Item Count per Month: I noticed that the items in each transaction were separated by a ";" so I used a code that would separate each item by the semicolon. I created a new dataframe for it and added columns whose values would contain the extracted month from each transaction. Afterwards, I created three more columns that would first extract the Quantity(x#), Real Quantity(#) and Item. I had to create the Quantity column first so that I would be able to easily extract the number of items bought per item/product type. I used an approach or code that is somehow similar to the initial steps(.split and .extract). The last step was using the .groupby and .sum function to total the item count per item/product type and per month. I presented it in a pivot table and a multiple line chart.

Total Sale Value of Each Item per Month: I created a def function that would split the items by the ";". I did not use the approach I had in the first task because I felt the need for a dictionary. I used the def function I created to parse the transaction items and create the dictionary. Afterwards, I extracted the month in a different manner by first formatting the transaction dates. I then created another dataframe wherein I filtered transactions that only had one item and proceeded to create more columns, namely Category, Item and Quantity. In case there were multiple transactions that only had one item, I used .drop_duplicates function to only keep the first occurrence. With the data I have, I was able to divide the transaction_value by the quantity and get the price for one item. I created another dataframe to house all of the data I have to clearly see the prices of each product and created a copy of it to be able to make the necessary changes. Using a for loop ad .loc function I was able to multiply the prices of each item to the dataframe with split items accordingly per month.

Customer Status/Metrics: I have 5 Customer 'Status' --- Engaged, Inactive, Repeater, Non-Repeater, New. For each status, I created a def function that would return a value of 1 if customer fits the said status and 0 if not for each month. I used .apply function for the def functions of each status. I first switched the columns and rows using the .transpose to be able to use the .apply(pd.value_counts) to count the 0s and 1s, and filtered out the 1s using .loc function.