GUI Application

In [1]:
```
print("Name : ")
print("Creation a GUI base application for data data visualization")
```

```
Name : 
Creation a GUI base application for data data visualization
```

```
In [ ]: #Done in previous class
        from ipywidgets import widgets
        from IPython.display import display, clear_output
        import pandas as pd
        from tkinter import Tk, filedialog
        import matplotlib.pyplot as plt
        graph_type = ['Choose one.. ','line','bar']
        operation =['choose','mean','max','min','sum','count']
        df = ''
        new_df = ''
        def select_files(b):
            clear_output()
            global df
            root = Tk()
            root.withdraw()
            file_name = filedialog.askopenfilename()
            df = pd.read_csv(file_name)
            print(file_name)
            df.replace( '', float('nan') ,inplace=True)
            df.replace( '0', float('nan') ,inplace=True)
            df = df.dropna()
            display(df)
            groupby_col_widget = widgets.Dropdown(options=df.columns)
            operation_col_widget = widgets.Dropdown(options = df.columns)
            operation_widget = widgets.Dropdown(options = operation)
            groupby_int = widgets.interactive(groupby_dataframe, groupby_column=gro
            display(groupby_int)
        def get_widget():
            global df
            global new_df
            xlabel_widget = widgets.Dropdown(options = new_df.columns)
            ylabel_widget = widgets.Dropdown(options = new_df.columns)
            graph_widget = widgets.Dropdown(options = graph_type)
            graph = widgets.interactive(display_plot, xaxis=xlabel_widget, yaxis= y
            display(graph)
        fileselect = widgets.Button(description="File select")
        fileselect.on_click(select_files)
        display(fileselect)
        #End of what was done in previous class
        # Code for groupby_dataframe function
        def groupby_dataframe(groupby_column, operation_column, operation):
            print("Group By")

        # pre defined code
        def display_plot(xaxis, yaxis, graph_type):
            global new_df
            if(graph_type == 'line'):
                plt.subplots(figsize=(19,8))
                plt.plot(new_df[xaxis], new_df[yaxis], linewidth=3.0)
                plt.xlabel(xaxis)
                plt.xticks(rotation='vertical')
                plt.ylabel(yaxis)
                plt.show()
            elif(graph_type == 'bar'):
                plt.subplots(figsize=(19,8))
                plt.bar(new_df[xaxis], new_df[yaxis], color=['red', 'green','blue',
                plt.xlabel(xaxis)
                plt.xticks(rotation='vertical')
                plt.ylabel(yaxis)
                plt.show()
            else:
                print("Choose valid graph")
        #pre defined code end

        def groupby_dataframe(groupby_column, operation_column, operation):
            global df
            global new_df
            try:
                print(df[operation_column].dtypes)
                if(operation == 'mean'):
                    if (df[operation_column].dtypes != 'float' or df[operation_colu
                        df[operation_column].dtypes = df[operation_column].astype(f
                    new_df = df.groupby(groupby_column)[operation_column].mean.inde
                    display(new_df)
                elif(operation == 'max'):
                    new_df = df.groupby(groupby_column)[operation_column].mean.inde
                    display(new_df)
                elif(operation == 'min'):
                    new_df = df.groupby(groupby_column)[operation_column].mean.inde
                    display(new_df)
                elif(operation == 'sum'):
                    if (df[operation_column].dtypes != 'float' or df[operation_colu
                        df[operation_column].dtypes = df[operation_column].astype(f
                    new_df = df.groupby(groupby_column)[operation_column].mean.inde
                    display(new_df)
                elif(operation == 'count'):
                    new_df = df.groupby(groupby_column)[operation_column].mean.inde
                    display(new_df)
                else:
                    print('Choose valid option')
                get_widget()
            except:
                print('This data is not structured, therefore I cannot perform this
```