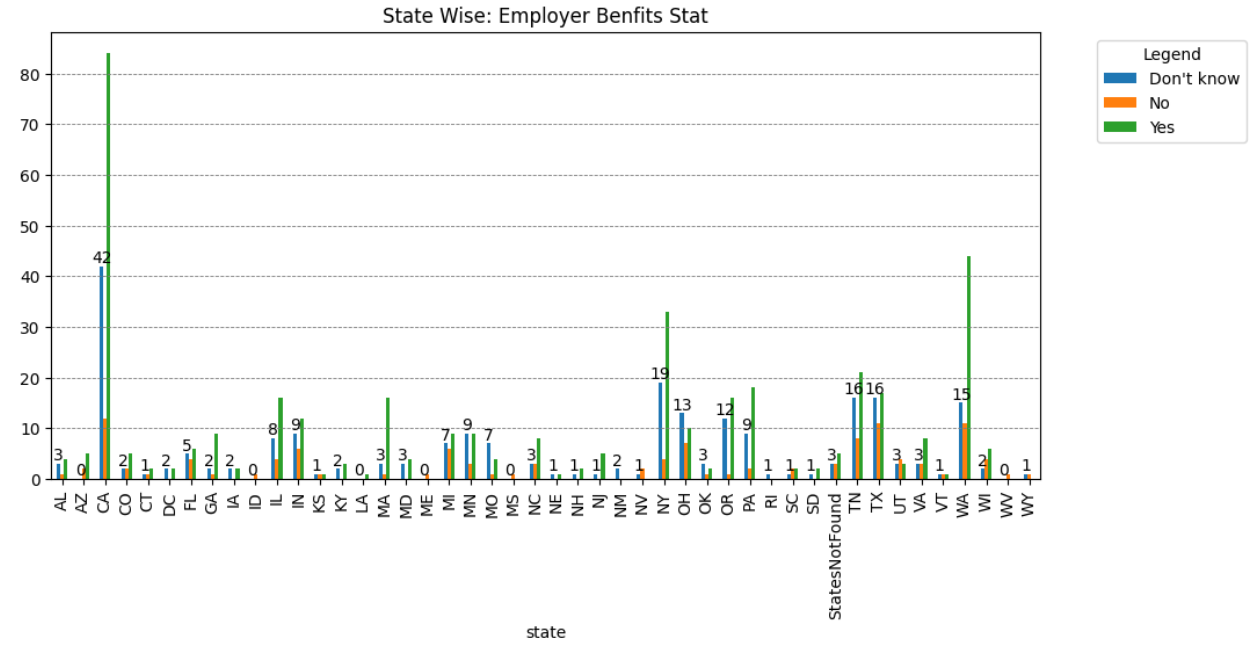
```
df_StAnalysBen= df_MentAtiAnalys.groupby(['state', 'benefits'],
observed= False)['benefits'].size().reset_index(name ='count')
df_StAnalysBen.head()
```

```
  state    benefits  count
0    AL  Don't know      3
1    AL          No      1
2    AL         Yes      4
3    AZ          No      2
4    AZ         Yes      5
```

```
df_StAnalysBen =MH_pivote(df_StAnalysBen, 'benefits','state','count')
```

```
df_StAnalysBen = df_StAnalysBen.fillna(value = 0)
```

```
df_StAnalysBenVis =bar_function(
    dfB = df_StAnalysBen,
    colsName1 = 'state',
    colsName2 = ['Don\'t know','No','Yes'],
    graphKind = 'bar',
    wt = 11,
    ht = 5,
    grphTitle = 'State Wise: Employer Benfits Stat',
    legnTitle = 'Legend',
    grdAxix = 'y')
```




```
df_StAnalysBen.head()
```

benefits state	Don't know	No	Yes
0 AL	3.0	1.0	4.0
1 AZ	0.0	2.0	5.0
2 CA	42.0	12.0	84.0
3 CO	2.0	2.0	5.0
4 CT	1.0	1.0	2.0

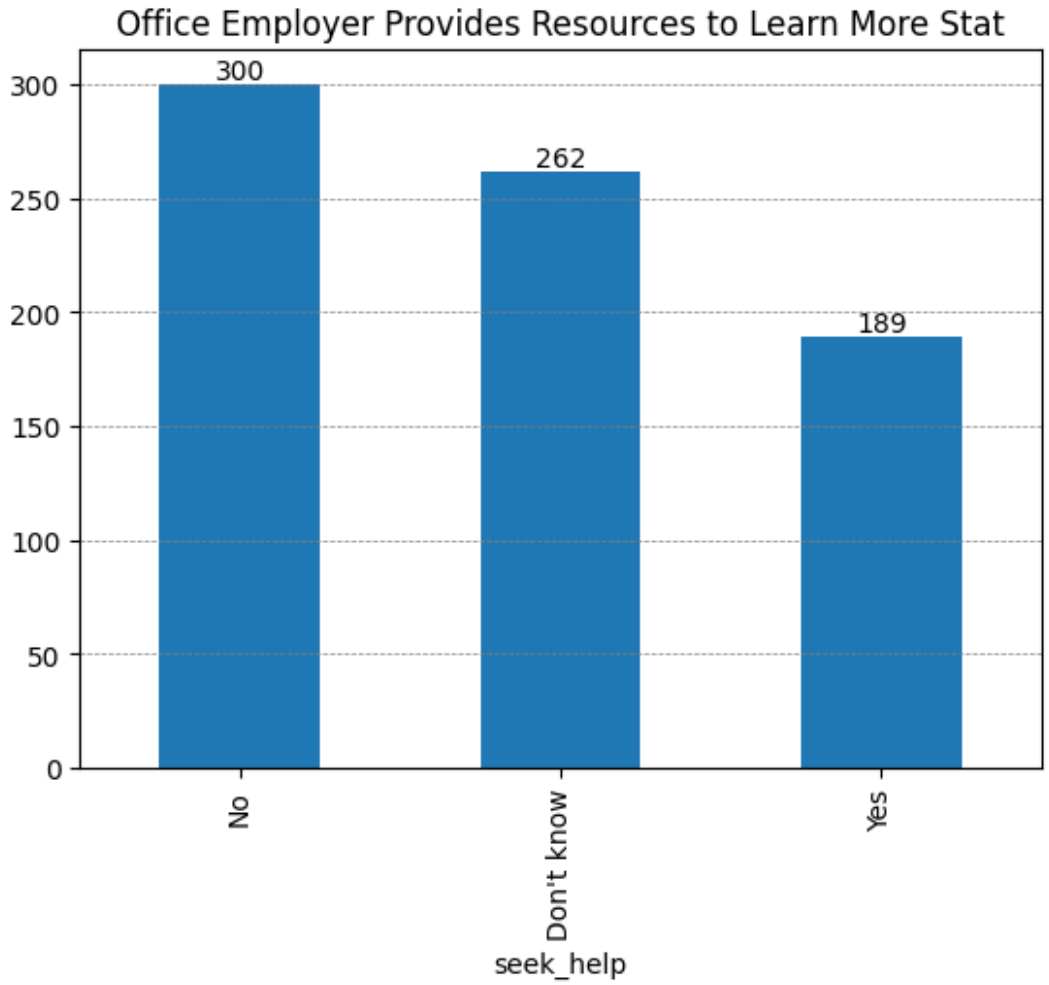
```
print()
print(f"Max Number of people get benefits from employer is
{df_StAnalysBen['Yes'].max()} in
{df_StAnalysBen[df_StAnalysBen['Yes']== df_StAnalysBen['Yes'].max()]
['state']}")
print()
print(f"At the same state {df_StAnalysBen[df_StAnalysBen['Yes']==
df_StAnalysBen['Yes'].max()]['state']}
{df_StAnalysBen[df_StAnalysBen['Yes']== df_StAnalysBen['Yes'].max()]
['No'].value_counts().sum} number of people does not get benefit from
employer")
```

```
Max Number of people get benefits from employer is 84.0 in 2    CA
Name: state, dtype: object

At the same state 2    CA
Name: state, dtype: object <bound method Series.sum of No
12.0    1
Name: count, dtype: int64> number of people does not get benefit from
employer
```

```
# seek_help, anonymity stat

### "seek_help" means Does your employer provide resources to learn more about mental health issues and how to seek help?
df_MentAtiAnalys['seek_help'].value_counts()
```

```
seek_help
No            300
Don't know    262
Yes           189
Name: count, dtype: int64
```

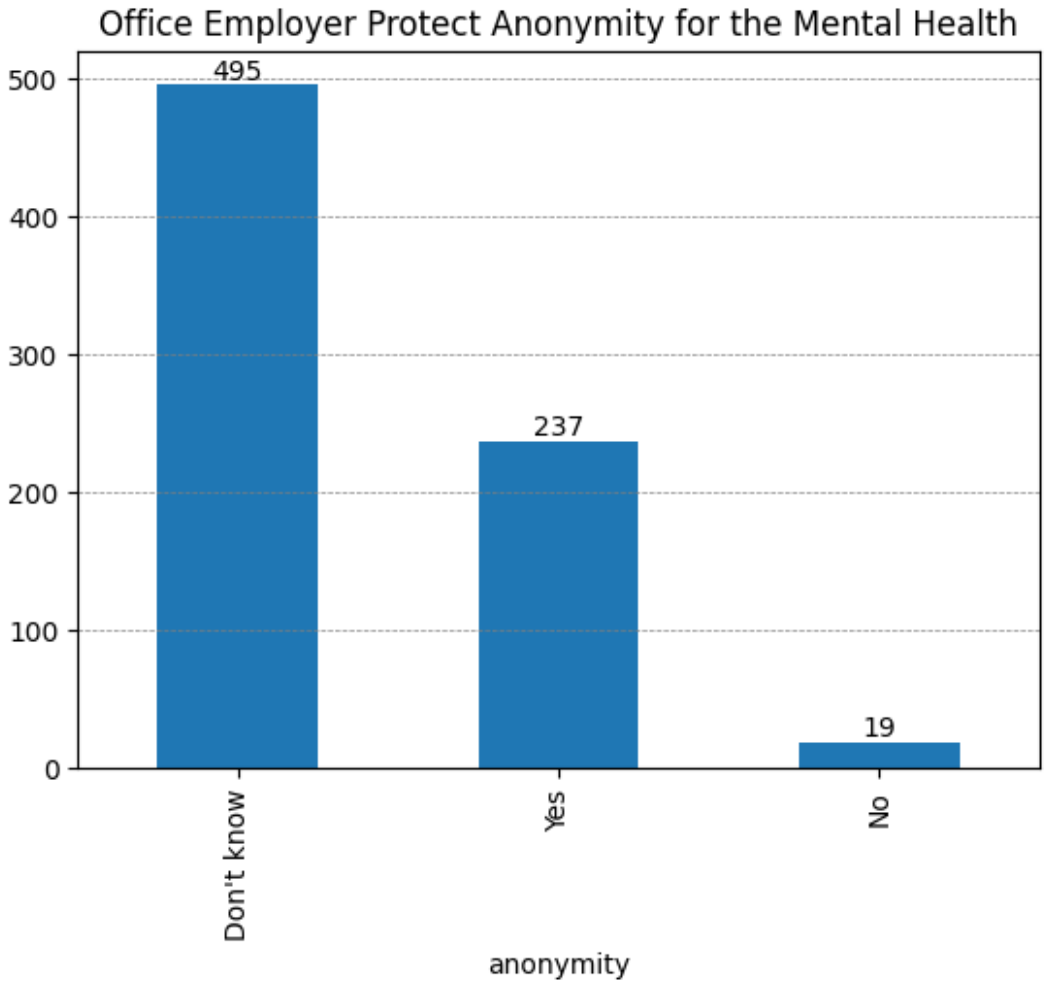
```
df_seHpVsBr= df_MentAtiAnalys['seek_help'].value_counts().plot.bar()
plt.title('Office Employer Provides Resources to Learn More Stat')
plt.grid(visible= True, axis = 'y', color ='gray', linestyle ='--',
linewidth = 0.5)

for cols in df_seHpVsBr.containers:
    df_seHpVsBr.bar_label( cols, label_type= 'edge')
```



```
## "anonymity" means - Is your anonymity protected if you choose to
take advantage of mental health or substance abuse treatment
resources?
df_MentAtiAnalys['anonymity'].value_counts()
```

```
anonymity
Don't know    495
Yes           237
No             19
Name: count, dtype: int64
```

```
df_AnoVsBr= df_MentAtiAnalys['anonymity'].value_counts().plot.bar()
plt.title('Office Employer Protect Anonymity for the Mental Health')
plt.grid(visible= True, axis = 'y', color ='gray', linestyle ='--',
linewidth = 0.5)

for cols in df_AnoVsBr.containers:
    df_AnoVsBr.bar_label( cols, label_type= 'edge')
```



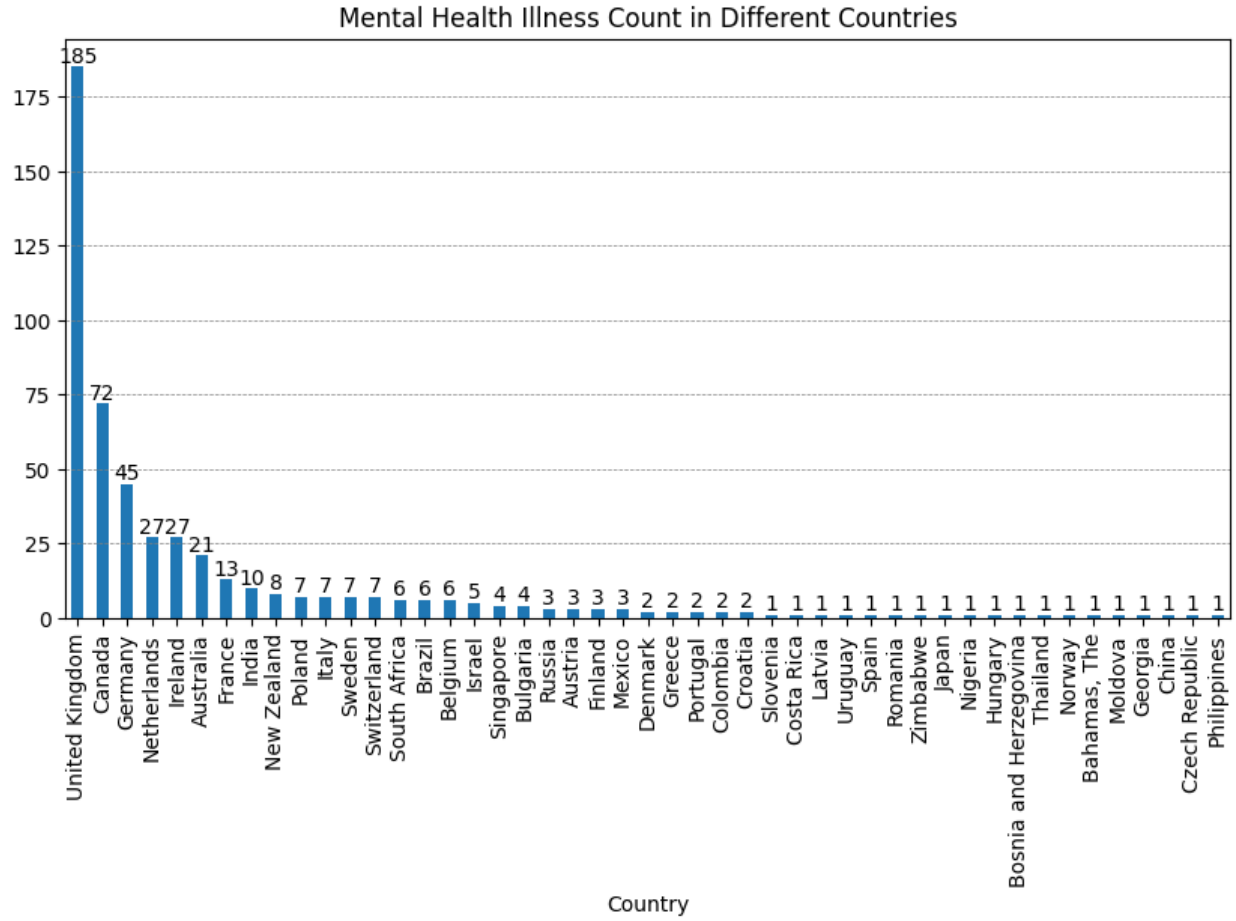
Out Side US Data

```
df_NonUS= df_WS[df_WS['Country'] !='United States'].copy()
df_NonUS['Country'].value_counts()
```

```
Country
United Kingdom             185
Canada                      72
Germany                     45
Netherlands                 27
Ireland                     27
Australia                   21
France                      13
India                       10
New Zealand                  8
Poland                       7
Italy                        7
Sweden                       7
Switzerland                  7
South Africa                 6
Brazil                       6
Belgium                      6
Israel                       5
Singapore                    4
Bulgaria                     4
Russia                       3
Austria                      3
Finland                      3
Mexico                       3
Denmark                      2
Greece                       2
Portugal                     2
Colombia                     2
Croatia                      2
Slovenia                     1
Costa Rica                   1
Latvia                       1
Uruguay                      1
Spain                        1
Romania                      1
Zimbabwe                     1
Japan                        1
Nigeria                      1
Hungary                      1
Bosnia and Herzegovina       1
Thailand                     1
Norway                       1
Bahamas, The                 1
Moldova                      1
Georgia                      1
```

```
China                          1
Czech Republic                 1
Philippines                    1
Name: count, dtype: int64
```

```
df_NusVis= df_WS[df_WS['Country'] !='United States']
['Country'].value_counts().plot.bar(figsize = (10,5))
plt.title('Mental Health Illness Count in Different Countries')
plt.grid(visible= True, axis = 'y', color ='gray', linestyle ='--',
linewidth = 0.5)

for cols in df_NusVis.containers:
    df_NusVis.bar_label( cols, label_type= 'edge')
```




```
## Measured by treatment

df_NonUStrPiv = df_NonUS[df_NonUS['Country'] !='United States']
[['Country',

'treatment']].groupby(['Country',
```

