AVIATION DATA

BUSINESS UNDERSTANDING

With this data We are going to be able to establish the various columns present in the dataset provided which eventually are going to help us know which aircraft to purchase. After the data filtering process we are going to identify which aircraft will have the lowest risk for the company to start this new business endavour

Importing the relevant libraries

With this step we are We are supposed to import the relevant libraries to be able to conduct the data cleaning and filtering process In this case we are supposed to import the pandas library to conduct Importing the matplotlib.pyplot this is to help us in the visualization of the data after the data cleaning process

In [16]:
```
import pandas as pd
import matplotlib.pyplot as plt
```

Loading the dataset

In this part we are supposed to load the dataset that was presented to us so that we can identify the columns and rows that need the cleaning and fitering process

Since this data is to big to be presented we only show the first five rows of the data with all the columns present

In [17]:
```
Aviation_Data = pd.read_csv("AviationData.csv", encoding="latin1")
Aviation_Data.head()
```

```
C:\Users\Serita\AppData\Local\Temp\ipykernel_21540\342147473.py:1: DtypeWarning: Col
umns (6,7,28) have mixed types. Specify dtype option on import or set low_memory=Fal
se.
  Aviation_Data = pd.read_csv("AviationData.csv", encoding="latin1")
```

Out[17]:

	Event.Id	Investigation.Type	Accident.Number	Event.Date	Location	Countr
0	20001218X45444	Accident	SEA87LA080	1948-10-24	MOOSE CREEK, ID	Unite State
1	20001218X45447	Accident	LAX94LA336	1962-07-19	BRIDGEPORT, CA	Unite State
2	20061025X01555	Accident	NYC07LA005	1974-08-30	Saltville, VA	Unite State
3	20001218X45448	Accident	LAX96LA321	1977-06-19	EUREKA, CA	Unite State
4	20041105X01764	Accident	CHI79FA064	1979-08-02	Canton, OH	Unite State

5 rows × 31 columns

Filtering the data

In this stage we are supposed to filter the data by removing all the rows with null or missing values to soften the amount of data present and reduce the amount of data to be filtered and cleaned

In [18]:

```
Aviation_Data.dropna()

Aviation_Data.dropna(axis=1)

Aviation_Data.head()
```

Out[18]:

	Event.Id	Investigation.Type	Accident.Number	Event.Date	Location	Countr
0	20001218X45444	Accident	SEA87LA080	1948-10-24	MOOSE CREEK, ID	Unite State
1	20001218X45447	Accident	LAX94LA336	1962-07-19	BRIDGEPORT, CA	Unite State
2	20061025X01555	Accident	NYC07LA005	1974-08-30	Saltville, VA	Unite State
3	20001218X45448	Accident	LAX96LA321	1977-06-19	EUREKA, CA	Unite State
4	20041105X01764	Accident	CHI79FA064	1979-08-02	Canton, OH	Unite State

5 rows × 31 columns

Selecting the relevant columns

With this part we are to select the only columns that we need even after removing the missing values and null values

This step also enables us to get the set of columns that we need to be able to conduct the relevant study after we have cleaned the data even further

With these relevant columns we are able to get what need to be able to conduct the necessary visualizations of this new dataset that we have created

In [19]:
```
relevant_columns = ["Event.Id", "Event.Date", "Total.Fatal.Injuries", "Location", "
```

Filtering The new dataset

In this process we filter what is in our new dataset which are the relevant columns that we have selected, so that we can remove whatever is not necessary in this case the null values which will not help us in either the data visualization process or the data cleaning process

Here we also show the shape of our new dataset ensuring that this new dataset ONLY has the columns that we selected

In [21]:
```
existing_columns = Aviation_Data.columns.intersection(relevant_columns)


Aviation_Data = Aviation_Data[existing_columns]

Aviation_Data.dropna()

Aviation_Data.dropna(axis=1)

print(Aviation_Data.shape)
```
```
(88889, 9)
```

Showing the new dataset

We display the new aviation dataset with only the relevant columns we need to simplify getting relevant information from this new data

In [22]:
```
Aviation_Data
```

Out[22]:

	Event.Id	Event.Date	Location	Country	Make	Model	Number.of
0	20001218X45444	1948-10-24	MOOSE CREEK, ID	United States	Stinson	108-3	
1	20001218X45447	1962-07-19	BRIDGEPORT, CA	United States	Piper	PA24-180	
2	20061025X01555	1974-08-30	Saltville, VA	United States	Cessna	172M	
3	20001218X45448	1977-06-19	EUREKA, CA	United States	Rockwell	112	
4	20041105X01764	1979-08-02	Canton, OH	United States	Cessna	501	
...	...	...	...	...	...	...	
88884	20221227106491	2022-12-26	Annapolis, MD	United States	PIPER	PA-28-151	
88885	20221227106494	2022-12-26	Hampton, NH	United States	BELLANCA	7ECA	
88886	20221227106497	2022-12-26	Payson, AZ	United States	AMERICAN CHAMPION AIRCRAFT	8GCBC	
88887	20221227106498	2022-12-26	Morgan, UT	United States	CESSNA	210N	
88888	20221230106513	2022-12-29	Athens, GA	United States	PIPER	PA-24-260	

