**SUMMATIVE SUBMISSION**

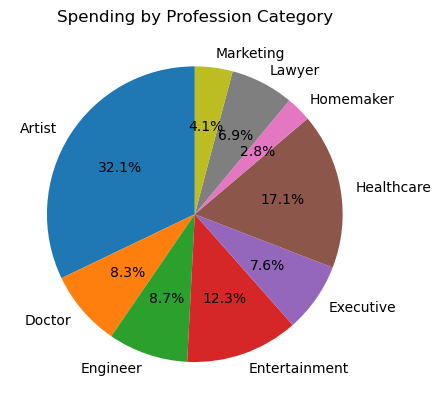
**User Interface Instructions:**

**Search**: This is located at the top of the interface, This allow user to look for a specific customers based on gender or profession or spending score or customerid

**Scroll**: This is located at the right hand side of the data display for user to scroll through all customers information collected

**Graphs**: The Graph were displayed based on the data collected and interpreted for the user to have a summary of the data without having to perform critical task

**Example of the Output**

****

We can see the distribution of average spending scores of all profession collected from the customers

The top three professions with the highest average score are Artist, Healthcare and Entertainment having 32.1%, 17.1% and 12.3% respectively

The three profession with the lowest average spending are Homemaker, Marketing and Lawyer having 2.8%, 4.1% and 6.9% respectively

Other professions such as Engineer, Doctor and Executive have moderate spending score of the total spending score

This shows there is considerable variation in spending score across different profession with some profession being more likely to spend more than others.

**PSEUDO CODE**

**Creating the table**

Set the columns for the table as "CustomerID", 'Gender', 'Age', 'Income($)', 'Score (1-100)', 'Profession', 'Work Experience', and 'Family Size'.

For each column in the table's columns:

Create heading for the column with the text of the column name.

**Creating Pie-Chart for Age group (18-30 and 30 above)**

**Create age groups**

* 1.1. Define bins with ranges [0, 30, 120] and labels ['18-30', '30+']
* 1.2. Cut the data['Age'] column into age groups based on the defined bins and labels
* 1.3. Save the result to a variable called 'age\_groups'

**Calculate the percentage of time spent shopping by age group**

* 2.1. Group the data by 'Age' column
* 2.2. Sum the 'Spending Score (1-100)' column for each age group
* 2.3. Divide the sum of each age group by the total sum of 'Spending Score (1-100)'
* 2.4. Multiply the result by 100 to get the percentage of time spent shopping by age group
* 2.5. Save the result to a variable called 'age\_shopping\_time'

**Create a frame for the shopping chart**

* 3.1. Create a frame called 'shopping\_frame'
* 3.2. Pack the 'shopping\_frame' to the left side of a window

**Create a figure for the shopping chart**

* 4.1. Create a figure called 'shopping\_fig' with a size of 5 by 4 and a dpi of 100
* 4.2. Add a subplot called 'shopping\_plot' to the 'shopping\_fig'
* 4.3. Create a pie chart on the 'shopping\_plot' subplot using the 'age\_shopping\_time' variable
* 4.4. Set the labels for each slice of the pie chart to the age groups in 'age\_shopping\_time.index'
* 4.5. Format the percentage values on each slice to one decimal place with '%1.1f%%'
* 4.6. Set the title of the pie chart to 'Percentage of Time Spent Shopping by Age Group'

**Creating a Bar- chart for the Gender income**

**Create subsets of the data for gender groups**

* 1.1. male\_data = data where Gender is "Male"
* 1.2. female\_data = data where Gender is "Female"

**Calculate total income for each gender group**

* 2.1 male\_income = sum of Annual Income ($) column in male\_data
* 2.2 female\_income = sum of Annual Income ($) column in female\_data

**Create a bar chart**

* 3.1 create a figure and axis object using subplots()
* 3.2 create a bar chart using ax.bar() with the x-axis labels as ["Male", "Female"] and corresponding values of [male\_income, female\_income]
* 3.3 set the y-axis label as "Total Income ($)" using ax.set\_ylabel()
* 3.4 set the chart title as "Total Income by Gender" using ax.set\_title()

**Creating a Pie-Chart for Spending based on profession**

**Group data by profession category and calculate total spending**

1.1 spending\_by\_profession = sum of Spending Score (1-100) column grouped by Profession category

**Create a pie chart**

* 2.1 create a figure object with size (4, 4) and dpi 100 using Figure()
* 2.2 add a subplot to the figure object using fig3.add\_subplot(111)
* 2.3 create a pie chart using ax3.pie() with:

values as spending\_by\_profession values

labels as spending\_by\_profession index

autopct format string as '%1.1f%%'

start angle as 90

* 2.4 set the chart title as 'Spending by Profession Category' using ax3.set\_title()

**COMPREHENSIVE EXPLANATION OF THE PSEUDO CODE**

**Creating the table:**

This involves setting up columns for the table named CustomerID, Gender, Age, Income($), Score (1-100), Profession, Work Experience, and 'Family Size'.