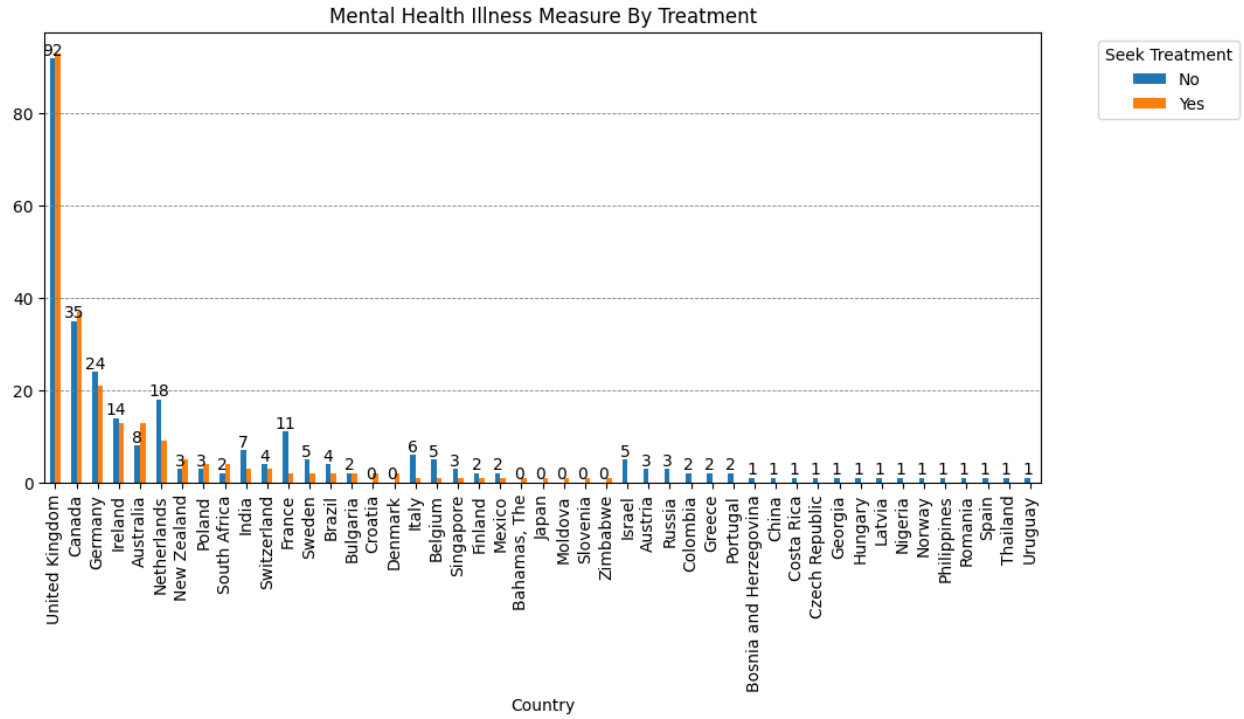
```
'treatment'],observed = False) ['treatment'].count().reset_index(name
= 'count')

df_NonUStrPiv.head()
```

```
        Country treatment  count
0     Australia        No      8
1     Australia       Yes     13
2       Austria        No      3
3  Bahamas, The       Yes      1
4       Belgium        No      5
```

```
df_NonUStrPiv =MH_pivote(df_NonUStrPiv,'treatment','Country','count')

df_NonUStrPiv.head()
```

```
treatment                  Country   No   Yes
0                        Australia  8.0  13.0
1                          Austria  3.0   NaN
2                       Bahamas, The  NaN   1.0
3                          Belgium  5.0   1.0
4            Bosnia and Herzegovina  1.0   NaN
```

```
df_NonUStrPiv[['No','Yes']] = df_NonUStrPiv[['No','Yes']].fillna(value
= 0)

df_NonUStrPiv[['No','Yes']] =
df_NonUStrPiv[['No','Yes']].astype('int64')

df_NonUStrPiv =df_NonUStrPiv.sort_values(by= ['Yes','No'], ascending =
False)

df_NonUStrPivVisbG = bar_function(
    dfB = df_NonUStrPiv,
    colsName1 = 'Country',
    colsName2 = ['No','Yes'],
    graphKind = 'bar',
    wt = 11,
    ht = 5,
    grphTitle = 'Mental Health Illness Measure By Treatment',
    legnTitle = 'Seek Treatment',
    grdAxix = 'y')
```



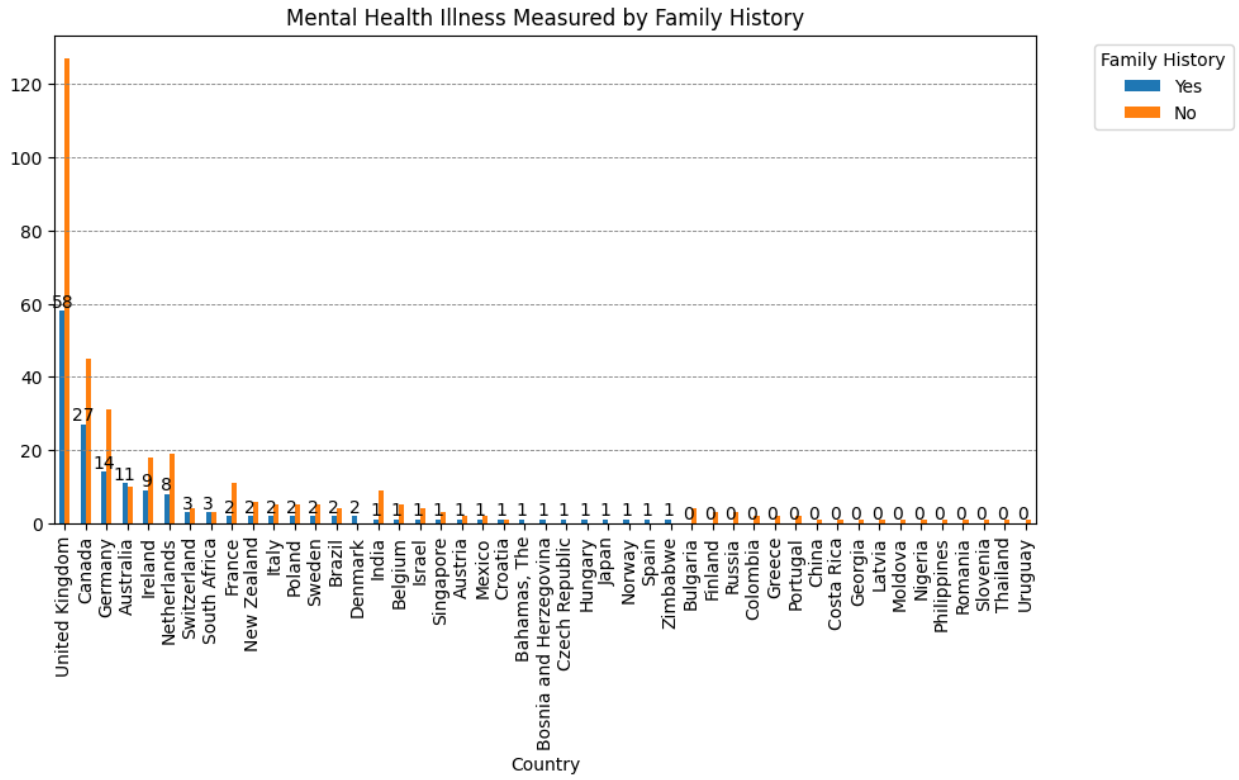

```
## Measured by family_history

df_NonUSfmlyHis=df_NonUS[['Country','family_history']].groupby(['Country',
                                                    'family_history'],
observed= False )['family_history'].count().reset_index(name =
'count')
df_NonUSfmlyHis.head()
```

	Country	family_history	count
0	Australia	No	10
1	Australia	Yes	11
2	Austria	No	2
3	Austria	Yes	1
4	Bahamas, The	Yes	1

```
df_NonUSfmlyHisPiv =
MH_pivote(df_NonUSfmlyHis,'family_history','Country','count')

df_NonUSfmlyHisPiv = df_NonUSfmlyHisPiv.sort_values(by= ['Yes','No'],
ascending = False)
df_NonUSfmlyHisPiv.head()
```

family_history	Country	No	Yes
44	United Kingdom	127.0	58.0
7	Canada	45.0	27.0
17	Germany	31.0	14.0

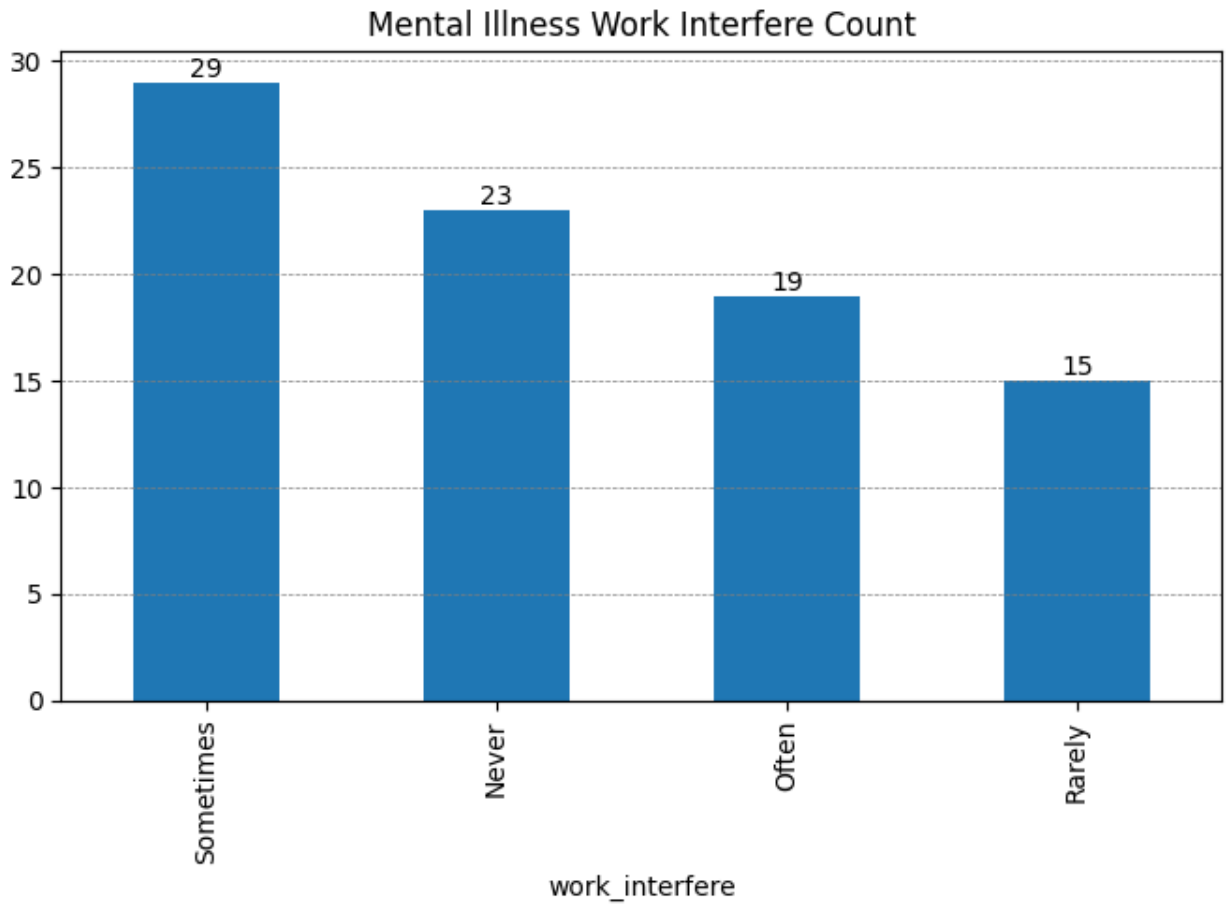
```
0                    Australia   10.0  11.0
21                     Ireland   18.0   9.0

df_NonUSfmlyHisPiv[['No','Yes']] =
df_NonUSfmlyHisPiv[['No','Yes']].fillna(value = 0)

df_NonUSfmlyHisPiv[['No','Yes']] =
df_NonUSfmlyHisPiv[['No','Yes']].astype('int64')

df_NonUSfmlyHisPiv.info()

<class 'pandas.core.frame.DataFrame'>
Index: 47 entries, 44 to 45
Data columns (total 3 columns):
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   Country  47 non-null     object
 1   No       47 non-null     int64
 2   Yes      47 non-null     int64
dtypes: int64(2), object(1)
memory usage: 1.5+ KB

df_NUSfmlyHisPivVisual =bar_function(
    dfB = df_NonUSfmlyHisPiv,
    colsName1 = 'Country',
    colsName2 = ['Yes','No'],
    graphKind = 'bar',
    wt = 10,
    ht = 5,
    grphTitle = 'Mental Health Illness Measured by Family History',
    legnTitle = 'Family History',
    grdAxix = 'y')
```




```
## Measured by work_interfere

df_NonUSinWrk=
df_NonUS[['Country','work_interfere']].groupby(['Country','work_interf
ere'], observed = False)['work_interfere'].count().reset_index(name =
'count')

df_NonUSinWrk.head()
```

	Country	work_interfere	count
0	Australia	Never	3
1	Australia	Often	5
2	Australia	Rarely	2
3	Australia	Sometimes	10
4	Austria	Sometimes	1

```
df_inWrkNonUS =
df_NonUSinWrk['work_interfere'].value_counts().plot.bar(figsize =
(8,4.5))
plt.title('Mental Illness Work Interfere Count')
plt.grid(visible= True, axis = 'y', color ='gray', linestyle ='--',
linewidth = 0.5)

for cols in df_inWrkNonUS.containers:
    df_inWrkNonUS.bar_label(cols, label_type= 'edge')
```



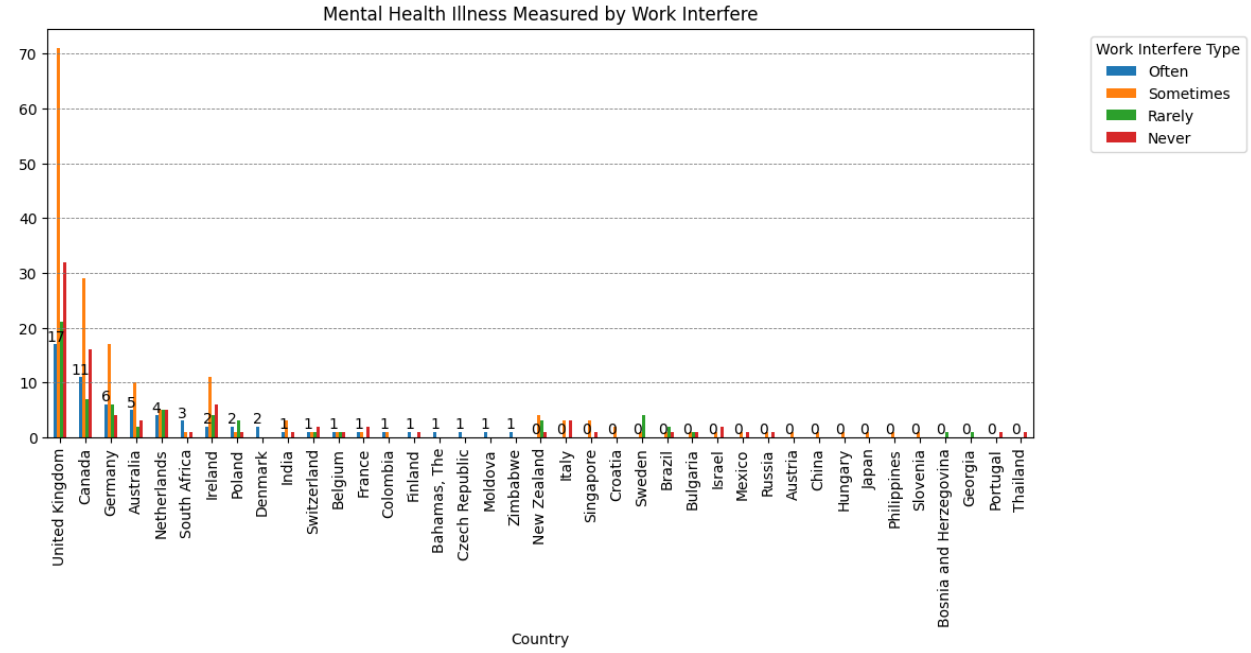

```
df_NonUSinWrkPiv = MH_pivote(df_NonUSinWrk,
'work_interfere','Country','count')

df_NonUSinWrkPiv.head()
```

work_interfere	Country	Never	Often	Rarely	Sometimes
0	Australia	3.0	5.0	2.0	10.0
1	Austria	NaN	NaN	NaN	1.0
2	Bahamas, The	NaN	1.0	NaN	NaN
3	Belgium	1.0	1.0	1.0	1.0
4	Bosnia and Herzegovina	NaN	NaN	1.0	NaN

```
df_NonUSinWrkPiv.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 39 entries, 0 to 38
```

```
Data columns (total 5 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Country    39 non-null     object
 1   Never      23 non-null     float64
 2   Often      19 non-null     float64
 3   Rarely     15 non-null     float64
 4   Sometimes  29 non-null     float64
dtypes: float64(4), object(1)
memory usage: 1.7+ KB
```

```
df_NonUSinWrkPiv[['Often','Sometimes','Rarely','Never']] =
df_NonUSinWrkPiv[['Often','Sometimes','Rarely','Never']].fillna(value
= 0)

df_NonUSinWrkPiv[['Often','Sometimes','Rarely','Never']] =
df_NonUSinWrkPiv[['Often','Sometimes','Rarely','Never']].astype('int64
')

df_NonUSinWrkPiv = df_NonUSinWrkPiv.sort_values(by =
['Often','Sometimes','Rarely','Never'], ascending = False)

df_VisinWrkPivNUS= bar_function(
    dfB = df_NonUSinWrkPiv,
    colsName1 = 'Country',
    colsName2 = ['Often','Sometimes','Rarely','Never'],
    graphKind = 'bar',
    wt = 12,
    ht = 5,
    grphTitle = 'Mental Health Illness Measured by Work Interfere',
    legnTitle = 'Work Interfere Type',
    grdAxix = 'y'
)
```




```
## Frequency of mental health illness and attitudes towards mental
health vary by geographic location?"
### How people think about, feel about, and respond to mental health
issues

## Measured by mental_health_consequence:
##Do people fear negative consequences if they speak up? based on the
experience

df_NonUSgrP= df_NonUS[['Country',
          'mental_health_consequence']].groupby(['Country',

'mental_health_consequence'],
                                                observed= False)
['mental_health_consequence'].count().reset_index(name = 'count')
df_NonUSgrP.head()
```

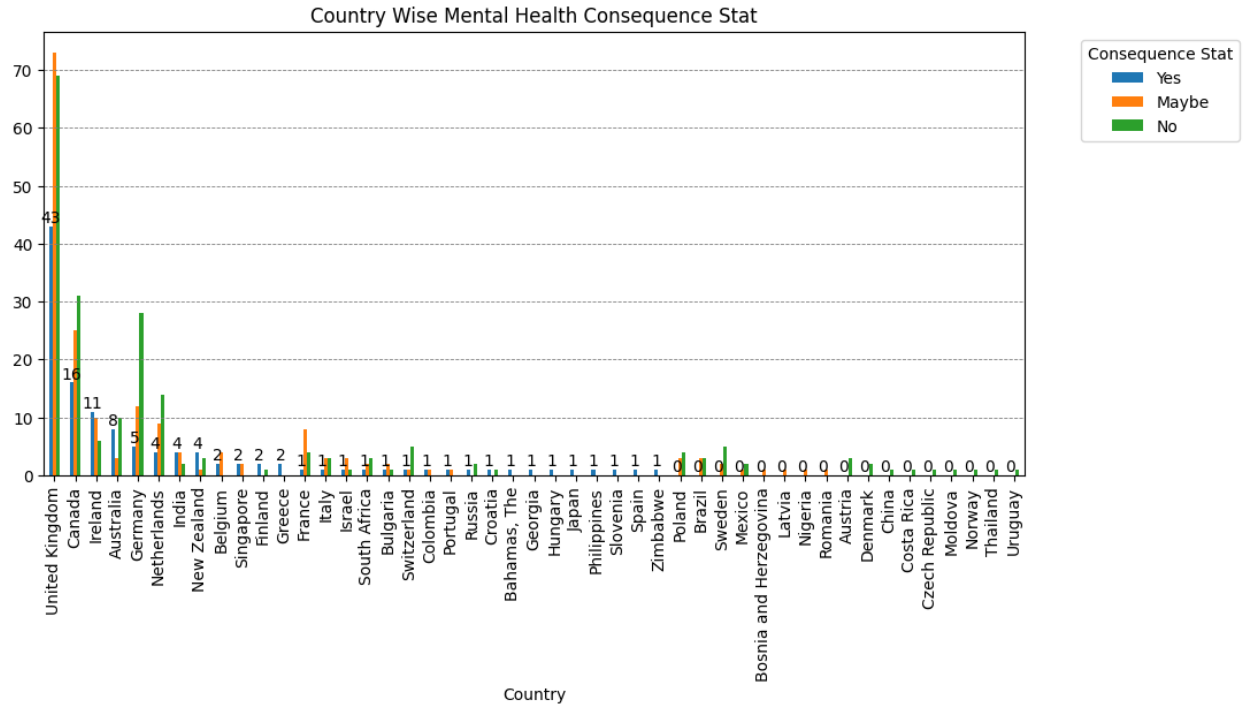
	Country	mental_health_consequence	count
0	Australia	Maybe	3
1	Australia	No	10
2	Australia	Yes	8
3	Austria	No	3
4	Bahamas, The	Yes	1

```
df_NonUSgrPiV=
MH_pivote(df_NonUSgrP,'mental_health_consequence','Country','count')

df_NonUSgrPiV.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 47 entries, 0 to 46
```

```
Data columns (total 4 columns):
 #   Column   Non-Null Count  Dtype  
---  ------   --------------  -----  
 0   Country  47 non-null     object 
 1   Maybe    26 non-null     float64
 2   No       30 non-null     float64
 3   Yes      30 non-null     float64
dtypes: float64(3), object(1)
memory usage: 1.6+ KB
```

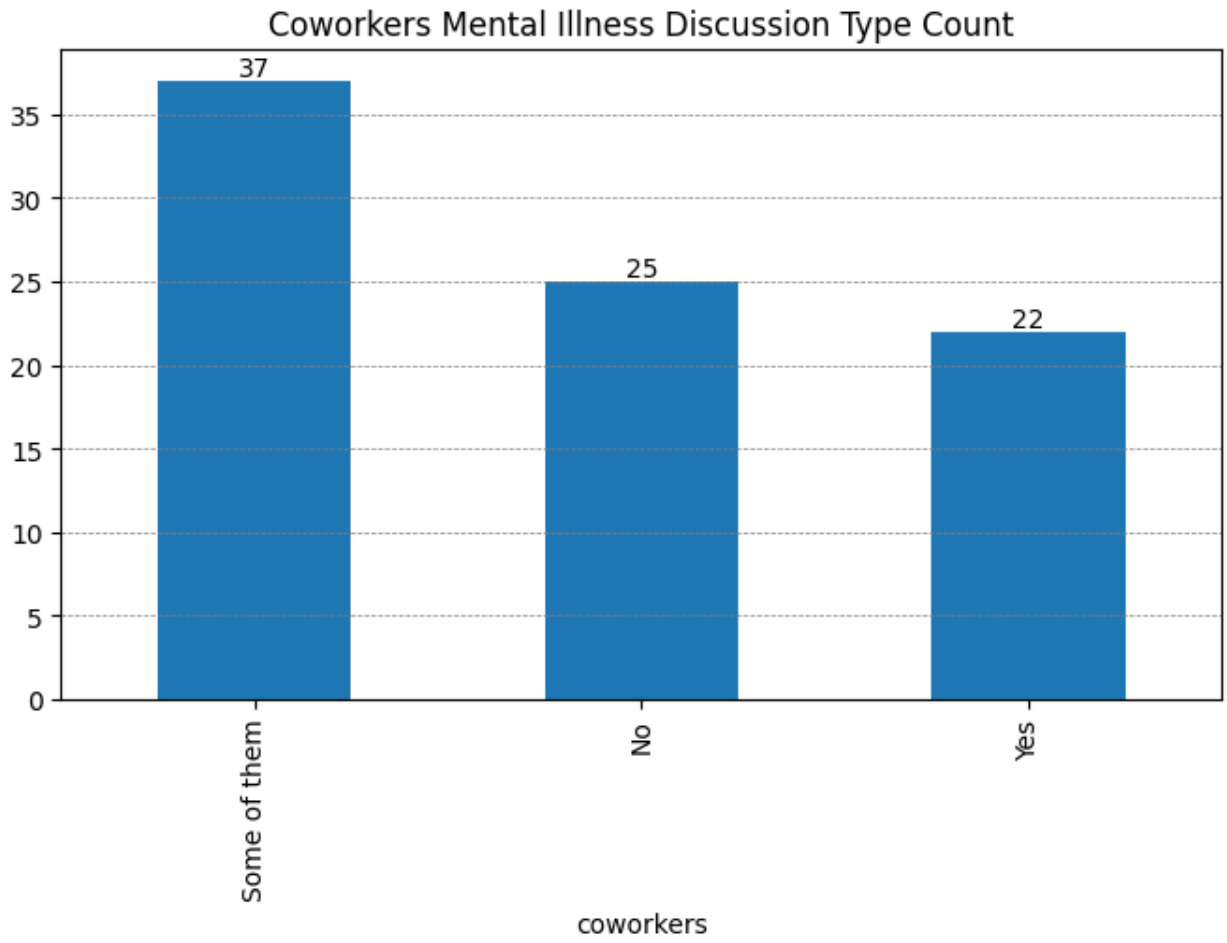
```
df_NonUSgrPiV[['Yes','Maybe','No']] =
df_NonUSgrPiV[['Yes','Maybe','No']].fillna(value = 0)

df_NonUSgrPiV[['Yes','Maybe','No']] =
df_NonUSgrPiV[['Yes','Maybe','No']].astype('int64')

df_NonUSgrPiV= df_NonUSgrPiV.sort_values(by = ['Yes','Maybe','No'],
ascending = False)

df_NonUSgrPviS = bar_function(
    dfB = df_NonUSgrPiV,
    colsName1 = 'Country',
    colsName2 = ['Yes','Maybe','No'],
    graphKind = 'bar',
    wt = 11,
    ht = 5,
    grphTitle = 'Country Wise Mental Health Consequence Stat',
    legnTitle = 'Consequence Stat',
    grdAxix = 'y'
)
```




```
### Measured by coworkers, supervisors: Are they willing to discuss
mental health at work?

df_CoWrNnUS =
df_NonUS[['Country','coworkers']].groupby(['Country','coworkers'],
observed = False)['coworkers'].count().reset_index(name = 'count')
df_CoWrNnUS.head()
```

	Country	coworkers	count
0	Australia	No	6
1	Australia	Some of them	11
2	Australia	Yes	4
3	Austria	Some of them	2
4	Austria	Yes	1

```
df_CoWrNnUS_1v =
df_CoWrNnUS['coworkers'].value_counts().plot.bar(figsize = (8,4.5))
plt.title('Coworkers Mental Illness Discussion Type Count')
plt.grid(visible= True, axis = 'y', color ='gray', linestyle ='--',
linewidth = 0.5)

for cols in df_CoWrNnUS_1v.containers:
    df_CoWrNnUS_1v.bar_label(cols, label_type= 'edge')
```

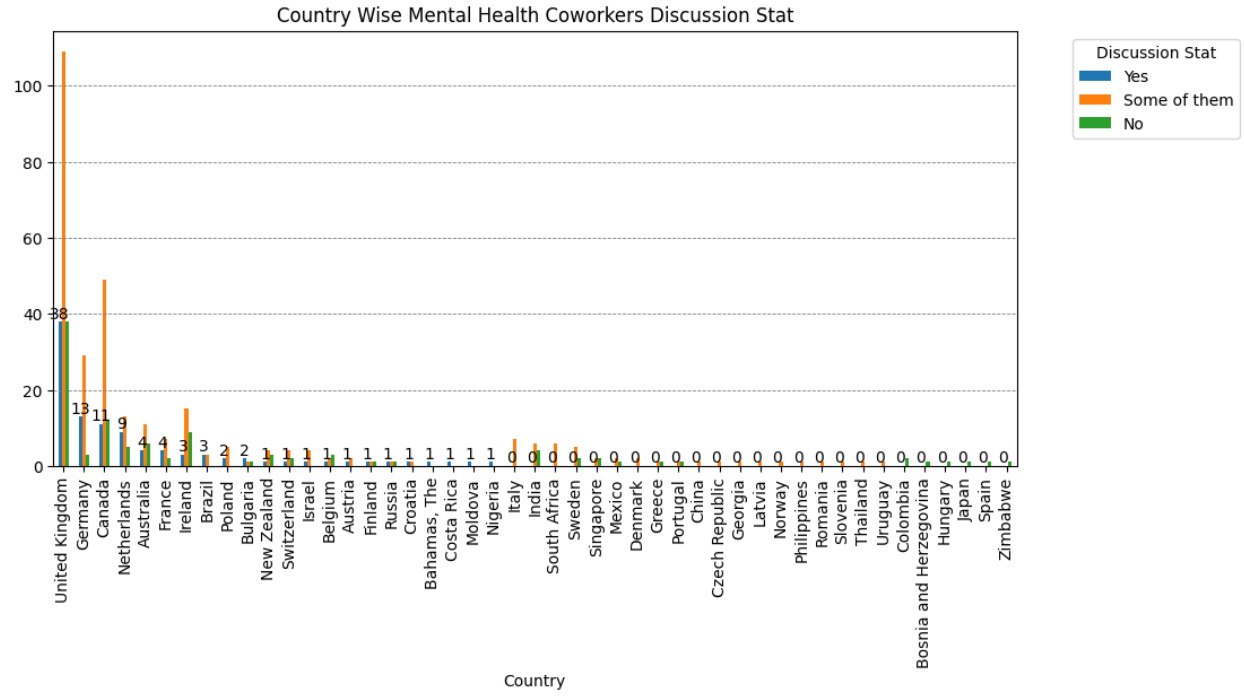



```
df_CoWrNnUSpVo= MH_pivote(df_CoWrNnUS,'coworkers','Country','count')

df_CoWrNnUSpVo.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 47 entries, 0 to 46
Data columns (total 4 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Country       47 non-null     object
 1   No            25 non-null     float64
 2   Some of them  37 non-null     float64
 3   Yes           22 non-null     float64
dtypes: float64(3), object(1)
memory usage: 1.6+ KB
```

```
df_CoWrNnUSpVo[['Yes','Some of them','No']] =
df_CoWrNnUSpVo[['Yes','Some of them','No']].fillna(value = 0)

df_CoWrNnUSpVo[['Yes','Some of them','No']] =
df_CoWrNnUSpVo[['Yes','Some of them','No']].astype('int64')
```

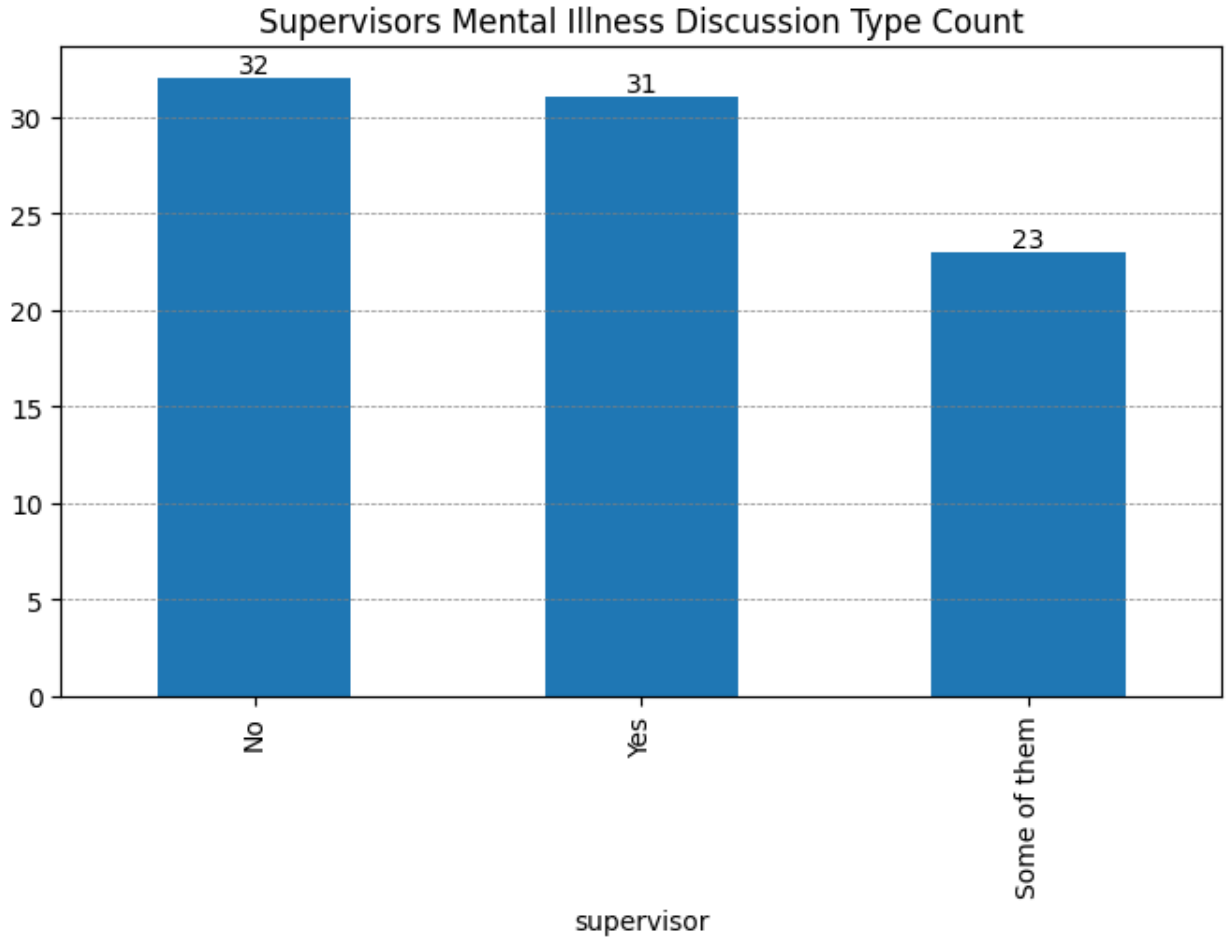
```
df_CoWrNnUSpVo= df_CoWrNnUSpVo.sort_values(by = ['Yes','Some of
them','No'], ascending = False)

df_CoWrNnUSVisPiv= bar_function(
    dfB = df_CoWrNnUSpVo,
    colsName1 = 'Country',
    colsName2 = ['Yes','Some of them','No'],
    graphKind = 'bar',
    wt = 11,
    ht = 5,
    grphTitle = 'Country Wise Mental Health Coworkers Discussion
Stat',
    legnTitle = 'Discussion Stat',
    grdAxix = 'y'
)
```




```
df_SupvNnUS=
df_NonUS[['Country','supervisor']].groupby(['Country','supervisor'],
observed = False)['supervisor'].count().reset_index(name = 'count')
df_SupvNnUS.head()
```

	Country	supervisor	count
0	Australia	No	7
1	Australia	Some of them	3
2	Australia	Yes	11
3	Austria	Some of them	2
4	Austria	Yes	1

```
df_SupvNnUS_1v=
df_SupvNnUS['supervisor'].value_counts().plot.bar(figsize = (8,4.5))
plt.title('Supervisors Mental Illness Discussion Type Count')
plt.grid(visible= True, axis = 'y', color ='gray', linestyle ='--',
linewidth = 0.5)

for cols in df_SupvNnUS_1v.containers:
    df_SupvNnUS_1v.bar_label(cols, label_type= 'edge')
```

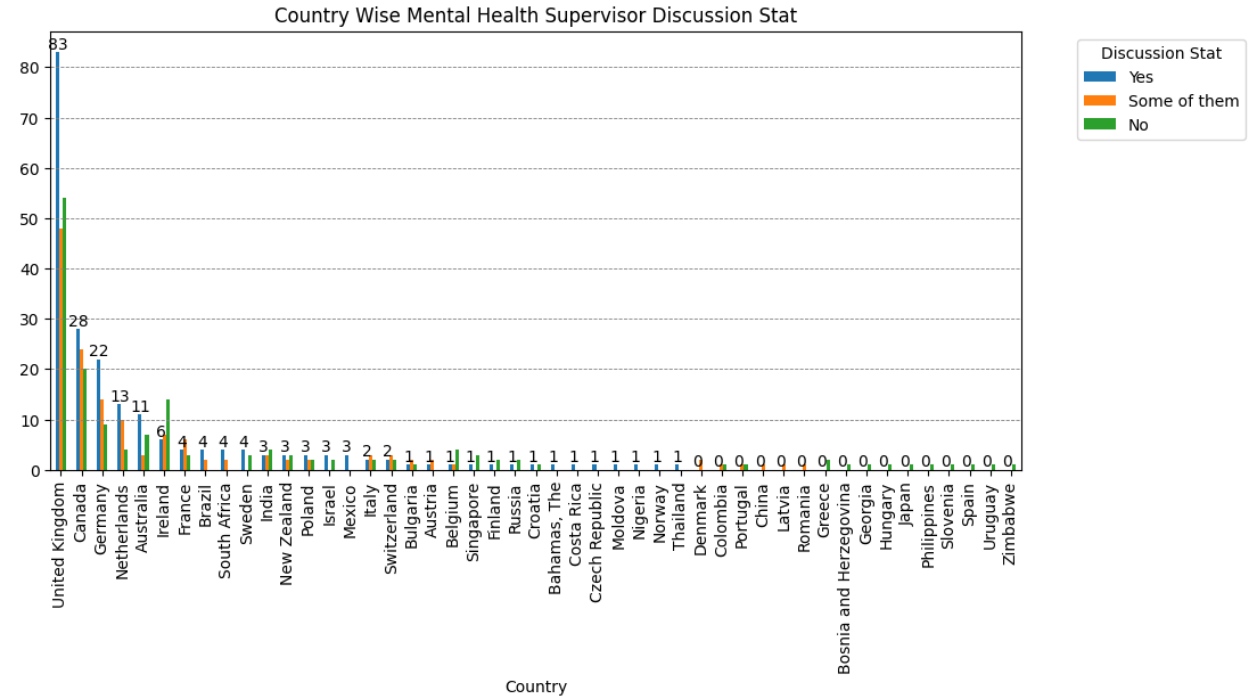



```
df_SupvNnUSpiV = MH_pivote(df_SupvNnUS,'supervisor','Country','count')

df_SupvNnUSpiV.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 47 entries, 0 to 46
Data columns (total 4 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Country       47 non-null     object
 1   No            32 non-null     float64
```

```
 2   Some of them  23 non-null     float64
 3   Yes           31 non-null     float64
dtypes: float64(3), object(1)
memory usage: 1.6+ KB
```

```
df_SupvNnUSpiV[['Yes','Some of them','No']] =
df_SupvNnUSpiV[['Yes','Some of them','No']].fillna(value = 0)

df_SupvNnUSpiV[['Yes','Some of them','No']] =
df_SupvNnUSpiV[['Yes','Some of them','No']].astype('int64')

df_SupvNnUSpiV= df_SupvNnUSpiV.sort_values(by = ['Yes','Some of
them','No'], ascending = False)

df_SupvNnUSpiV_2v =bar_function(
    dfB = df_SupvNnUSpiV,
    colsName1 = 'Country',
    colsName2 = ['Yes','Some of them','No'],
    graphKind = 'bar',
    wt = 11,
    ht = 5,
    grphTitle = 'Country Wise Mental Health Supervisor Discussion
Stat',
    legnTitle = 'Discussion Stat',
    grdAxix = 'y'
)
```



```
## Measured by benefits, seek_help, anonymity:
### Do companies support mental health?

# "df_BenCuNnUS" variable holds the country wise employer  "benefits"
stat
df_BenCuNnUS= df_NonUS[['Country', 'benefits']].groupby(['Country',
'benefits'], observed = False)['benefits'].count().reset_index(name =
'count')
df_BenCuNnUS.head()
```

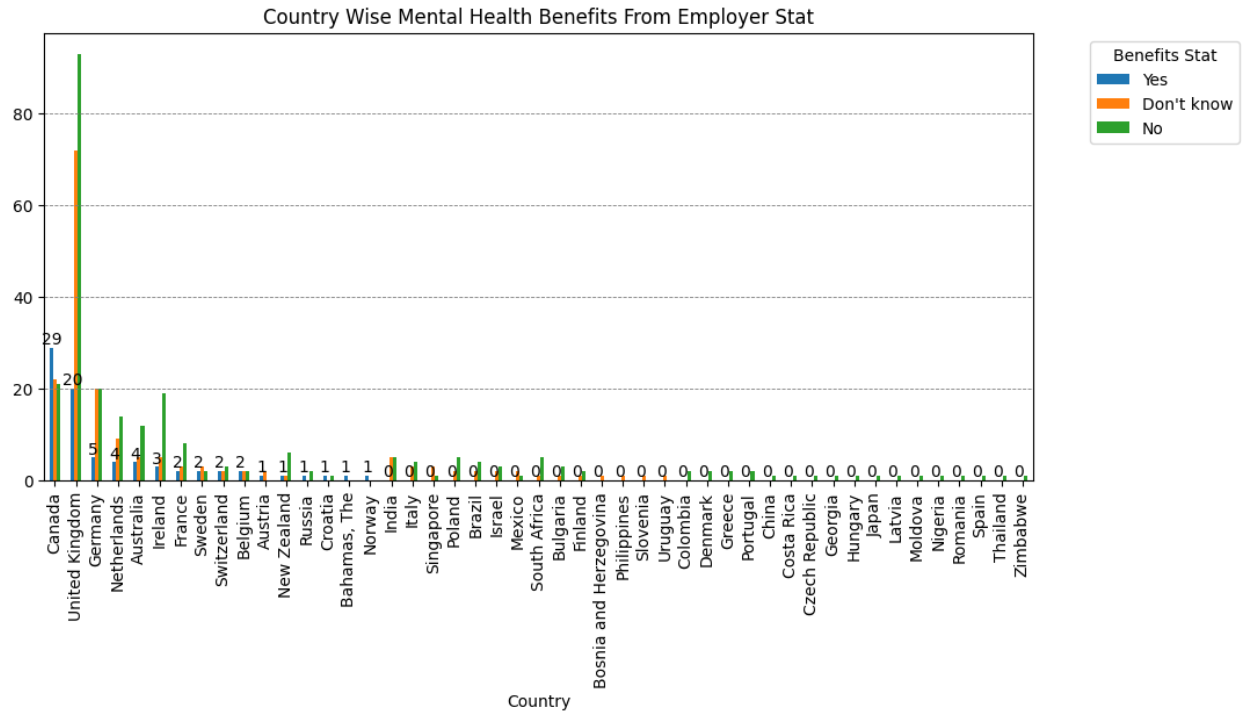
```
     Country    benefits  count
0  Australia  Don't know      5
1  Australia          No     12
2  Australia         Yes      4
3    Austria  Don't know      2
4    Austria         Yes      1
```

```
df_BenCuNnUSpiV= MH_pivote(df_BenCuNnUS,'benefits','Country','count')

df_BenCuNnUSpiV.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 47 entries, 0 to 46
Data columns (total 4 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   Country     47 non-null     object
 1   Don't know  26 non-null     float64
 2   No          40 non-null     float64
 3   Yes         16 non-null     float64
dtypes: float64(3), object(1)
memory usage: 1.6+ KB
```

```
df_BenCuNnUSpiV[['Yes','Don\'t know','No']] =
df_BenCuNnUSpiV[['Yes','Don\'t know','No']].fillna(value = 0)

df_BenCuNnUSpiV[['Yes','Don\'t know','No']] =
df_BenCuNnUSpiV[['Yes','Don\'t know','No']].astype('int64')

df_BenCuNnUSpiV= df_BenCuNnUSpiV.sort_values(by = ['Yes','Don\'t
know','No'], ascending = False)

df_BenCuNnUSpViS= bar_function(
    dfB = df_BenCuNnUSpiV,
    colsName1 = 'Country',
    colsName2 = ['Yes','Don\'t know','No'],
    graphKind = 'bar',
    wt = 11,
    ht = 5,
    grphTitle = 'Country Wise Mental Health Benefits From Employer
Stat',
```

```
    legnTitle = 'Benefits Stat',
    grdAxix = 'y'
)
```



Conclusion:

From the above two graph the it clearly visible thet -

In Canada most of the employer provides benefits to the mental patients

In UK most of the employer doesn't provides benefits to the mental patients

```
## Measured by seek_help

# "df_BenCuNnUS" variable holds the country wise employer  "benefits" stat
df_SekCuNnUS= df_NonUS[['Country', 'seek_help']].groupby(['Country', 'seek_help'], observed = False)['seek_help'].count().reset_index(name = 'count')
df_SekCuNnUS.head()
```

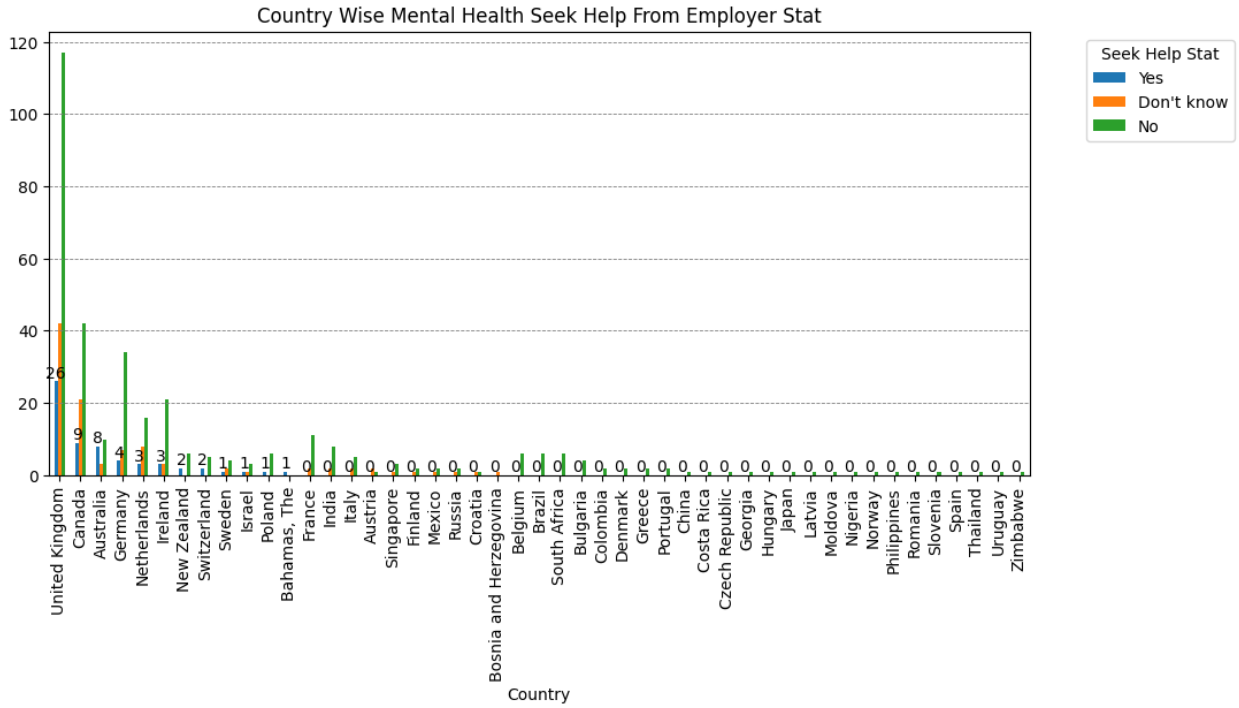
	Country	seek_help	count
0	Australia	Don't know	3
1	Australia	No	10
2	Australia	Yes	8
3	Austria	Don't know	2
4	Austria	No	1

```
df_SekCuNnUSpiV= MH_pivote(df_SekCuNnUS,'seek_help','Country','count')
```

```
df_SekCuNnUSpiV.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 47 entries, 0 to 46
Data columns (total 4 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   Country     47 non-null     object 
 1   Don't know  18 non-null     float64
 2   No          45 non-null     float64
 3   Yes         12 non-null     float64
dtypes: float64(3), object(1)
memory usage: 1.6+ KB
```

```
df_SekCuNnUSpiV[['Yes','Don\'t know','No']]=df_SekCuNnUSpiV[['Yes','Don\'t know','No']].fillna(value = 0)

df_SekCuNnUSpiV[['Yes','Don\'t know','No']]=df_SekCuNnUSpiV[['Yes','Don\'t know','No']].astype('int64')

df_SekCuNnUSpiV= df_SekCuNnUSpiV.sort_values(by= ['Yes','Don\'t know','No'], ascending = False)

df_SekCuNnUSVis = bar_function(
    dfB = df_SekCuNnUSpiV,
    colsName1 = 'Country',
    colsName2 = ['Yes','Don\'t know','No'],
    graphKind = 'bar',
    wt = 11,
    ht = 5,
    grphTitle = 'Country Wise Mental Health Seek Help From Employer Stat',
    legnTitle = 'Seek Help Stat',
    grdAxix = 'y',
)
```



Conclusion:

From the above two graph the it clearly visible thet -

In UK 24 (out 180 near 13.33% of the) employer provides helps to the mental patients to learn about the resources.

In UK most of the (out 180 near 65.55% of the) employer doesn't provides helps to the mental patients to learn about the resources

```
## Measured by anonymity:

# "df_BenCuNnUS" variable holds the country wise employer  "benefits" stat
df_CountAnoNnUS= df_NonUS[['Country', 'anonymity']].groupby(['Country', 'anonymity'], observed = False)['anonymity'].count().reset_index(name = 'count')
df_CountAnoNnUS.head()
```

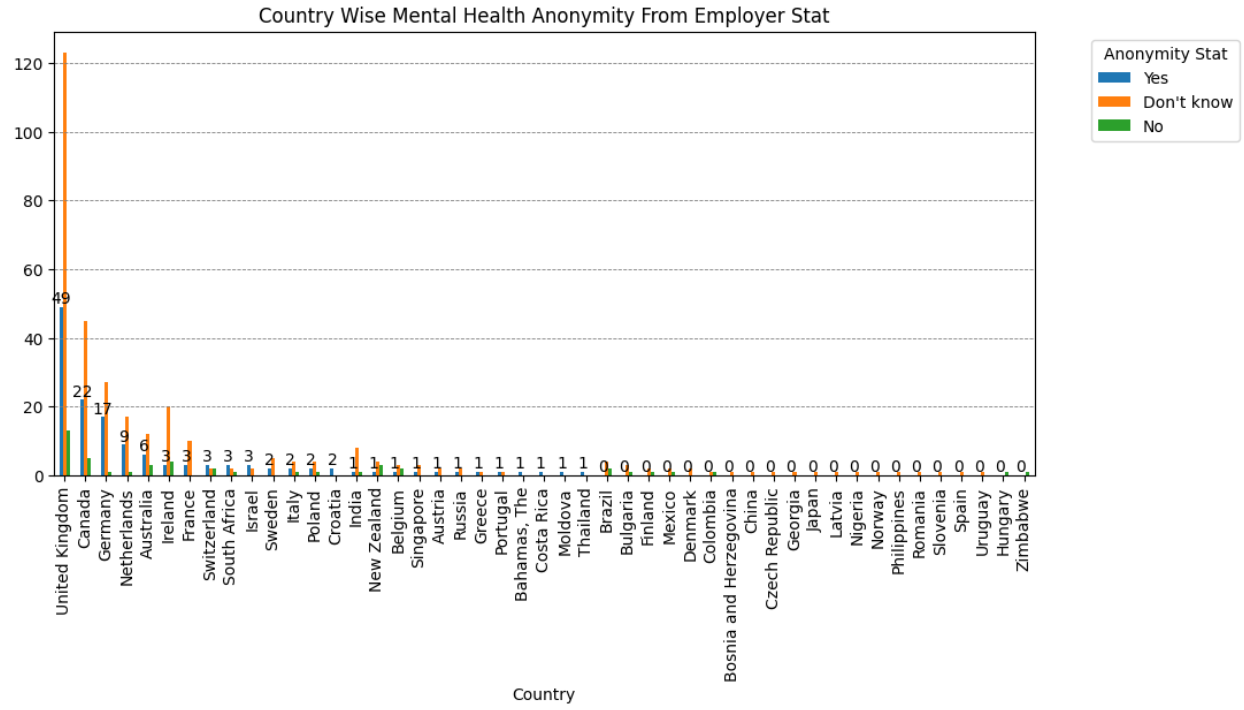
	Country	anonymity	count
0	Australia	Don't know	12
1	Australia	No	3
2	Australia	Yes	6
3	Austria	Don't know	2
4	Austria	Yes	1

```
df_CountAnoNnUSpiV= MH_pivote(df_CountAnoNnUS,'anonymity','Country','count')
```

```
df_CountAnoNnUSpiV.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 47 entries, 0 to 46
Data columns (total 4 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   Country     47 non-null     object
 1   Don't know  40 non-null     float64
 2   No          20 non-null     float64
 3   Yes         26 non-null     float64
dtypes: float64(3), object(1)
memory usage: 1.6+ KB
```

```
df_CountAnoNnUSpiV[['Yes','Don\'t
know','No']]=df_CountAnoNnUSpiV[['Yes','Don\'t
know','No']].fillna(value = 0)

df_CountAnoNnUSpiV[['Yes','Don\'t
know','No']]=df_CountAnoNnUSpiV[['Yes','Don\'t
know','No']].astype('int64')

df_CountAnoNnUSpiV= df_CountAnoNnUSpiV.sort_values(by= ['Yes','Don\'t
know','No'], ascending = False)

df_CountAnoNnUSVis = bar_function(
    dfB = df_CountAnoNnUSpiV,
    colsName1 = 'Country',
    colsName2 = ['Yes','Don\'t know','No'],
    graphKind = 'bar',
    wt = 11,
    ht = 5,
    grphTitle = 'Country Wise Mental Health Anonymity From Employer
Stat',
    legnTitle = 'Anonymity Stat',
    grdAxix = 'y',
)
```




```
df_CountAnoNnUSpiV.head()
```

anonymity	Country	Don't know	No	Yes
44	United Kingdom	123	13	49
7	Canada	45	5	22
17	Germany	27	1	17
28	Netherlands	17	1	9
0	Australia	12	3	6

Conclusion:

From the above two graph the it clearly visible thet -

In UK 49 (out 185 near 26.49% of the) employer protect anonymity of the mental patients

In UK most of the ((out 185 near 66.49% of the) employer whethere protect anonymity of the mental patients or not that data is not avilable.

So might be beople feared if they shared their mental health issue with their employer they might face several challenges in his/ her work places.

Problem Statement 2:

What are the strongest predictors of mental health illness or certain attitudes towards mental health in the workplace?"

Stongest predictor of mental health ***variables*** are -

those who seek ***"treatment"***

Happened ***"work_interfere"***

faced ***"mental_health_consequence"***

"obs_consequence": bserved negative consequences for coworkers with mental health conditions

Note:

obs_consequence: Have you heard of or observed negative consequences for coworkers with mental health conditions in your workplace?

FOR US:

```
#checking main working dataset
df_WS.head()
```

```
            Timestamp  Age  Gender         Country state self_employed
\
0 2014-08-27 11:29:31   37  Female   United States    IL           NaN

1 2014-08-27 11:29:37   44    Male   United States    IN           NaN

2 2014-08-27 11:29:44   32    Male          Canada   NaN           NaN

3 2014-08-27 11:29:46   31    Male  United Kingdom   NaN           NaN

4 2014-08-27 11:30:22   31    Male   United States    TX           NaN


  family_history treatment work_interfere    no_employees  ...  \
0             No       Yes          Often            6-25  ...
1             No        No         Rarely  More than 1000  ...
2             No        No         Rarely            6-25  ...
3            Yes       Yes          Often          26-100  ...
4             No        No          Never         100-500  ...

  mental_health_consequence phys_health_consequence     coworkers
supervisor  \
0                        No                       No  Some of them
Yes
1                     Maybe                       No            No
No
2                        No                       No           Yes
Yes
3                        Yes                     Yes  Some of them
No
4                        No                       No  Some of them
Yes
```

```
  mental_health_interview phys_health_interview mental_vs_physical  \
0                      No                  Maybe                 Yes
1                      No                     No          Don't know
2                     Yes                    Yes                  No
3                   Maybe                  Maybe                  No
4                     Yes                    Yes          Don't know

  obs_consequence comments Timestamp_MEfeq
0              No      NaN     August 2014
1              No      NaN     August 2014
2              No      NaN     August 2014
3             Yes      NaN     August 2014
4              No      NaN     August 2014

[5 rows x 28 columns]
```

```
df_wsfQ2 = df_WS[df_WS['Country']== 'United States']
[['state','family_history','treatment','work_interfere',

'mental_health_consequence', 'obs_consequence']].copy()
df_wsfQ2.head()
```

```
  state family_history treatment work_interfere
mental_health_consequence  \
0    IL              No       Yes          Often
No
1    IN              No        No         Rarely
Maybe
4    TX              No        No          Never
No
5    TN             Yes        No      Sometimes
No
6    MI             Yes       Yes      Sometimes
Maybe

  obs_consequence
0              No
1              No
4              No
5              No
6              No
```

```
# Removing Duplicate values not a right decession, because many people
might faced or have same types of situation opinion
df_wsfQ2.duplicated().sum()
```

```
np.int64(272)
```

```
##For Family History Yes

df_wsfQ2_1= df_wsfQ2[(df_wsfQ2 ['family_history']== 'Yes') & (df_wsfQ2
```

```
['treatment']== 'Yes') & (df_wsfQ2 ['work_interfere'].isin(['Often',

'Sometimes',

'Rarely']))& (df_wsfQ2 ['mental_health_consequence'].isin(['Yes',

'Maybe']))]

df_wsfQ2_1.head()
```

```
   state family_history treatment work_interfere
mental_health_consequence  \
6     MI            Yes       Yes      Sometimes
Maybe
8     IL            Yes       Yes      Sometimes
Maybe
12    CA            Yes       Yes      Sometimes
Yes
20    NY            Yes       Yes      Sometimes
Maybe
25    TN            Yes       Yes      Sometimes
Yes

   obs_consequence
6               No
8               No
12             Yes
20              No
25              No
```

```
df_wsfQ2_1V= df_wsfQ2_1['obs_consequence'].value_counts()
print (f'Here is The obs_consequence Stat {df_wsfQ2_1V}')
```

```
Here is The obs_consequence Stat obs_consequence
No     141
Yes     34
Name: count, dtype: int64
```

Conclusion:

With family history Yes, 34 number of people have heard of or observed negative consequences for coworkers with mental health conditions

```
##For Family History No

df_wsfQ2_2= df_wsfQ2[(df_wsfQ2 ['family_history']== 'No')& (df_wsfQ2
['treatment']== 'Yes') & (df_wsfQ2 ['work_interfere'].isin(['Often',

'Sometimes',
```

```
'Rarely']))& (df_wsfQ2 ['mental_health_consequence'].isin(['Yes',

'Maybe']))]

df_wsfQ2_2.head()
```

```
   state family_history treatment work_interfere
mental_health_consequence  \
17    TN              No       Yes      Sometimes
Maybe
22    MA              No       Yes          Often
Maybe
34    WI              No       Yes      Sometimes
Maybe
83    NY              No       Yes          Often
Maybe
88    FL              No       Yes      Sometimes
Maybe

   obs_consequence
17              No
22              No
34              No
83              No
88             Yes
```

```
df_wsfQ2_2V= df_wsfQ2_2['obs_consequence'].value_counts()
print (f'Here is The obs_consequence Stat (Family History No)
{df_wsfQ2_2V}')
```

```
Here is The obs_consequence Stat (Family History No) obs_consequence
No     75
Yes    18
Name: count, dtype: int64
```

Conclusion:

With family history No, 18 number of people have heard of or observed negative consequences for coworkers with mental health conditions

#Finding The stat for "obs_consequence"= Yes

```
df_wsfObsCon = df_wsfQ2[['state',
          'obs_consequence']].groupby(['state', 'obs_consequence'],
                                      observed = False)
['obs_consequence'].count().reset_index(name = 'count')

df_wsfObsCon.head()
```

```
  state obs_consequence  count
0    AL              No      6
```

```
1    AL              Yes      2
2    AZ               No      7
3    CA               No    122
4    CA              Yes     16
```

```
df_wsfObsConPiv=
MH_pivote(df_wsfObsCon,'obs_consequence','state','count')
```

```
df_wsfObsConPiv= df_wsfObsConPiv.sort_values(by= ['Yes','No'],
ascending = False)
```

```
df_wsfObsConPiv.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 46 entries, 2 to 44
Data columns (total 3 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   state   46 non-null     object
 1   No      44 non-null     float64
 2   Yes     28 non-null     float64
dtypes: float64(2), object(1)
memory usage: 1.4+ KB
```

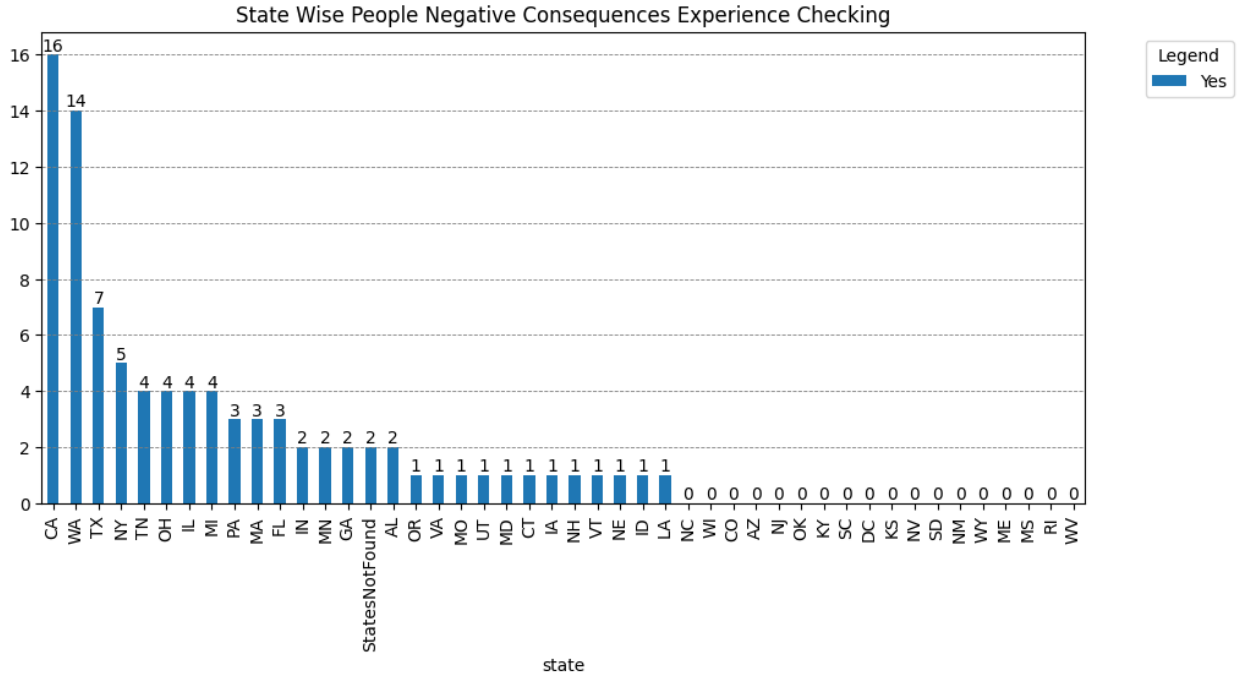
```
df_wsfObsConPiv[['Yes','No']]=
df_wsfObsConPiv[['Yes','No']].fillna(value = 0)
```

```
df_wsfObsConPiv[['Yes','No']]=
df_wsfObsConPiv[['Yes','No']].astype('int64')
```

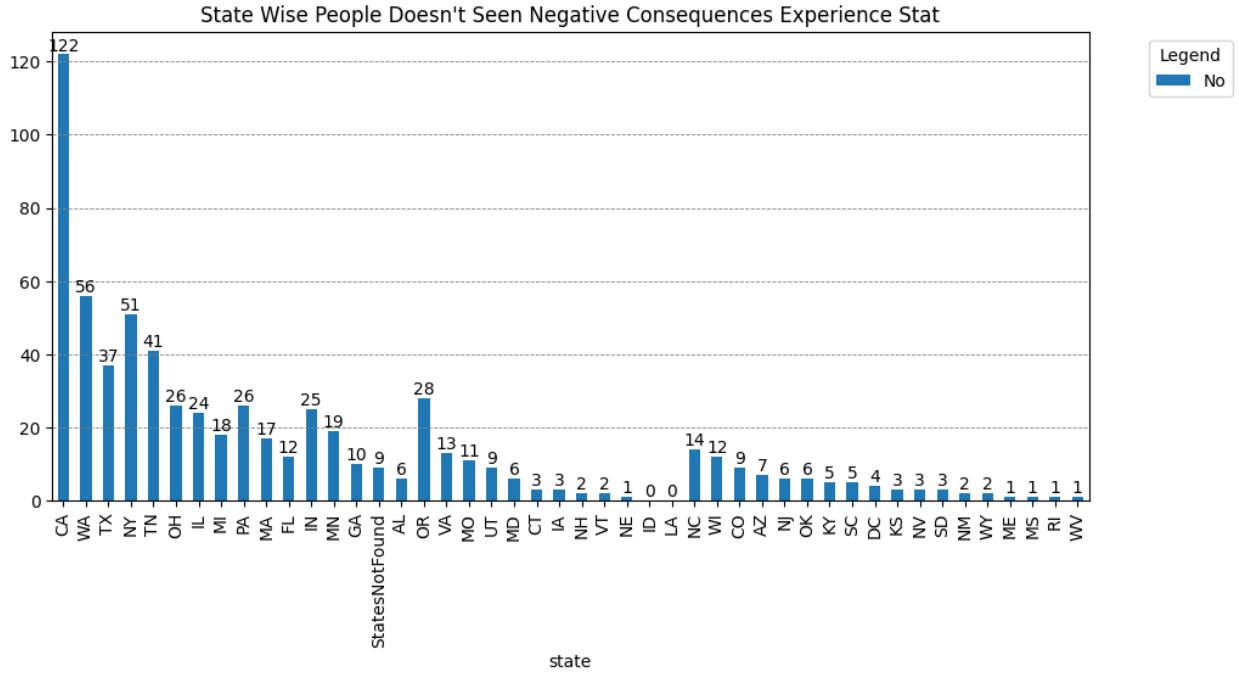
```
df_wsfObsConPiv.head()
```

```
obs_consequence state   No  Yes
2                 CA  122   16
42                WA   56   14
38                TX   37    7
28                NY   51    5
37                TN   41    4
```

```
df_wsfObsConVis= bar_function(
    dfB = df_wsfObsConPiv,
    colsName1 = 'state',
    colsName2 = 'Yes',
    graphKind = 'bar',
    wt = 11,
    ht = 5,
    grphTitle = 'State Wise People Negative Consequences Experience
Checking',
    legnTitle = 'Legend',
    grdAxix = 'y'
)
```