88889 rows × 9 columns

Identifying the least number of accidents From a particular make

In this step we ensure that we have the least number of accidents that a specific make has

We are shown the makes that have made the least number of appearances in this accident dataset so that we can be able to identify which aircraft make has the least amount of risks and show promise when flown

This will be able to give us a clear image of what we as the stakeholders can invest and which aircraft to buy with the least risk present

This information is very critical

In [23]:
```
bottom_make_series = Aviation_Data['Make'].value_counts()


sorted_make_series = bottom_make_series.sort_values()


bottom_make = list(sorted_make_series.index[:5])
bottom_make_counts = list(sorted_make_series.values[:5])

# Print the results
print("Makes:", bottom_make)
print("Counts:", bottom_make_counts)
```

```
Makes: ['ERICKSON AIR CRANE', 'George', 'Timothy J Brown', 'James B. Edwards', 'Ultr
alight']
Counts: [1, 1, 1, 1, 1]
```

Data visualization

Here we present our data with the relevant information that we have gotten

We get a clear picture of what we need as we plot both a scatter plot to show us the relationship between the number of engines and total fatal engines and a bar graph to show which model has those specific number of engines

With this information we are able to identify how many number of engines cause the least number of fatal injuries

we are also able to know an aircraft with how many number of engines cause the least amount of accidents

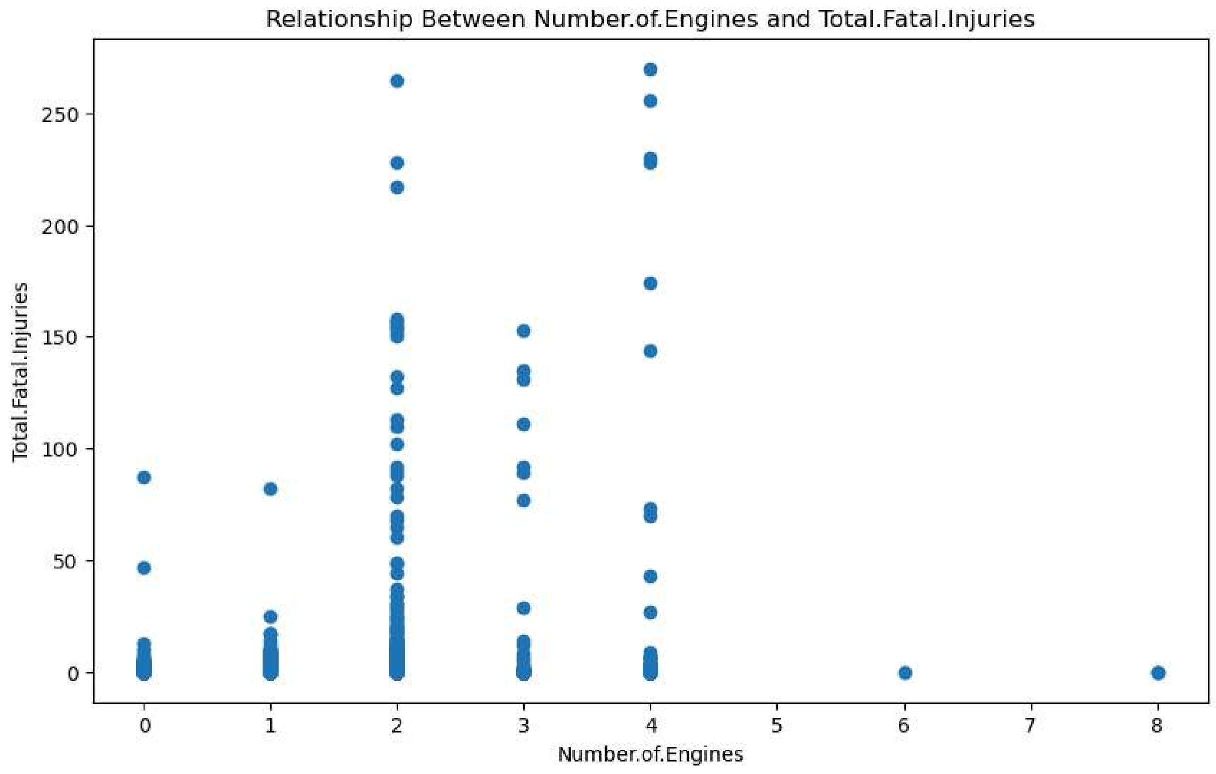
This way we will identify a type of aircraft that will have the least risk present

Scatterplot for data visualization

This specific scatterplot shows us the relationship between the number of engines and the number of accidents that have caused fatalities

It show gives us a clear indication how many number of engines we need in the aircraft that we are supposed to buy