**Creating Pie-Chart for Age group (18-30 and 30 above):**

* We start by creating age groups by defining with ranges [0, 30, 120] and labels ['18-30', '30+'].
* The Age column is then cut into age groups based on the defined bins and labels.
* The result is saved to a variable called Age\_groups
* The percentage of time spent shopping by age group is then calculated by grouping the data by Age column then summing the Spending Score (1-100' column for each age group, dividing the sum of each age group by the total sum of Spending Score (1-100), and then multiplying the result by 100 to get the percentage of time spent shopping by age group. This result is saved to a variable called age\_shopping\_time.
* A frame for the shopping chart is then created and packed to the left side of a window.
* A figure for the shopping chart is also created with a size of 5 by 4 and a dpi of 100.
* A subplot called shopping\_plot is added to the shopping\_fig figure.
* A pie chart is created on the shopping\_plot subplot using the age\_shopping\_time variable.
* The labels for each slice of the pie chart are set to the age groups in age\_shopping\_time.index, and the percentage values on each slice are formatted to one decimal place with '%1.1f%%'.
* The title of the pie chart is set to 'Percentage of Time Spent Shopping by Age Group'.

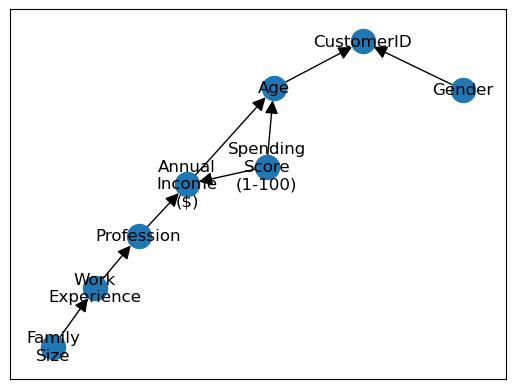
**Creating a Bar- chart for the Gender income:**

* This involves creating subsets of the data for gender groups, where male\_data is the data where Gender is "Male" and female\_data is the data where Gender is "Female".
* The total income for each gender group is then calculated by summing the Annual Income ($) column for male\_data and female\_data, respectively.
* A bar chart is then created by creating a figure and axis object using subplots(), creating a bar chart using ax.bar() with the x-axis labels as ["Male", "Female"] and corresponding values of [male\_income, female\_income], setting the y-axis label as "Total Income ($)" using ax.set\_ylabel(), and setting the chart title as "Total Income by Gender" using ax.set\_title().

**Creating a Pie-Chart for Spending based on profession:**

* This step involves grouping the data by profession category and calculating the total spending for each category.
* A pie chart is then created by creating a figure object with size (4, 4) and dpi 100 using Figure(), adding a subplot to the figure object using fig3.add\_subplot(111), creating a pie chart using ax3.pie() with values as spending\_by\_profession values, labels as spending\_by\_profession index, autopct format string as '%1.1f%%', start angle as 90, and setting the chart title as 'Spending by Profession Category' using ax3.set\_title().

**FLOWCHARTS**

****

**CustomerID**: This is the starting point of the diagram and isn't related to anything else.

**Gender:** There's an arrow from CustomerID to Gender, which means that there may be a difference in the number of men and women among your customers.

**Age**: There's an arrow from CustomerID to Age, which means that there may be a difference in the ages of your customers.

**Annual Income ($):** There's an arrow from Age to Annual Income ($), which suggests that there may be a relationship between age and income. An arrow from Spending Score (1-100) to Annual Income ($), which suggests that there may be a relationship between income and spending.

**Spending Score (1-100):** An arrow from Age to Spending Score (1-100), which suggests that there may be a relationship between Age and Spending Score. There's also an arrow from Annual Income ($) to Spending Score (1-100), which suggests that there exist a relationship between the Annual Income and Spending Score.

**Profession**: I placed an arrow from Annual Income ($) to Profession, which suggests that certain professions may pay more than others.

**Work Experience**: There's an arrow from Profession to Work Experience, which suggests that certain professions require more experience or education than others.

**Family Size:** An arrow from Work Experience to Family Size, which suggests that people with more experience may have had more time to start a family.

The flow chart displays connections between the categories of information in the data. For example age may be related to income and spending, profession may be related to annual income and work experience. Taking a look at the flow chart, anyone can get a better understanding of the data and find areas that have potential of deep learning and understanding

**PROBLEM FACED WHILE BUIDING THE GUI**

**Data preparation**: Before plotting the charts and describing what the graph should interpret, the data needs to be preprocessed to ensure that it is in the correct format. This includes cleaning the data, removing any missing values, and grouping the data by profession, gender and family size.

**GUI layout**: Designing a visually appealing and intuitive GUI layout is also a challenge. Deciding on the placement and size of each chart, as well as any additional interface elements such as buttons or text boxes, requires careful consideration.

**Chart Building**: Creating multiple pie charts requires assigning to each chart's colors, labels, and other properties. Making sure that each chart looks visually appealing and easy to read for the user.

**User interaction**: Depending on the intended use of the GUI, allowing the user to interact with the charts or data may be necessary..

**Performance**: Depending on the size of the dataset, generating and displaying multiple charts could potentially impact the GUI's performance. Careful consideration needs to be given to optimizing the code and minimizing any delays or any error in displaying the data and charts.

In Order to overcame this problems, I use the following approaches:

**Reading and cleaning the data**: I employ the pandas library to read the data and then cleaned the data by dropping any missing values and removing duplicates although there weren’t any duplicate and missing values.

**Creating the GUI layout**: I used the Tkinter library to create the GUI layout, including the buttons, labels, and frames. I arranged the layout to be visually appealing and easy for user to quickly understand the data at hand

**Plotting the pie charts**: I used the Matplotlib library to plot the pie charts and bar chart based on spending by profession and income earned by each gender and also time spent shopping by age group. I ensured that the labels, colors, and sizes of the charts were clear and easy to understand.

**Handling user inputs**: I used the Tkinter library to handle user inputs and ensure that the appropriate charts were displayed based on user selection although there were still problems faced and soon to be handled so I plotted the graph from the start to give the user the overview of everything needed to be known.

**Testing and debugging**: I made sure the GUI functioned as expected and that there were no errors or bugs.

I learned some things by working through this problems:

* Cleaning of Data and preparing of data are essential steps in the building Graphical User Interface. Small issues in the data such as missing values or inconsistent values canl cause major problems when generating pie charts or bar chart or other form of visualization.
* It is important to carefully consider the layout and design of the GUI to ensure that users can easily interpret the data. it is also important to think about how the user will interact with the visualizations and how they can be combined for the user to ger a better picture of the data.
* When working with large datasets, it is often necessary to use specialized tools and libraries to efficiently process and analyze the data given. For example in this project, I used Pandas for data preprocessing the daata and Matplotlib for generating the visualizations based on the data and also Tkinter to create the GUI for the data.
* Debugging and problem-solving are necessary skills when working on any software project, and this project was no exception. I learned to work through errors and issues then using print statements and other debugging techniques to identify and resolve the root cause of problems.

The GUI has could use more improvement while I’m gaining more knowledge in building a better one. These includes:

**Limited interactivity**: The GUI is static and does not allow users interaction with the data and graphs such as filtering and sorting the data. In order to improve this, I would add more features that allow users to interact with the data in real-time and help more data-analyzing concepts.

**Little visualization options**: The current GUI only allows for pie charts and bar charts and not more than three to be plotted. To improve this, Adding more visualization options e.g. line charts, and scatterplots would provide users with a more comprehensive view of the data.

**Limited data exploration**: The GUI only focuses on three variables, spending based on profession income based on gender and shopping time based on age. I would lie add more variables to the GUI to provide users with better view of the data.

**Small data size**: The dataset used for this GUI is small. In future case, I would use a larger dataset to provide users with a more robust analysis of the data and also provide more detail revealing tools.

**Limited platform support**: The current GUI was built using PyGui, which is limited to the Python programming language. To create a better GUI, I would consider building the GUI using other platforms such as R Shiny or Tableau which support multiple programming languages and provide a more interactive user experience.

**REFERENCE:**

Customer Data, Data Scientist Anna (Owner), manually generated, Open Database License, https://www.kaggle.com/datasets/datascientistanna/customers-dataset

Pandas code helpers 1004 , StarkOverflow, Official Documentation is hosted on PyData.org: <https://pandas.pydata.org/pandas-docs/stable>. License: BSD 3

Matplotlib, J. D. Hunter, ‘Matplotlib A 2D Graphics Environment’, Computing in Science & Engineering, vol. 9, no. 3, pp. 90-95, 2007, <https://doi.org/10.1109/MCSE.2007.55>. Development mailing list: [matplotlib-devel@python.org](mailto:matplotlib-devel@python.org)

Tkinter, John W. Shipman, [GitHub - HamidMusayev/Tkinter-en: Complete Python Tkinter reference](https://github.com/HamidMusayev/Tkinter-en)