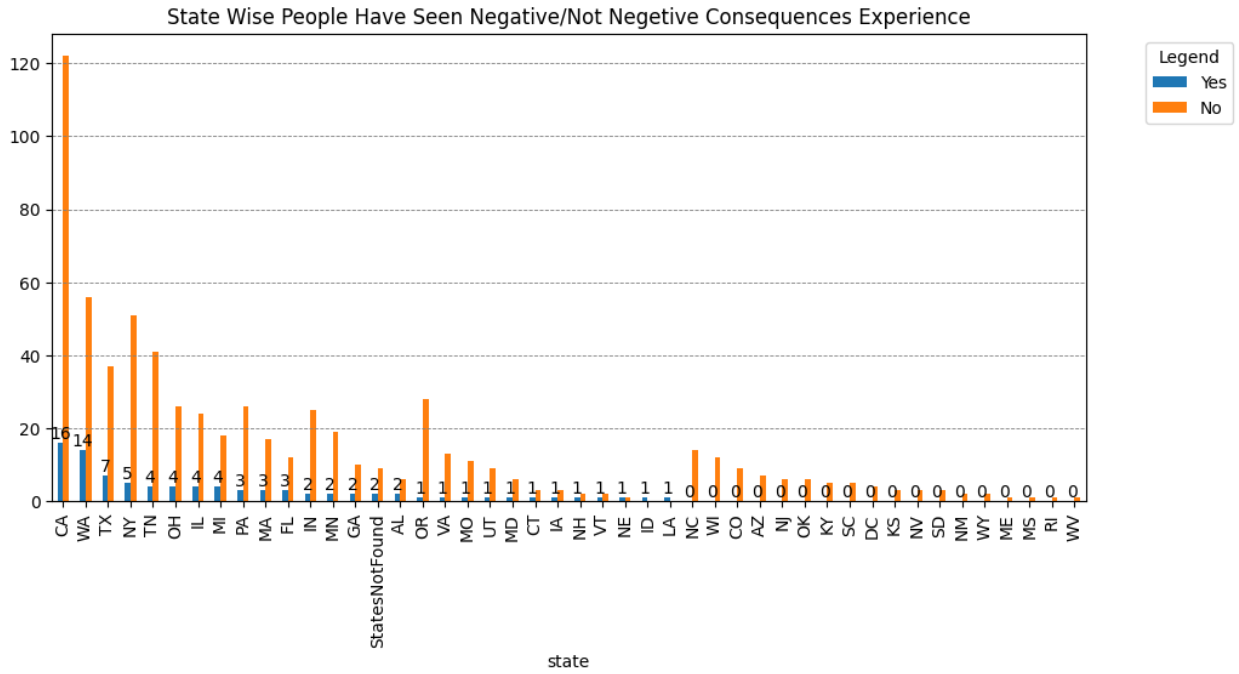



```
df_wsfObsConVis2= bar_function(
    dfB = df_wsfObsConPiv,
    colsName1 = 'state',
    colsName2 = 'No',
    graphKind = 'bar',
    wt = 11,
    ht = 5,
    grphTitle = 'State Wise People Doesn\'t Seen Negative Consequences
Experience Stat',
    legnTitle = 'Legend',
    grdAxix = 'y'
)
```



COMPARISION:

```
df_wsfObsConVis3= bar_function(
    dfB = df_wsfObsConPiv,
    colsName1 = 'state',
    colsName2 = ['Yes','No'],
    graphKind = 'bar',
    wt = 11,
    ht = 5,
    grphTitle = 'State Wise People Have Seen Negative/Not Negetive
Consequences Experience',
    legnTitle = 'Legend',
    grdAxix = 'y'
)
```



Conclusion:

Canada is such a state of the United States where among the employee - -

- **Highest number of people (16)** reported **negative consequences experience** checking in their work places
- **Highest number of people (122)** reported **no negative consequences experience** checking in their work places

Finding in what type of organisation people faced most meantal heath consiquences:

```
#for "family_history" = Yes
df_wsYesMayCount = df_WS[(df_WS['Country']== 'United States') &
(df_WS['family_history']== 'Yes') &
(df_WS['mental_health_consequence'].isin(['Yes','Maybe']))][['state' ,
'no_employees','mental_health_consequence']]
df_wsYesMayCount.head()
```

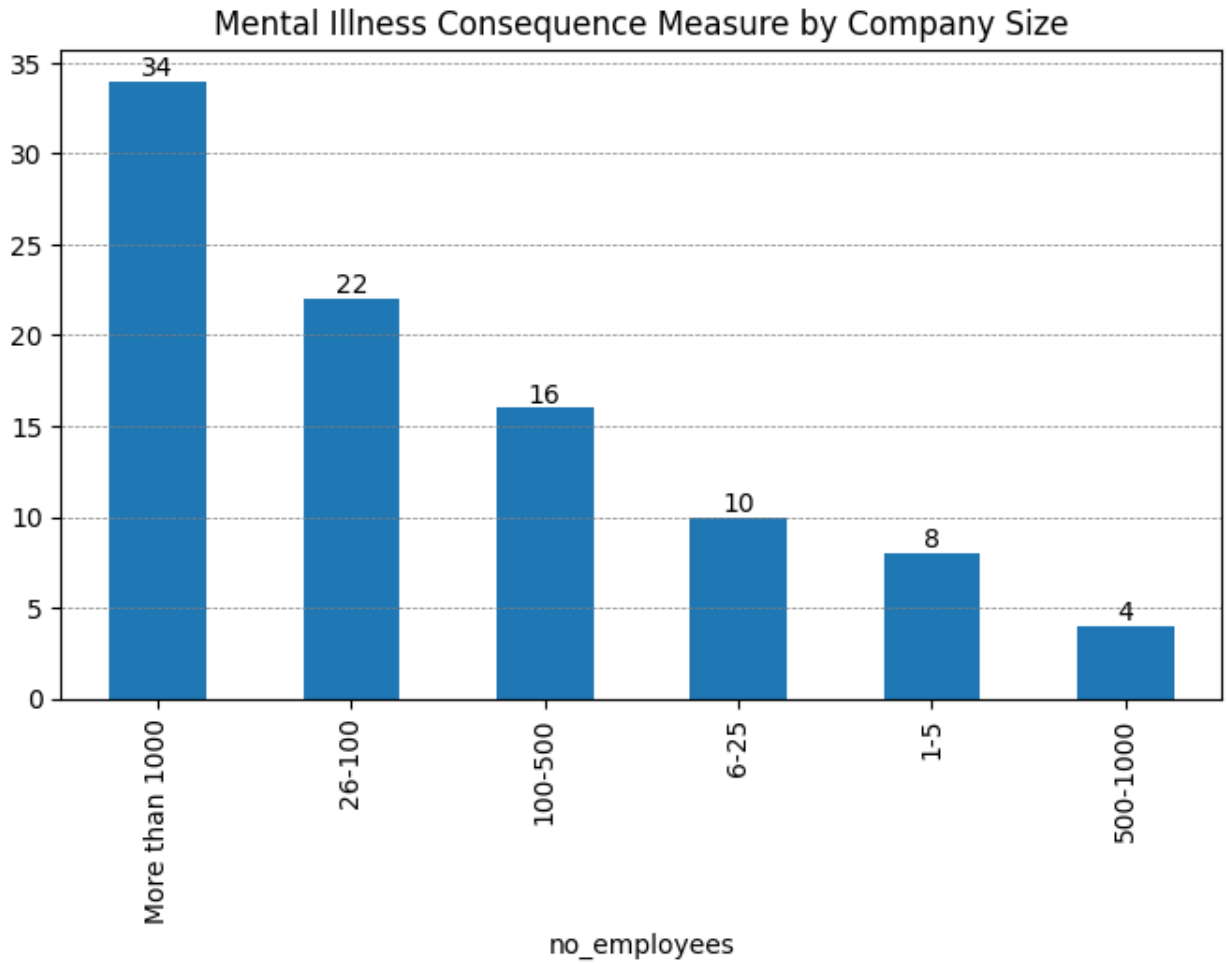
```
   state    no_employees mental_health_consequence
6     MI             1-5                     Maybe
8     IL         100-500                     Maybe
12    CA          26-100                       Yes
20    NY         100-500                     Maybe
25    TN  More than 1000                       Yes
```

```
#Finding "Yes" Stat:
df_wsYesMayCount_1=
df_wsYesMayCount[df_wsYesMayCount['mental_health_consequence']==
```

```
'Yes']['no_employees'].value_counts()
df_wsYesMayCount_1
```

```
no_employees
More than 1000    34
26-100            22
100-500           16
6-25              10
1-5                8
500-1000           4
Name: count, dtype: int64
```

```
df_wsYesMayCount_1V= df_wsYesMayCount_1.plot.bar(figsize = (8,4.5))
plt.title('Mental Illness Consequence Measure by Company Size')
plt.grid(visible= True, axis = 'y', color ='gray', linestyle ='--',
linewidth = 0.5)

for cols in df_wsYesMayCount_1V.containers:
    df_wsYesMayCount_1V.bar_label(cols, label_type= 'edge')
```



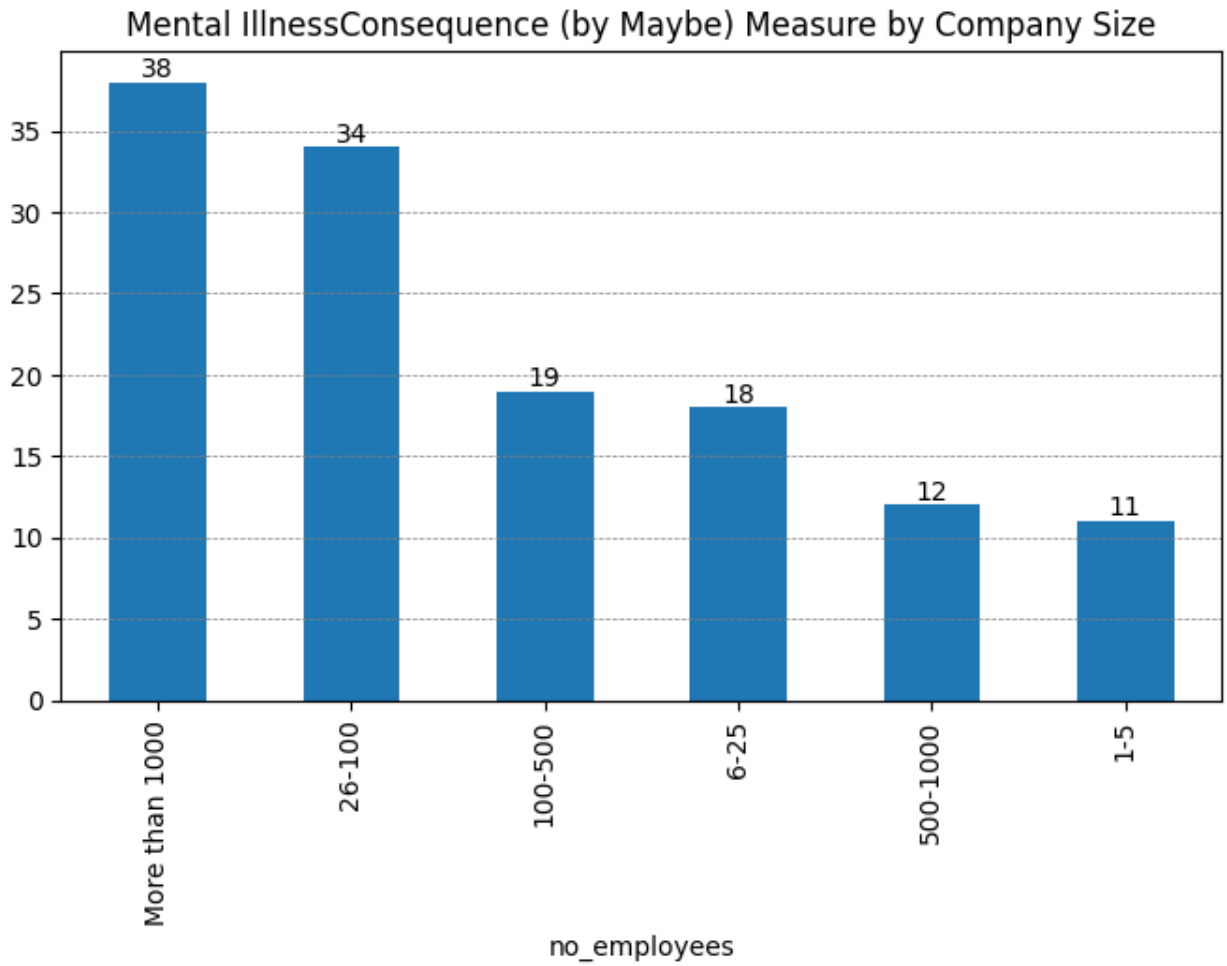
Conclusion:

Most of the people think in big firms yes they feel they will face or they might have seen mental health consiquences

```
#Finding "Maybe" Stat:
df_wsYesMayCount_2=
df_wsYesMayCount[df_wsYesMayCount['mental_health_consequence']==
'Maybe']['no_employees'].value_counts()
df_wsYesMayCount_2
```

```
no_employees
More than 1000    38
26-100            34
100-500           19
6-25              18
500-1000          12
1-5               11
Name: count, dtype: int64
```

```
df_wsYesMayCount_2V= df_wsYesMayCount_2.plot.bar(figsize = (8,4.5))
plt.title('Mental IllnessConsequence (by Maybe) Measure by Company
Size')
plt.grid(visible= True, axis = 'y', color ='gray', linestyle ='--',
linewidth = 0.5)

for cols in df_wsYesMayCount_2V.containers:
    df_wsYesMayCount_2V.bar_label(cols, label_type= 'edge')
```



Conclusion:

Most of the people think in big firms followed by small-medium (26-100 employees) yes they feel they will face mental health consiquences.

```
##for "family_history" = No

df_NoMayCount= df_WS[(df_WS['Country']== 'United States') &
(df_WS['family_history']== 'No') &
(df_WS['mental_health_consequence'].isin(['Yes',

'Maybe']))][['state',

'no_employees',

'mental_health_consequence']].copy()
df_NoMayCount.head()
```

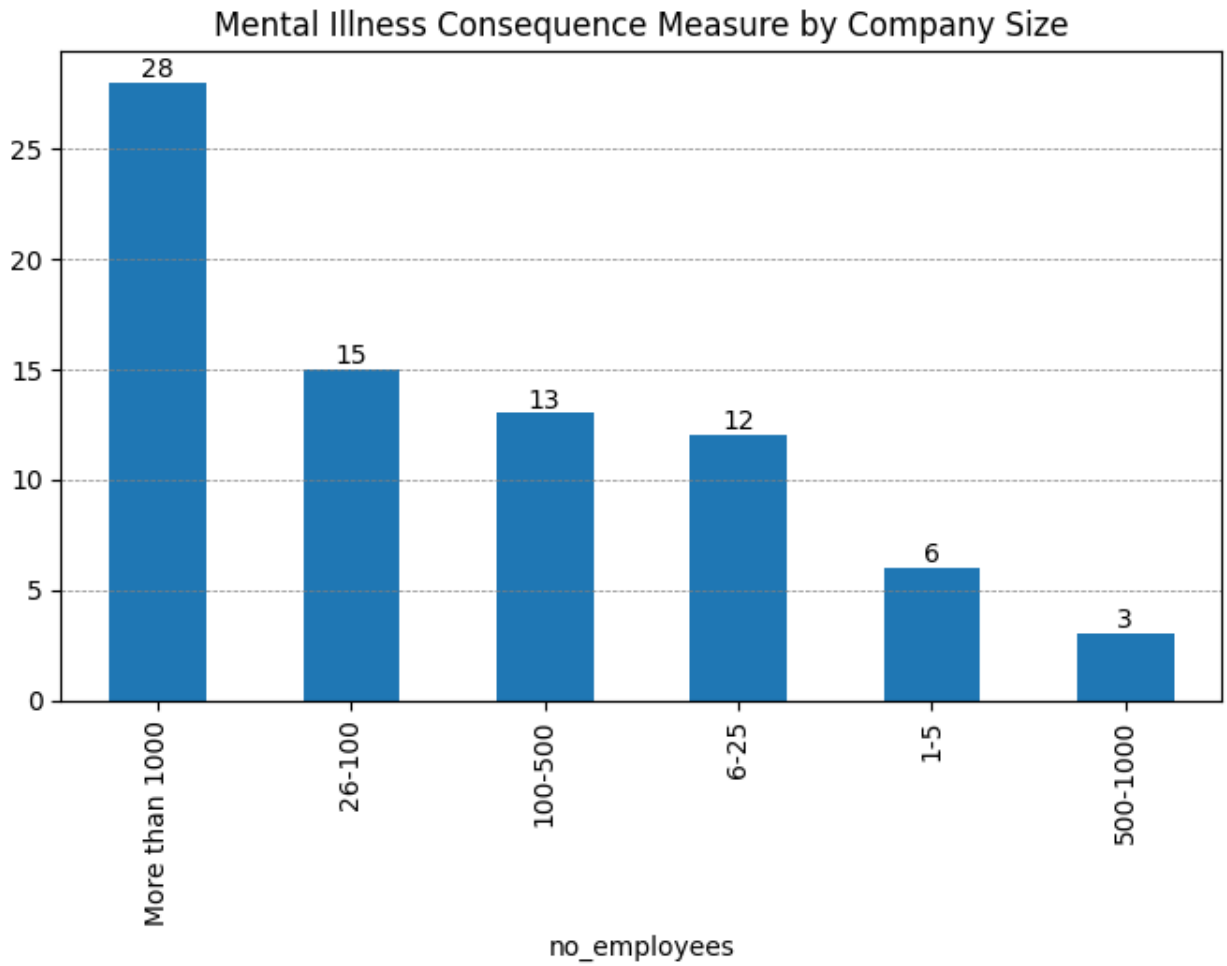
```
   state    no_employees mental_health_consequence
1     IN  More than 1000                      Maybe
```

```
17    TN            6-25                        Maybe
22    MA          26-100                        Maybe
23    IA  More than 1000                        Maybe
26    TN             1-5                        Maybe
```

```
#Finding "Yes" Stat:
df_NoMayCount_1=
df_NoMayCount[df_NoMayCount['mental_health_consequence']== 'Yes']
['no_employees'].value_counts()
df_NoMayCount_1
```

```
no_employees
More than 1000    28
26-100            15
100-500           13
6-25              12
1-5                6
500-1000           3
Name: count, dtype: int64
```

```
##for "family_history" = No
df_NoMayCount_1V= df_NoMayCount_1.plot.bar(figsize = (8,4.5))
plt.title('Mental Illness Consequence Measure by Company Size')
plt.grid(visible= True, axis = 'y', color ='gray', linestyle ='--',
linewidth = 0.5)

for cols in df_NoMayCount_1V.containers:
    df_NoMayCount_1V.bar_label(cols, label_type= 'edge')
```



Conclusion:

Among "family_history" = No

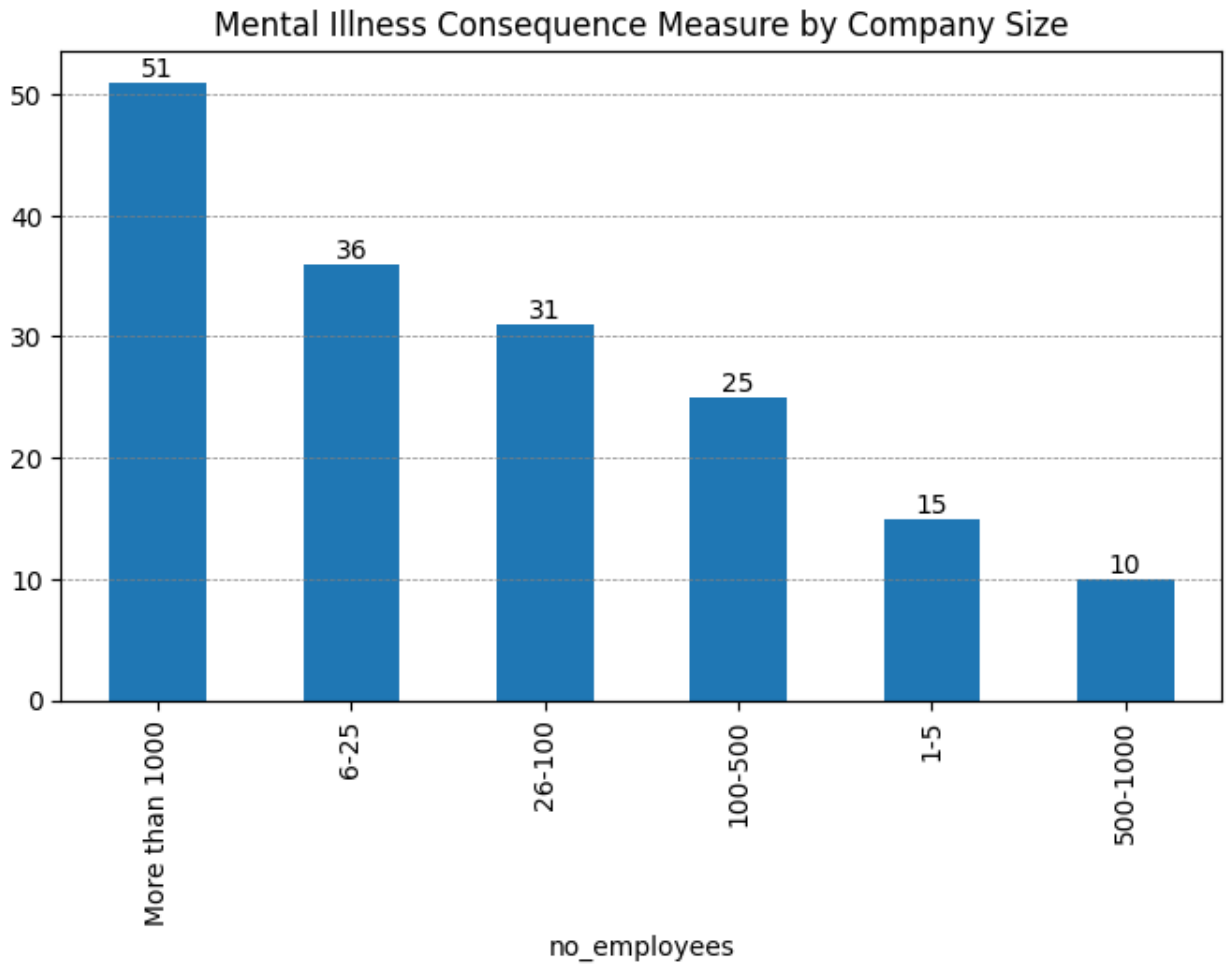
-
  - 35% Big working people think yes they feel they will face or they might have seen mental health consiquences in their organisation

```
#Finding "Maybe" Stat:
df_NoMayCount_2=
df_NoMayCount[df_NoMayCount['mental_health_consequence']== 'Maybe']
['no_employees'].value_counts()
df_NoMayCount_2
```

```
no_employees
More than 1000    51
6-25              36
26-100            31
100-500           25
1-5               15
```

```
500-1000           10
Name: count, dtype: int64
```

```
##for "family_history" = No
df_NoMayCount_2V= df_NoMayCount_2.plot.bar(figsize = (8,4.5))
plt.title('Mental Illness Consequence Measure by Company Size')
plt.grid(visible= True, axis = 'y', color ='gray', linestyle ='--',
linewidth = 0.5)

for cols in df_NoMayCount_2V.containers:
    df_NoMayCount_2V.bar_label(cols, label_type= 'edge')
```



Conclusion:

Among "family_history" = No

-
  - 64.56% Big working people think **Maybe** they feel they will face or they might have seen mental health consiquences in their organisation

Finding in which states what type of organisation people--

--faced most meantal heath consiquences: Irespective of their family background

```
## "df_UsYsStCount" holds state wise data for =>
'mental_health_consequence' = ['Yes','Maybe']
df_UsYsStCount = df_WS[(df_WS['Country']== 'United States') &
(df_WS['mental_health_consequence'].isin(['Yes','Maybe']))]
[['state','no_employees','mental_health_consequence']]
df_UsYsStCount.head()
```

```
   state    no_employees mental_health_consequence
1     IN  More than 1000                     Maybe
6     MI             1-5                     Maybe
8     IL         100-500                     Maybe
12    CA          26-100                       Yes
17    TN            6-25                     Maybe
```

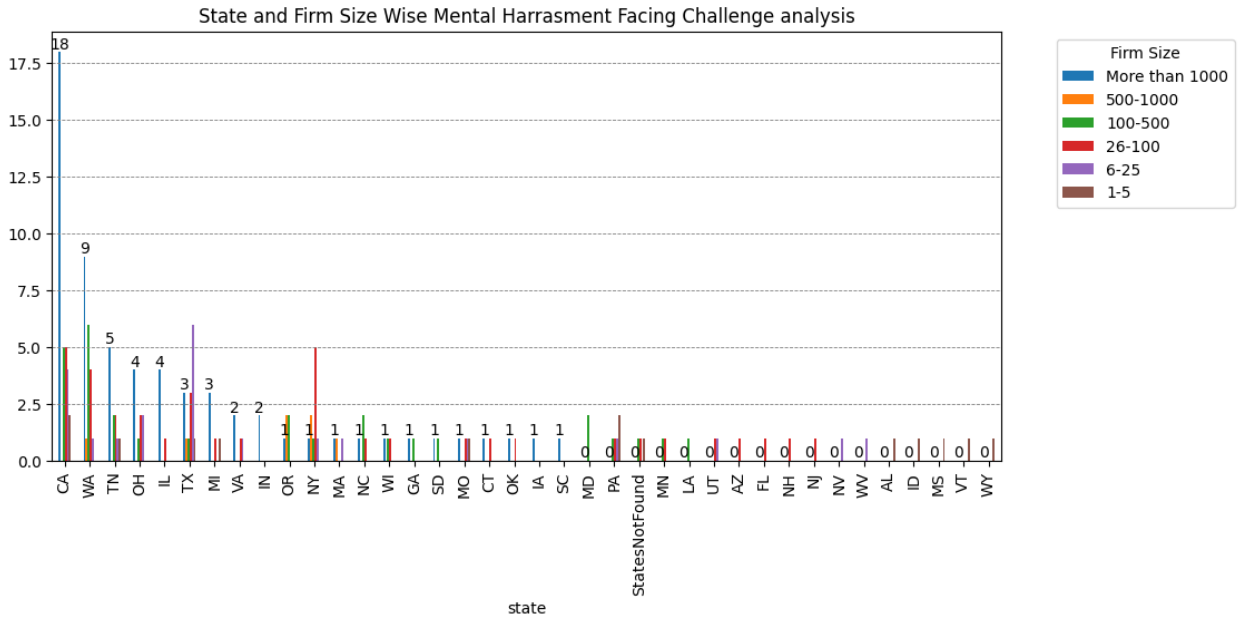
```
#Yes data
df_UsYsStCountpiV=df_UsYsStCount[df_UsYsStCount['mental_health_consequ
ence']== 'Yes'].groupby(['state','no_employees'],

observed = False)['no_employees'].count().reset_index(name = 'Count')
df_UsYsStCountpiV.head()
```

```
  state no_employees  Count
0    AL           1-5      1
1    AZ        26-100      1
2    CA           1-5      2
3    CA       100-500      5
4    CA        26-100      5
```

```
df_UsYsStCountpiV
=MH_pivote(df_UsYsStCountpiV,'no_employees','state','Count')

df_UsYsStCountpiV.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 38 entries, 0 to 37
Data columns (total 7 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   state           38 non-null     object
 1   1-5             12 non-null     float64
 2   100-500         16 non-null     float64
 3   26-100          22 non-null     float64
 4   500-1000        5 non-null      float64
 5   6-25            13 non-null     float64
 6   More than 1000  21 non-null     float64
dtypes: float64(6), object(1)
memory usage: 2.2+ KB
```

```
df_UsYsStCountpiV[['More than 1000','500-1000','100-500','26-100','6-
25','1-5']]= df_UsYsStCountpiV[['More than 1000','500-1000','100-
500','26-100','6-25','1-5']].fillna(value = 0)

df_UsYsStCountpiV[['More than 1000','500-1000','100-500','26-100','6-
25','1-5']]=df_UsYsStCountpiV[['More than 1000','500-1000','100-
500','26-100','6-25','1-5']].astype('int64')

df_UsYsStCountpiV= df_UsYsStCountpiV.sort_values(by= ['More than
1000','500-1000','100-500','26-100','6-25','1-5'], ascending = False)

df_UsYsStCountVis = bar_function(
    dfB = df_UsYsStCountpiV,
    colsName1 = 'state',
    colsName2 = ['More than 1000','500-1000','100-500','26-100','6-
25','1-5'],
    graphKind = 'bar',
    wt = 11,
    ht = 5,
    grphTitle = 'State and Firm Size Wise Mental Harrasment Facing
Challenge analysis',
    legnTitle = 'Firm Size',
    grdAxix = 'y')
```



Conclusion:

In CA maximum percentage of big firm working people think they will or they might have seen mental harrasment facing challenge

```
#Yes data
df_UsMaybeStCntpiV=df_UsYsStCount[df_UsYsStCount['mental_health_conseq
```

```
uence']== 'Maybe'].groupby(['state','no_employees'],

observed = False)['no_employees'].count().reset_index(name = 'Count')
df_UsMaybeStCntpiV.head()
```

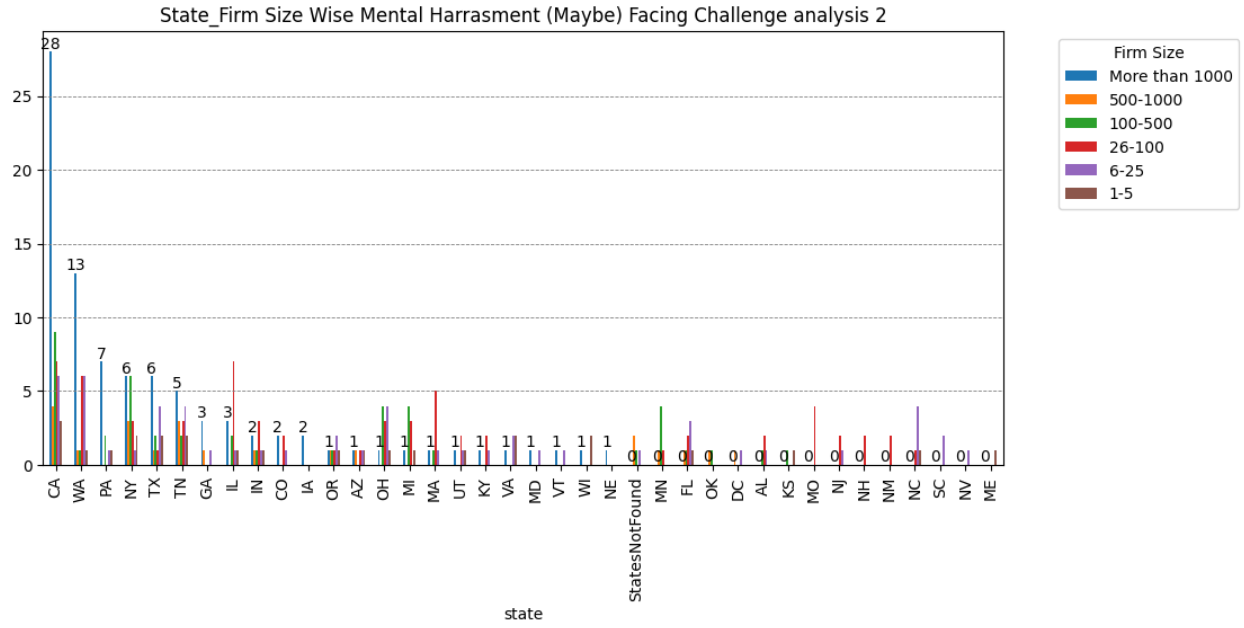
```
  state no_employees  Count
0    AL      100-500      1
1    AL       26-100      2
2    AL         6-25      1
3    AZ          1-5      1
4    AZ       26-100      1
```

```
df_UsMaybeStCntpiV
=MH_pivote(df_UsMaybeStCntpiV,'no_employees','state','Count')

df_UsMaybeStCntpiV.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 38 entries, 0 to 37
Data columns (total 7 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   state           38 non-null     object
 1   1-5             19 non-null     float64
 2   100-500         18 non-null     float64
 3   26-100          23 non-null     float64
 4   500-1000        14 non-null     float64
 5   6-25            27 non-null     float64
 6   More than 1000  23 non-null     float64
dtypes: float64(6), object(1)
memory usage: 2.2+ KB
```

```
df_UsMaybeStCntpiV[['More than 1000','500-1000','100-500','26-100','6-
25','1-5']]= df_UsMaybeStCntpiV[['More than 1000','500-1000','100-
500','26-100','6-25','1-5']].fillna(value = 0)

df_UsMaybeStCntpiV[['More than 1000','500-1000','100-500','26-100','6-
25','1-5']]=df_UsMaybeStCntpiV[['More than 1000','500-1000','100-
500','26-100','6-25','1-5']].astype('int64')

df_UsMaybeStCntpiV= df_UsMaybeStCntpiV.sort_values(by= ['More than
1000','500-1000','100-500','26-100','6-25','1-5'], ascending = False)

## Stat for 'mental_health_consequence' == 'Maybe'
df_UsMaybeStCnt_vs1 =bar_function(
    dfB = df_UsMaybeStCntpiV,
    colsName1 = 'state',
    colsName2 = ['More than 1000','500-1000','100-500','26-100','6-
25','1-5'],
    graphKind = 'bar',
    wt = 11,
```

```
    ht = 5,
    grphTitle = 'State_Firm Size Wise Mental Harrasment (Maybe) Facing
Challenge analysis 2',
    legnTitle = 'Firm Size',
    grdAxix = 'y')
```



Conclusion:

In CA maximum percentage of big firm working people think they might be face mental harrasment challenge

FOR NON US:

Stongest predictor of mental health ***variables*** are -

those who seek ***"treatment"***

Happened ***"work_interfere"***

faced ***"mental_health_consequence"***

"obs_consequence": bserved negative consequences for coworkers with mental health conditions

Note:

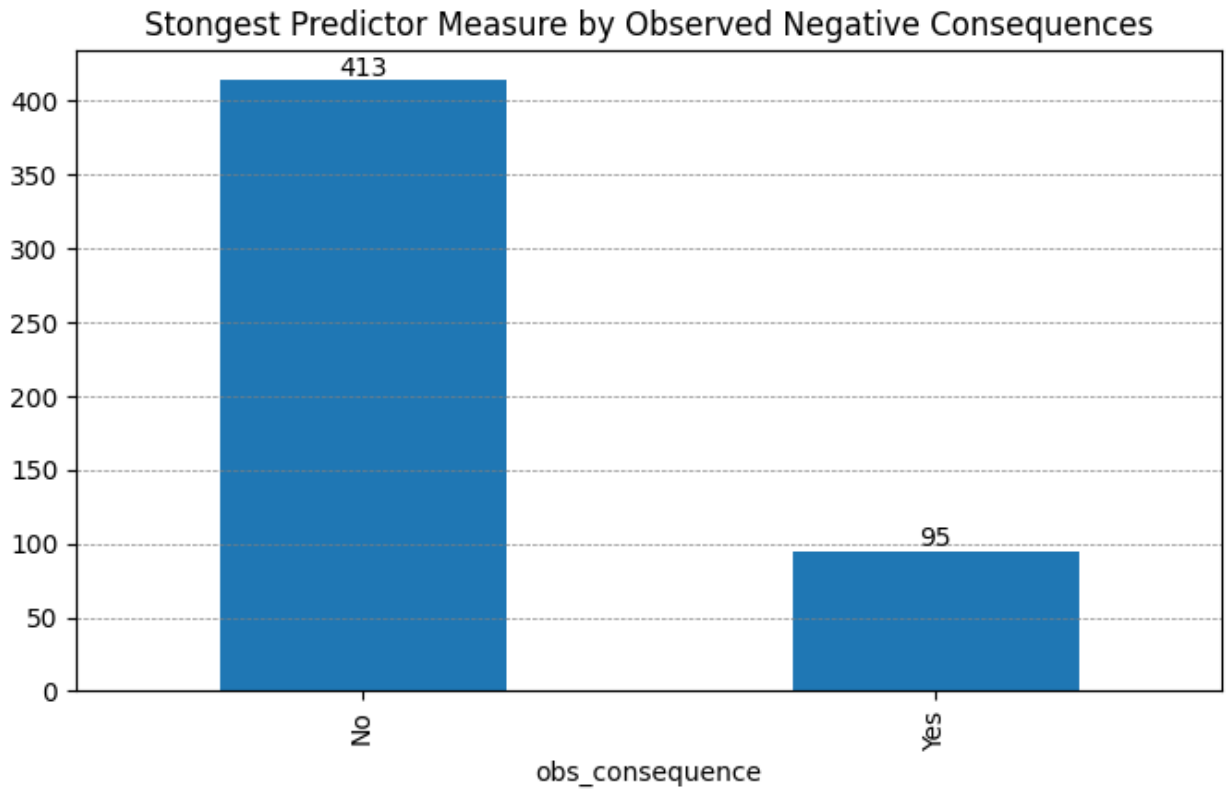
obs_consequence: Have you heard of or observed negative consequences for coworkers with mental health conditions in your workplace?

```
df_NonUSQ2 =
df_NonUS[['Country','family_history','treatment','work_interfere',
```

```
'mental_health_consequence', 'obs_consequence']].copy()
df_NonUSQ2.head()
```

```
           Country family_history treatment work_interfere  \
2           Canada             No        No         Rarely
3   United Kingdom            Yes       Yes          Often
7           Canada             No        No          Never
9           Canada             No        No          Never
11        Bulgaria             No        No          Never

   mental_health_consequence obs_consequence
2                         No              No
3                        Yes             Yes
7                         No              No
9                         No              No
11                        No              No
```

```
df_NonUSQ2tVis=
df_NonUSQ2['obs_consequence'].value_counts().plot.bar(figsize =
(8,4.5))
plt.title('Stongest Predictor Measure by Observed Negative
Consequences')
plt.grid(visible= True, axis = 'y', color ='gray', linestyle ='--',
linewidth = 0.5)

for cols in df_NonUSQ2tVis.containers:
    df_NonUSQ2tVis.bar_label(cols, label_type= 'edge')
```



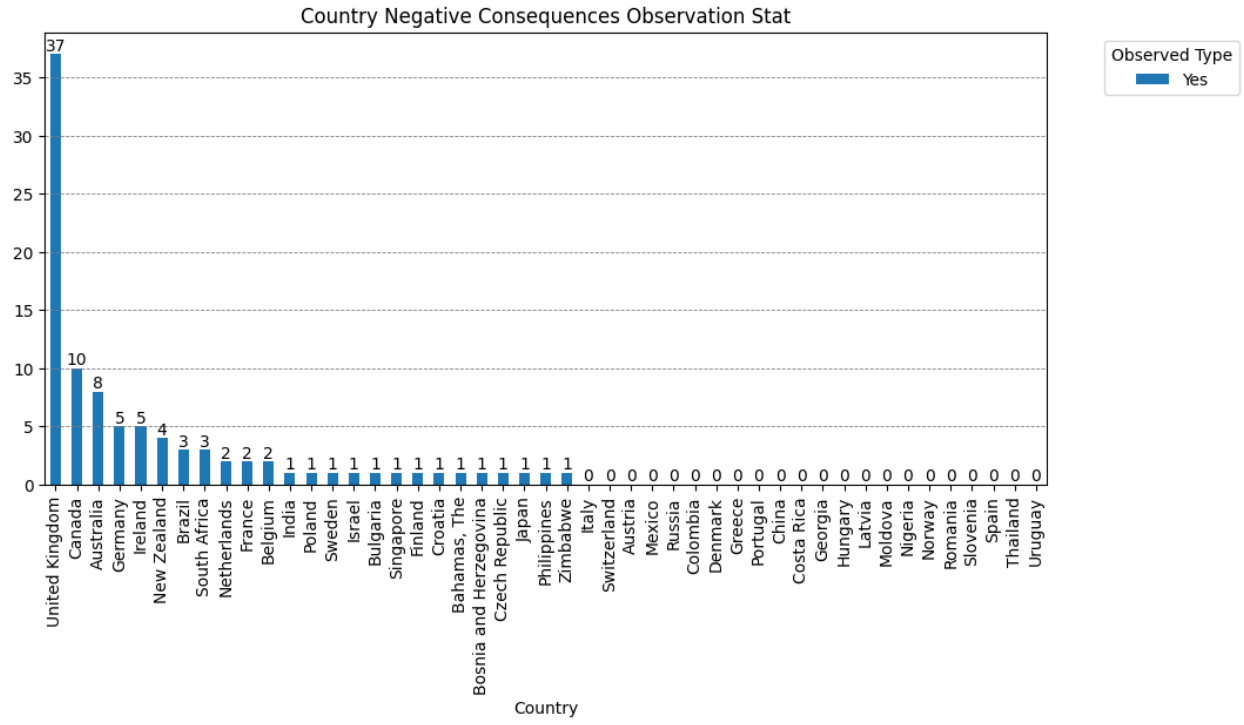

```
df_NonUSQ2piV= df_NonUSQ2[['Country',

'obs_consequence']].groupby(['Country','obs_consequence'],
                                                        observed =
False)['obs_consequence'].count().reset_index(name = 'count')
df_NonUSQ2piV.head()
```

```
        Country obs_consequence  count
0     Australia              No     13
1     Australia             Yes      8
2       Austria              No      3
3  Bahamas, The             Yes      1
4       Belgium              No      4
```

```
df_NonUSQ2piV= MH_pivote(df_NonUSQ2piV, 'obs_consequence', 'Country',
'count')

df_NonUSQ2piV.info()
```

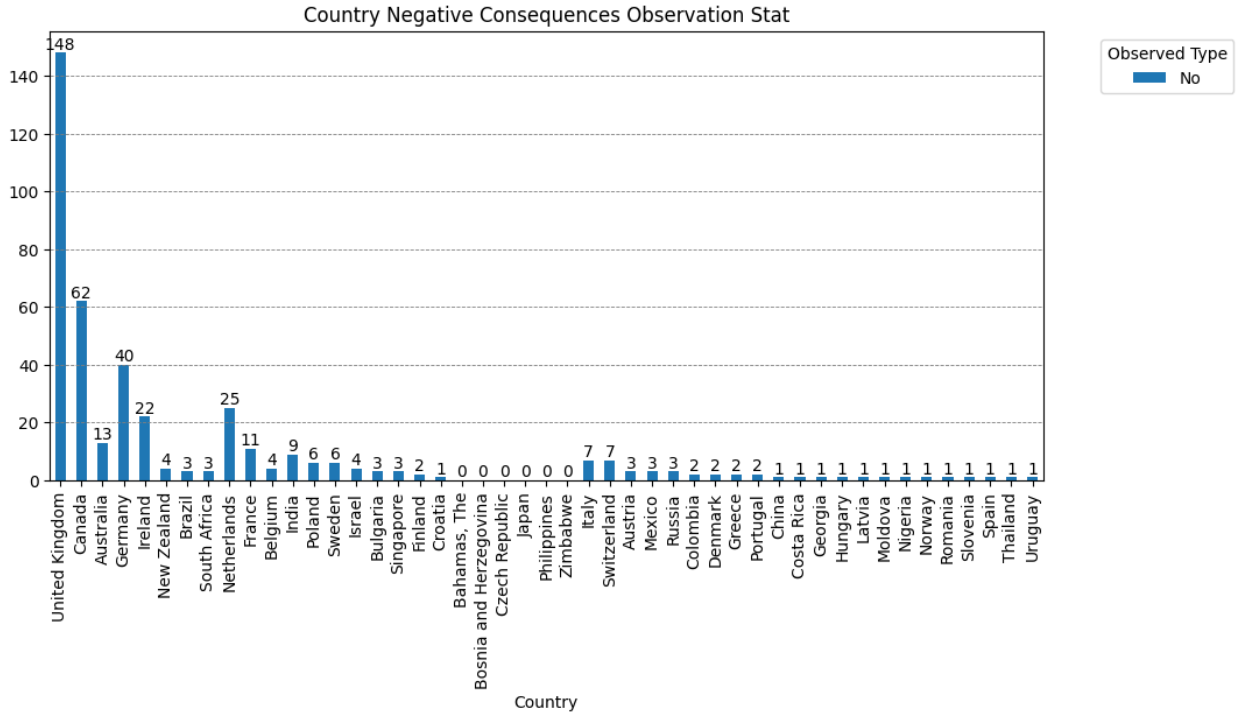
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 47 entries, 0 to 46
Data columns (total 3 columns):
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   Country  47 non-null     object
 1   No       41 non-null     float64
```

```
 2   Yes      25 non-null     float64
dtypes: float64(2), object(1)
memory usage: 1.2+ KB

df_NonUSQ2piV[['Yes','No']] = df_NonUSQ2piV[['Yes','No']].fillna(value
= 0)

df_NonUSQ2piV[['Yes','No']]=
df_NonUSQ2piV[['Yes','No']].astype('int64')

df_NonUSQ2piV = df_NonUSQ2piV.sort_values(by = ['Yes','No'], ascending
= False)

# Country Wise Observed negative consequences measurement Yes
df_NonUSQ2VisP= bar_function(
    dfB = df_NonUSQ2piV,
    colsName1 = 'Country',
    colsName2 = 'Yes',
    graphKind = 'bar',
    wt = 11,
    ht = 5,
    grphTitle = 'Country Negative Consequences Observation Stat',
    legnTitle = 'Observed Type',
    grdAxix = 'y')
```

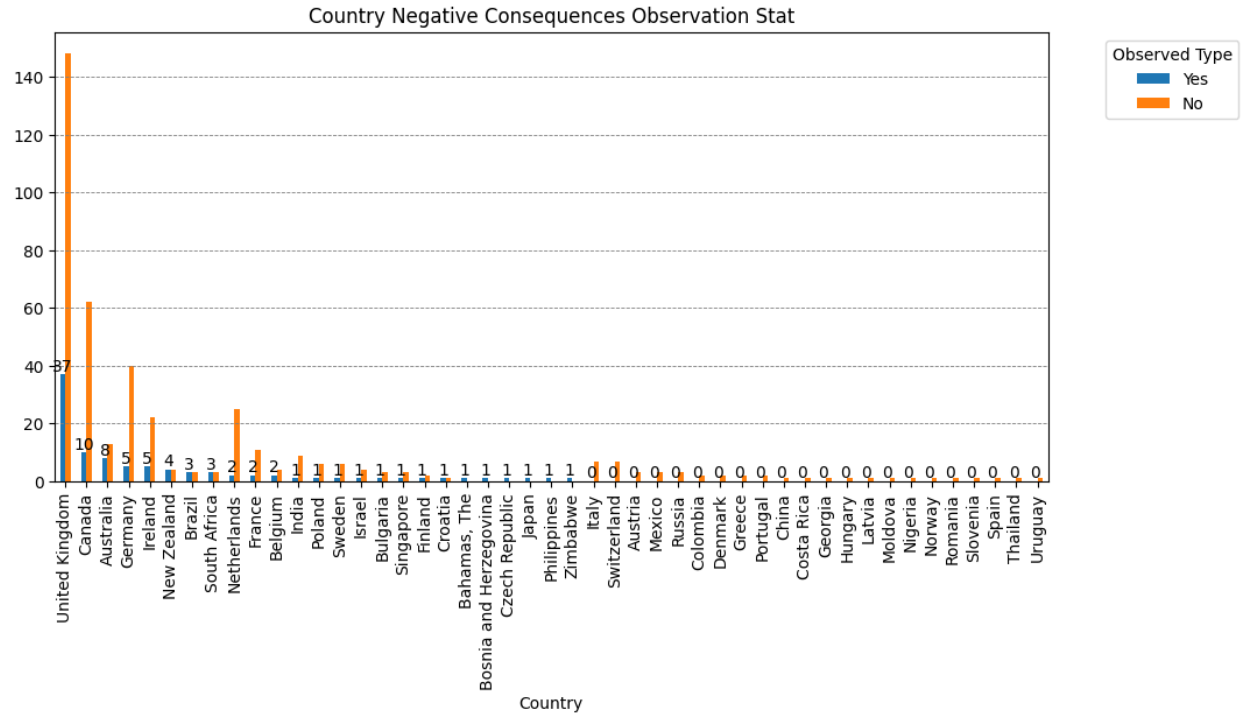



```
# Country Wise Observed negative consequences measurement No
df_NonUSQ2VisP= bar_function(
```

```
    dfB = df_NonUSQ2piV,
    colsName1 = 'Country',
    colsName2 = 'No',
    graphKind = 'bar',
    wt = 11,
    ht = 5,
    grphTitle = 'Country Negative Consequences Observation Stat',
    legnTitle = 'Observed Type',
    grdAxix = 'y')
```




```
# Country Wise Observed negative consequences measurement Yes vs No
df_NonUSQ2VisP= bar_function(
    dfB = df_NonUSQ2piV,
    colsName1 = 'Country',
    colsName2 = ['Yes','No'],
    graphKind = 'bar',
    wt = 11,
    ht = 5,
    grphTitle = 'Country Negative Consequences Observation Stat',
    legnTitle = 'Observed Type',
    grdAxix = 'y')
```




```
df_NonUSQ2.head()
```

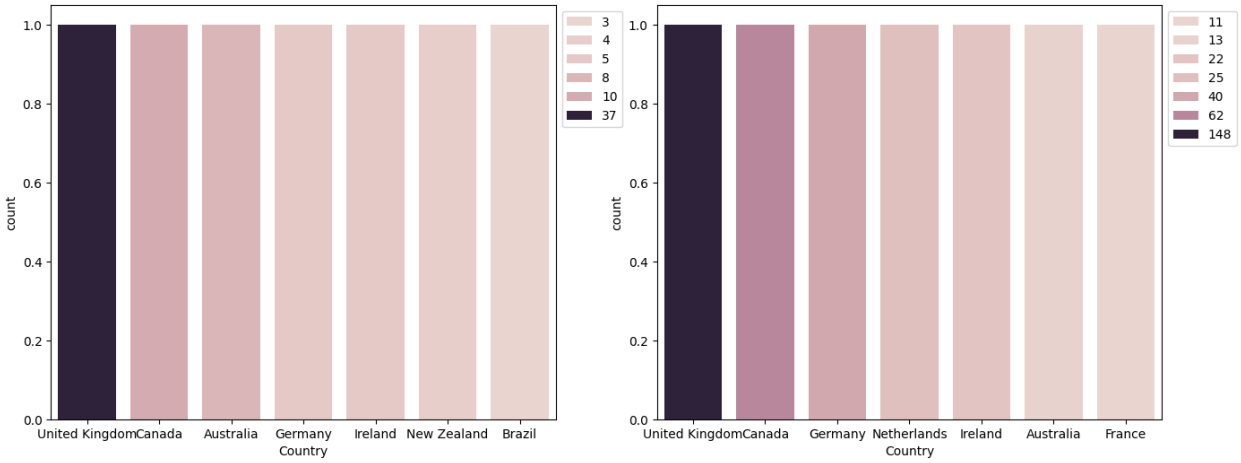
	Country	family_history	treatment	work_interfere	\
2	Canada	No	No	Rarely	
3	United Kingdom	Yes	Yes	Often	
7	Canada	No	No	Never	
9	Canada	No	No	Never	
11	Bulgaria	No	No	Never	

	mental_health_consequence	obs_consequence
2	No	No
3	Yes	Yes
7	No	No
9	No	No
11	No	No

```
plt.figure(figsize= (16,6))
plt.subplot(1,2,1)
sns.countplot(data = df_NonUSQ2piV.sort_values(by ='Yes', ascending =
False).head(7),
              x='Country',
              hue='Yes',legend='full')
plt.legend(bbox_to_anchor= (1.00,1),
           loc = 'upper left')

plt.subplot(1,2,2)
sns.countplot(data = df_NonUSQ2piV.sort_values(by ='No', ascending =
False).head(7),
```

```
              x='Country',
              hue='No', legend='full')
plt.legend(bbox_to_anchor= (1.00,1),
           loc = 'upper left')
```

```
<matplotlib.legend.Legend at 0x1e014f7a210>
```



Conclusion:

In UK maximum percentage people reported they din't observed negative consequences at their work place

5. Solution to Business Objective

What do you suggest the client to achieve Business Objective ?

Ans:

Provide mental health benefits in all company sizes.

Promote supervisor training and wellness programs.

Normalize discussion of mental health through open channels.

Actionable recommendations for HR and leadership teams.

Tangible improvements in workplace culture.

Recommendations for employer policy improvements especially in CA under United states and United Kingdom.

Conclusion

Mens fear most to share their mental health condition with other as compared to women. Infact people fear most when they have any mental patient from their family background, even though they are completely free from mental illness.

In Canada under United States, and UK mental health of the people are very poor. In fact employees fear most to share their mental health condition with the employer, supervisor and coworkers.

Here employer, supervisor need to be more broader minded towards mental patient and introduce more supporting program to reduce burnout, absenteeism, turnover for improving employee well-being and productivity.

Hurrah! You have successfully completed your EDA Capstone Project !!!