In [24]:
```
scatter_plot_title = 'Relationship Between Number.of.Engines and Total.Fatal.Injuri
Number_of_Engines_label = 'Number.of.Engines'
Total_Fatal_Injuries_label = 'Total.Fatal.Injuries'

Number_of_Engines_figure, ax = plt.subplots(figsize=(10, 6))


ax.scatter(Aviation_Data['Number.of.Engines'], Aviation_Data['Total.Fatal.Injuries'
ax.set_title(scatter_plot_title)

ax.set_xlabel(Number_of_Engines_label)

ax.set_ylabel(Total_Fatal_Injuries_label)

plt.show()

type(Number_of_Engines_figure), len(Number_of_Engines_figure.axes)
```



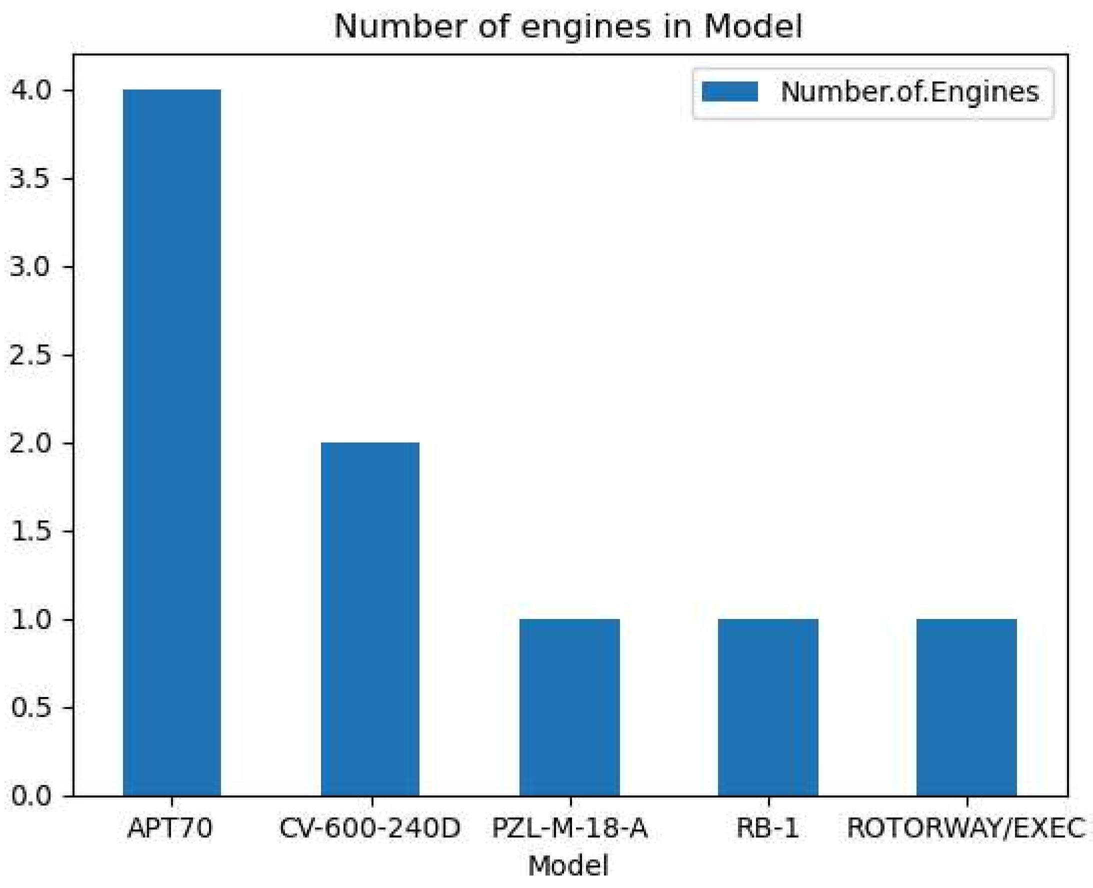
```
Out[24]: (matplotlib.figure.Figure, 1)
```

A bar graph for visualization of data

This specific bar graph where we plot the model of a specific aircraft against the number of engines will now narrow down our search to which aircraft model we need

With this it corresponds to the above scatter plot where we were identifying the relationship of number of engines to the number fatalities caused and with this we are able to identify which aircraft model and the number of engines to give us a specific model that doesn't have a lot of risks

```
In [25]: x = "Model"
y = "Number.of.Engines"

subset = Aviation_Data[Aviation_Data["Model"].isin([
    "APT70", "RB-1", "PZL-M-18-A", "ROTORWAY/EXEC", "CV-600-240D"
])]
grouped_by_make = subset.groupby("Model").mean(numeric_only=True).reset_index()
ax = grouped_by_make.plot.bar(x=x, y=y, rot=0, title="Number of engines in Model");
```



Saving the new dataset

After the cleaning process, filtering process and visualizing on what product we need we save the new dataset to be presented for our stakeholders and give them a picture of what they will need to invest in

In [26]:
```
output_file_name = 'Cleaned_Aviation_Data.csv'

Aviation_Data.to_csv(output_file_name, index=False)

print(f'Cleaned data has been saved to {output_file_name}')
```

```
Cleaned data has been saved to Cleaned_Aviation_Data.csv
```

In [ ]: