TITLE: ASSESSING LOW-RISK AIRCRAFT FO AVIATION EXPANSION

INTRODUCTION

Airplane Image

Datan Africa is interested in purchasing and operating airplanes for private and commercial pupo diversify its portfolio.The risks of this venture are unknown and an in depth risk analysis is require inorder to determine the aircraft with the lowest risk.The data for this analysis was obtained from National Transportation Safety Board that includes aviation accident data from 1962 to 2023 abo aviation accidents and selected incidents in the United States and international waters.

1.0 OBJECTIVES

```
1.To determine the aircraft with the lowest risk.
2.To translate findings into actionable insights.
```

1.1 BUSINESS UNDERSTANDING

The aircraft industry is diverse, requiring investors to understand its complexities before choosing Datan Africa aims to invest in purchasing and operating planes for commercial and private use, fo this research on identifying the least risky aircraft. The primary risk is accidents, which can make airlines and planes unattractive to customers, leading to significant financial losses. Given that ai are costly with minimal salvage value post-accident, our research will concentrate on analyzing tl frequency and impact of accidents associated with different aircraft.

2.0 DATA UNDERSTANDING

2.0.1 Uploading the relevant libraries and uploading the data.

In [3]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
%matplotlib inline
warnings.filterwarnings("ignore")
df=pd.read_csv("/home/moringa/ochieng/phase_one_project/data/AviationData
df
```

Out[3]:

	Event.Id	Investigation.Type	Accident.Number	Event.Date	Location
0	20001218X45444	Accident	SEA87LA080	1948-10-24	MOOSE CREEK, ID
1	20001218X45447	Accident	LAX94LA336	1962-07-19	BRIDGEPORT, CA
2	20061025X01555	Accident	NYC07LA005	1974-08-30	Saltville, VA
3	20001218X45448	Accident	LAX96LA321	1977-06-19	EUREKA, CA
4	20041105X01764	Accident	CHI79FA064	1979-08-02	Canton, OH
...	...	...	...	...	...
88884	20221227106491	Accident	ERA23LA093	2022-12-26	Annapolis, MD
88885	20221227106494	Accident	ERA23LA095	2022-12-26	Hampton, NH
88886	20221227106497	Accident	WPR23LA075	2022-12-26	Payson, AZ
88887	20221227106498	Accident	WPR23LA076	2022-12-26	Morgan, UT
88888	20221230106513	Accident	ERA23LA097	2022-12-29	Athens, GA

88889 rows × 31 columns

2.0.2 Data overview

In [4]:

```
df.info()#gives a general summary of the data.
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 88889 entries, 0 to 88888
Data columns (total 31 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   Event.Id                88889 non-null  object 
 1   Investigation.Type      88889 non-null  object 
 2   Accident.Number         88889 non-null  object 
 3   Event.Date              88889 non-null  object 
 4   Location                88837 non-null  object 
 5   Country                 88663 non-null  object 
 6   Latitude                34382 non-null  object 
 7   Longitude               34373 non-null  object 
 8   Airport.Code            50249 non-null  object 
 9   Airport.Name            52790 non-null  object 
 10  Injury.Severity         87889 non-null  object 
 11  Aircraft.damage         85695 non-null  object 
 12  Aircraft.Category       32287 non-null  object 
 13  Registration.Number     87572 non-null  object 
 14  Make                    88826 non-null  object 
 15  Model                   88797 non-null  object 
 16  Amateur.Built           88787 non-null  object 
 17  Number.of.Engines       82805 non-null  float64
 18  Engine.Type             81812 non-null  object 
 19  FAR.Description         32023 non-null  object 
 20  Schedule                12582 non-null  object 
 21  Purpose.of.flight       82697 non-null  object 
 22  Air.carrier             16648 non-null  object 
 23  Total.Fatal.Injuries    77488 non-null  float64
 24  Total.Serious.Injuries  76379 non-null  float64
 25  Total.Minor.Injuries    76956 non-null  float64
 26  Total.Uninjured         82977 non-null  float64
 27  Weather.Condition       84397 non-null  object 
 28  Broad.phase.of.flight   61724 non-null  object 
 29  Report.Status           82508 non-null  object 
 30  Publication.Date        75118 non-null  object 
dtypes: float64(5), object(26)
memory usage: 21.0+ MB
```

In [5]:
```
df.shape#Gives the number of rows and columns of the data frame
```

Out[5]:
```
(88889, 31)
```

In [6]:
```
#Checking the first and last 5 rows of the dataframe respectively.
df.head()
df.tail()
```

Out[6]:

	Event.Id	Investigation.Type	Accident.Number	Event.Date	Location	Cour
88884	20221227106491	Accident	ERA23LA093	2022-12-26	Annapolis, MD	Uni Sta
88885	20221227106494	Accident	ERA23LA095	2022-12-26	Hampton, NH	Uni Sta
88886	20221227106497	Accident	WPR23LA075	2022-12-26	Payson, AZ	Uni Sta
88887	20221227106498	Accident	WPR23LA076	2022-12-26	Morgan, UT	Uni Sta
88888	20221230106513	Accident	ERA23LA097	2022-12-29	Athens, GA	Uni Sta

5 rows × 31 columns

In [7]:

```
#checking for missing values in the dataframe
missing_values=df.isna().sum()
missing_values

#df has missing values except in the first four columns.
```

Out[7]:

```
Event.Id                      0
Investigation.Type            0
Accident.Number               0
Event.Date                    0
Location                     52
Country                     226
Latitude                  54507
Longitude                 54516
Airport.Code              38640
Airport.Name              36099
Injury.Severity            1000
Aircraft.damage            3194
Aircraft.Category         56602
Registration.Number        1317
Make                         63
Model                        92
Amateur.Built               102
Number.of.Engines          6084
Engine.Type                7077
FAR.Description           56866
Schedule                  76307
Purpose.of.flight          6192
Air.carrier               72241
Total.Fatal.Injuries      11401
Total.Serious.Injuries    12510
Total.Minor.Injuries      11933
Total.Uninjured            5912
Weather.Condition          4492
Broad.phase.of.flight     27165
Report.Status              6381
Publication.Date          13771
dtype: int64
```

In [8]:

```
#checking fo column names
df.columns
```

```
Out[8]: Index(['Event.Id', 'Investigation.Type', 'Accident.Number', 'Event.Date',
               'Location', 'Country', 'Latitude', 'Longitude', 'Airport.Code',
               'Airport.Name', 'Injury.Severity', 'Aircraft.damage',
               'Aircraft.Category', 'Registration.Number', 'Make', 'Model',
               'Amateur.Built', 'Number.of.Engines', 'Engine.Type', 'FAR.Description
               'Schedule', 'Purpose.of.flight', 'Air.carrier', 'Total.Fatal.Injuries
               'Total.Serious.Injuries', 'Total.Minor.Injuries', 'Total.Uninjured',
               'Weather.Condition', 'Broad.phase.of.flight', 'Report.Status',
               'Publication.Date'],
              dtype='object')
```

The dataframe contains 88889 variables(rows)and 31 observations(columns).It also contains mis values.

2.1 DATA ANALYSIS

2.1.1 Data cleaning

```
In [9]: #relevant columns for analysis.
        relevant_columns = ['Event.Date', 'Country', 'Injury.Severity','Location'
                   'Aircraft.damage','Make','Model','Amateur.Built','Number.of.En
                    'Engine.Type','Purpose.of.flight','Total.Fatal.Injuries','Tot
        print(relevant_columns)
```

```
['Event.Date', 'Country', 'Injury.Severity', 'Location', 'Aircraft.damage', '
'Model', 'Amateur.Built', 'Number.of.Engines', 'Engine.Type', 'Purpose.of.fli
'Total.Fatal.Injuries', 'Total.Uninjured', 'Weather.Condition',
'Broad.phase.of.flight']
```

```
In [10]: #creating a new dataframe with onlythe relevant columns.
         df_relevant=df[relevant_columns]
         #confirming the shape of our new data frame.
         df_relevant.shape
         #our new data frame contains 15 columns.
```

```
Out[10]: (88889, 15)
```

```
In [11]: #removing the dot and stripping the whitespaces from the column names.
         df_relevant.columns = df_relevant.columns.str.replace('.', ' ', regex=Fal
         df_relevant.columns
```

```
Out[11]: Index(['Event Date', 'Country', 'Injury Severity', 'Location',
                'Aircraft Damage', 'Make', 'Model', 'Amateur Built',
                'Number Of Engines', 'Engine Type', 'Purpose Of Flight',
                'Total Fatal Injuries', 'Total Uninjured', 'Weather Condition',
                'Broad Phase Of Flight'],
               dtype='object')
```

```
In [12]: #printing the first 5 rows of our dataframe
         df_relevant.head()
```

Out[12]:

	Event Date	Country	Injury Severity	Location	Aircraft Damage	Make	Model	Amateur Built
0	1948-10-24	United States	Fatal(2)	MOOSE CREEK, ID	Destroyed	Stinson	108-3	No
1	1962-07-19	United States	Fatal(4)	BRIDGEPORT, CA	Destroyed	Piper	PA24-180	No
2	1974-08-30	United States	Fatal(3)	Saltville, VA	Destroyed	Cessna	172M	No
3	1977-06-19	United States	Fatal(2)	EUREKA, CA	Destroyed	Rockwell	112	No
4	1979-08-02	United States	Fatal(1)	Canton, OH	Destroyed	Cessna	501	No

In [13]:
```
df_relevant.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 88889 entries, 0 to 88888
Data columns (total 15 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   Event Date            88889 non-null  object 
 1   Country               88663 non-null  object 
 2   Injury Severity       87889 non-null  object 
 3   Location              88837 non-null  object 
 4   Aircraft Damage       85695 non-null  object 
 5   Make                  88826 non-null  object 
 6   Model                 88797 non-null  object 
 7   Amateur Built         88787 non-null  object 
 8   Number Of Engines     82805 non-null  float64
 9   Engine Type           81812 non-null  object 
 10  Purpose Of Flight     82697 non-null  object 
 11  Total Fatal Injuries  77488 non-null  float64
 12  Total Uninjured       82977 non-null  float64
 13  Weather Condition     84397 non-null  object 
 14  Broad Phase Of Flight 61724 non-null  object 
dtypes: float64(3), object(12)
memory usage: 10.2+ MB
```

In [ ]:
```
#filling the missing values with the approrpriate mean,median,mode
df_relevant['Injury Severity'].fillna(df_relevant['Injury Severity'].mode
df_relevant.dropna(subset=["Location"],inplace=True)
df_relevant.dropna(subset=["Country"],inplace=True)
df_relevant.dropna(subset=["Aircraft Damage"],inplace=True)
df_relevant.dropna(subset=["Engine Type"],inplace=True)
df_relevant['Broad Phase Of Flight'].fillna(df_relevant['Broad Phase Of F
df_relevant['Total Fatal Injuries'].fillna(df_relevant['Total Fatal Injur
df_relevant=df_relevant.dropna()
df_relevant.shape
```

In [ ]:
```
our data frame now contains cleaned 65184rows x 15 columns.
```

In [15]:
```
#adding a column Year.Gives the year when the event occurred.
df_relevant['Year'] = df_relevant['Event Date'].str[0:4]

df_relevant["Year"]
```

Out[15]:
```
0        1948
1        1962
3        1977
6        1981
7        1982
         ...
88639    2022
88647    2022
88661    2022
88735    2022
88767    2022
Name: Year, Length: 70667, dtype: object
```

In [16]:
```
#removing Event Date from the data frame.
df_relevant.drop(columns=["Event Date"],inplace=True)
```

In [17]:
```
type(df_relevant["Year"])
```

Out[17]:
```
pandas.core.series.Series
```

In [18]:
```
#changing the Year to numeric type
df_relevant['Year'] = pd.to_numeric(df_relevant['Year'], errors='coerce')
```

Cleaning the columns

In [19]:
```
#formatting the country columns
df_relevant["Country"]=df_relevant["Country"].str.lower().str.title()
```

In [20]:
```
#dropping rows with the attribute"availablein the injury severity column
df_relevant.drop(df_relevant[df_relevant["Injury Severity"] == 'Unavailab
df_relevant["Injury Severity"].value_counts()
```

Out[20]:
```
Non-Fatal      57655
Fatal(1)        4514
Fatal           2831
Fatal(2)        2721
Incident        1061
Fatal(3)         817
Fatal(4)         595
Fatal(5)         139
Fatal(6)         103
Minor             69
Fatal(7)          32
Fatal(8)          31
Serious           25
Fatal(10)         10
Fatal(14)          5
Fatal(9)           5
Fatal(11)          4
Fatal(12)          3
Fatal(25)          3
Fatal(13)          3
Fatal(34)          2
Fatal(18)          2
Fatal(70)          2
Fatal(82)          2
Fatal(17)          2
Fatal(23)          2
Unavailable        2
Fatal(29)          2
Fatal(87)          1
Fatal(73)          1
Fatal(270)         1
Fatal(110)         1
Fatal(27)          1
Fatal(153)         1
Fatal(135)         1
Fatal(88)          1
Fatal(31)          1
Fatal(174)         1
Fatal(68)          1
Fatal(15)          1
Fatal(37)          1
Fatal(20)          1
Fatal(132)         1
Fatal(156)         1
Fatal(230)         1
Fatal(131)         1
Fatal(144)         1
Fatal(111)         1
Fatal(43)          1
Fatal(256)         1
Fatal(28)          1
Fatal(78)          1
Fatal(47)          1
Name: Injury Severity, dtype: int64
```

In [21]:
```
df_relevant["Make"].str.lower().str.title()
```

Out[21]:
```
0                   Stinson
1                     Piper
3                  Rockwell
6                   Cessna
7                   Cessna
               ...
88639               Cessna
88647               Cessna
88661                Beech
88735    Stephen J Hoffman
88767             Luscombe
Name: Make, Length: 70667, dtype: object
```

In [22]:
```
#cleaning location
df_relevant["Location"].str.lower().str.title()
```

Out[22]:
```
0        Moose Creek, Id
1         Bridgeport, Ca
3             Eureka, Ca
6            Cotton, Mn
7           Pullman, Wa
              ...
88639           Iola, Tx
88647         Dacula, Ga
88661        Ardmore, Ok
88735        Houston, Tx
88767     Bridgeport, Tx
Name: Location, Length: 70667, dtype: object
```

In [23]:
```
#checking for the values in the relevant column
df_relevant["Purpose Of Flight"].unique()
df_relevant
```

Out[23]:

	Country	Injury Severity	Location	Aircraft Damage	Make	Model	Amateur Built	Nu Eng
0	United States	Fatal(2)	MOOSE CREEK, ID	Destroyed	Stinson	108-3	No	
1	United States	Fatal(4)	BRIDGEPORT, CA	Destroyed	Piper	PA24-180	No	
3	United States	Fatal(2)	EUREKA, CA	Destroyed	Rockwell	112	No	
6	United States	Fatal(4)	COTTON, MN	Destroyed	Cessna	180	No	
7	United States	Non-Fatal	PULLMAN, WA	Substantial	Cessna	140	No	
...	...	...	...	...	...	...	...	
88639	United States	Non-Fatal	Iola, TX	Substantial	CESSNA	150	No	
88647	United States	Non-Fatal	Dacula, GA	Substantial	CESSNA	177RG	No	
88661	United States	Non-Fatal	Ardmore, OK	Substantial	BEECH	B-60	No	
88735	United States	Minor	Houston, TX	Substantial	STEPHEN J HOFFMAN	MS-500	Yes	
88767	United States	Non-Fatal	Bridgeport, TX	Substantial	LUSCOMBE	8E	No	

70667 rows × 15 columns

In [24]:

```
#dropping columns with the value unknown in th purpose of flight column
df_relevant=df_relevant.drop(df_relevant[df_relevant["Purpose Of Flight"]
```

In [25]:

```
df_relevant["Purpose Of Flight"].value_counts()
```

Out[25]:
```
Personal                         42755
Instructional                     9497
Aerial Application                4254
Business                          3536
Positioning                       1328
Other Work Use                     954
Ferry                              715
Aerial Observation                 619
Public Aircraft                    573
Executive/corporate                456
Flight Test                        270
Skydiving                          126
External Load                       83
Banner Tow                          81
Public Aircraft - Federal           75
Public Aircraft - Local             64
Air Race show                       57
Public Aircraft - State             54
Glider Tow                          34
Firefighting                        18
Air Race/show                       15
Air Drop                             7
PUBS                                 2
ASHO                                 2
PUBL                                 1
Name: Purpose Of Flight, dtype: int64
```

In [26]:
```
#checking for the Weather condition column
df_relevant["Weather Condition"].value_counts()
```

Out[26]:
```
VMC    61096
IMC     3954
UNK      462
Unk       64
Name: Weather Condition, dtype: int64
```

In [27]:
```
df_relevant["Broad Phase Of Flight"]
```

Out[27]:
```
0         Cruise
1        Unknown
3         Cruise
6        Unknown
7        Takeoff
          ...
88639    Landing
88647    Landing
88661    Landing
88735    Landing
88767    Landing
Name: Broad Phase Of Flight, Length: 65576, dtype: object
```

In [28]:
```
#dropping all columns with unknown
df_relevant=df_relevant.drop(df_relevant[df_relevant["Broad Phase Of Fligl
```

In [29]:
```
df_relevant["Broad Phase Of Flight"].value_counts()
```

```
Out[29]: Landing        30385
         Takeoff         9866
         Cruise          7959
         Maneuvering     6254
         Approach        4847
         Climb           1414
         Taxi            1401
         Descent         1323
         Go-around       1142
         Standing         515
         Other             78
         Name: Broad Phase Of Flight, dtype: int64
```

```
In [30]: df_relevant.shape
```

```
Out[30]: (65184, 15)
```

```
In [31]: #checking for duplicates
         duplicates=df_relevant.duplicated()
         duplicates
         #our data does not have duplicate rows.
```

```
Out[31]: 0        False
         3        False
         7        False
         8        False
         9        False
                  ...
         88639    False
         88647    False
         88661    False
         88735    False
         88767    False
         Length: 65184, dtype: bool
```

our cleaned data contains 65184 rows and 15 columns

Creating a csv file for the cleaned data

```
In [32]: df_relevant.to_csv('cleaned2_AviationData.csv', index=False)
```

```
In [33]: #Loading the cleaned data
         df_cleaned=pd.read_csv("cleaned2_AviationData.csv")
         df_cleaned
```

Out[33]:

	Country	Injury Severity	Location	Aircraft Damage	Make	Model	Amateur Built	Nu En
0	United States	Fatal(2)	MOOSE CREEK, ID	Destroyed	Stinson	108-3	No	
1	United States	Fatal(2)	EUREKA, CA	Destroyed	Rockwell	112	No	
2	United States	Non-Fatal	PULLMAN, WA	Substantial	Cessna	140	No	
3	United States	Non-Fatal	EAST HANOVER, NJ	Substantial	Cessna	401B	No	
4	United States	Non-Fatal	JACKSONVILLE, FL	Substantial	North American	NAVION L-17B	No	
...	...	...	...	...	...	...	...	
65179	United States	Non-Fatal	Iola, TX	Substantial	CESSNA	150	No	
65180	United States	Non-Fatal	Dacula, GA	Substantial	CESSNA	177RG	No	
65181	United States	Non-Fatal	Ardmore, OK	Substantial	BEECH	B-60	No	
65182	United States	Minor	Houston, TX	Substantial	STEPHEN J HOFFMAN	MS-500	Yes	
65183	United States	Non-Fatal	Bridgeport, TX	Substantial	LUSCOMBE	8E	No	

65184 rows × 15 columns

In [34]:
```
df_cleaned.shape
```

Out[34]:
```
(65184, 15)
```

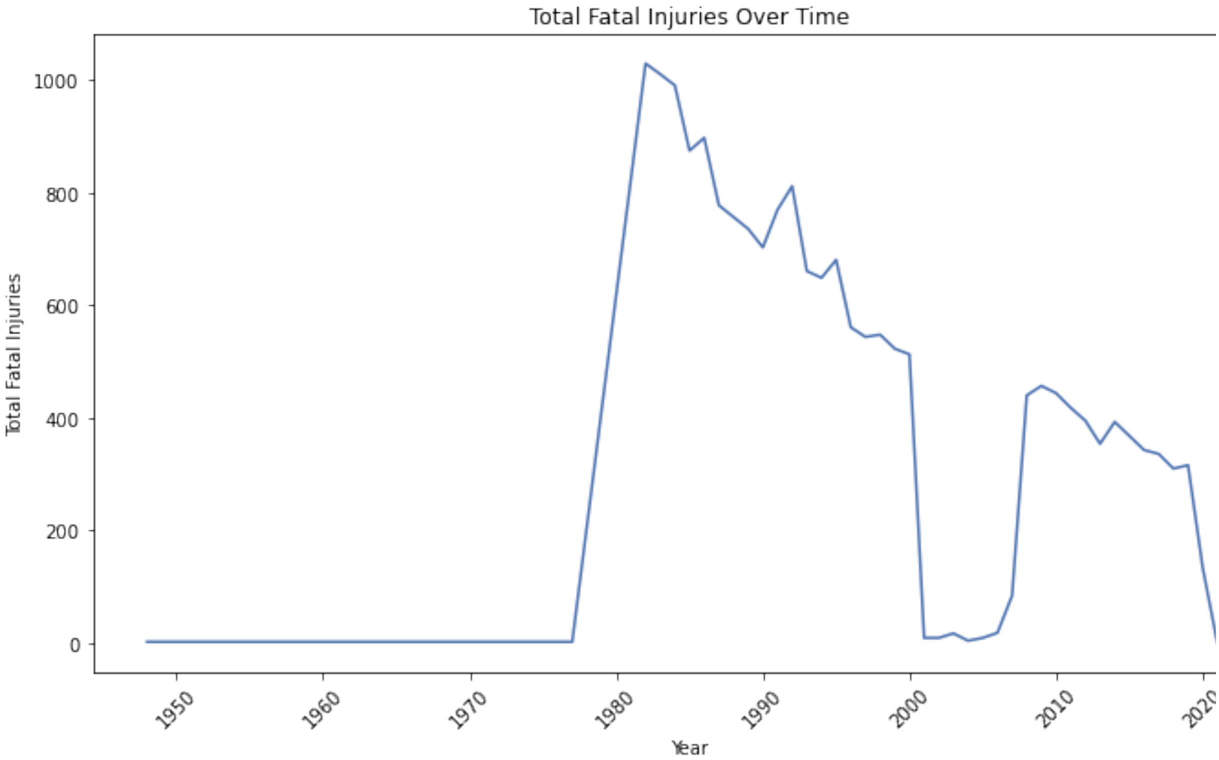
2.1.2 Data Visualization

In [35]:
```
#import the relevant libraries
import matplotlib.pyplot as plt
import seaborn as sns

# Set up the plotting environment
sns.set_palette("deep")
```

Distribution of accidents from 1962 to 2023 and injury severity

Is it safe to invest in the aviation industry?How many accidents have been seen over the years a fatal were they?

```python
In [36]: # Plot 3: Number of Fatal Injuries by Year
plt.figure(figsize=(10, 6))
sns.lineplot(data=df_cleaned.groupby('Year')['Total Fatal Injuries'].sum(
plt.title('Total Fatal Injuries Over Time')
plt.xlabel('Year')
plt.ylabel('Total Fatal Injuries')
plt.xticks(rotation=45)  # Rotate x labels
plt.tight_layout()
plt.show()
```

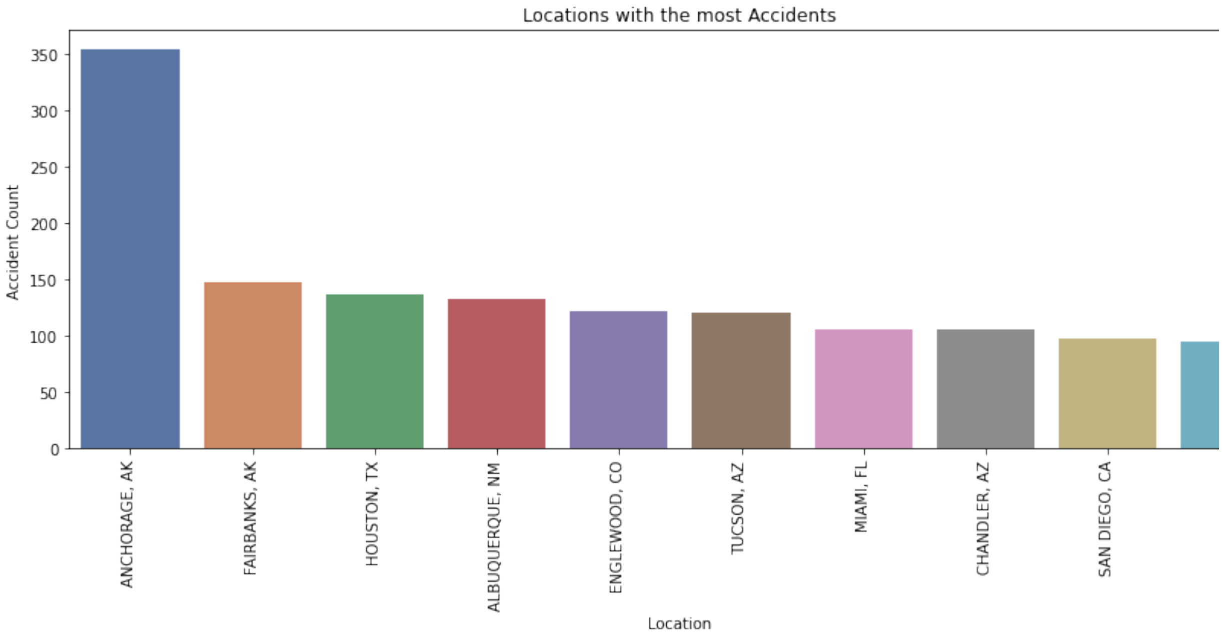


The overall trend suggests significant progress in aviation safety, with fatal injuries becoming rar
safety measures evolved.It is however notted that despite the decline trend in the number of acci
other factors such as travel restrictions affect the aviation indusrty.This is seen by a sharp decline
number of fatal accidents 2020 due to travel bans caused by covid 19.

Accidents location

top ten locations with most accidents.

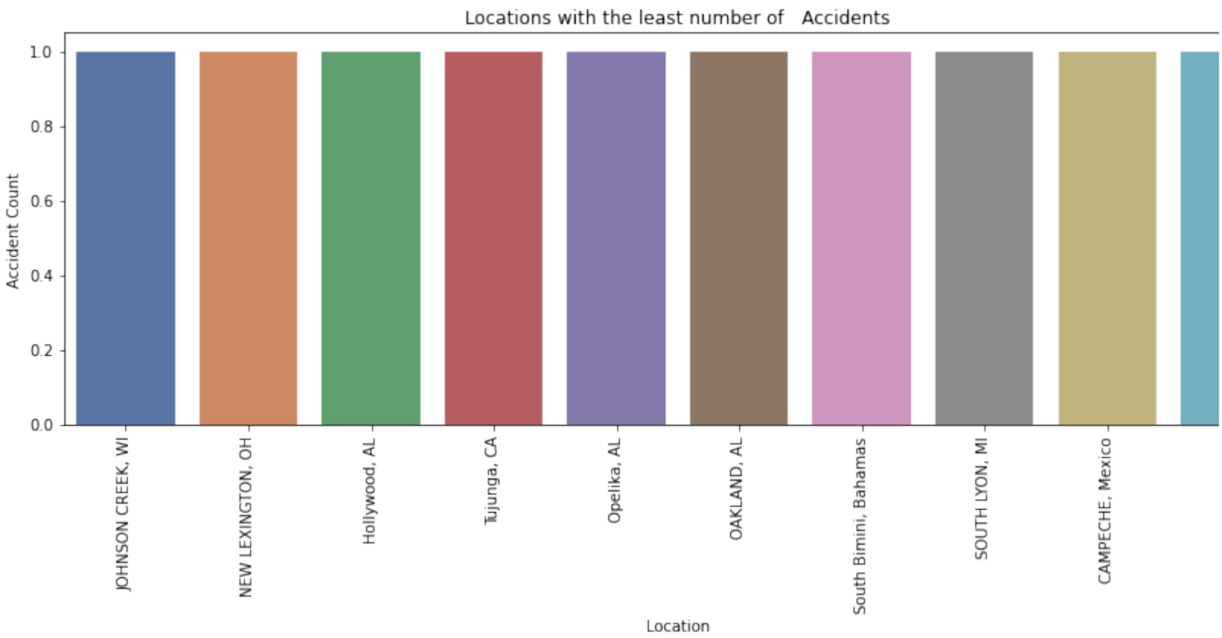
```
In [37]: #Plot accidents distribution by location
plt.figure(figsize=(12, 6))

# Get the top 10 most frequent locations for accidents
locations_with_most_accidents = df_cleaned['Location'].value_counts().nla

# Plot the top 10 locations
sns.barplot(x=locations_with_most_accidents .index, y=locations_with_most_
plt.title('Locations with the most Accidents')
plt.xlabel('Location')
plt.ylabel('Accident Count')
plt.xticks(rotation=90)
plt.tight_layout()
plt.show()
```



Anchorage, Alaska has an accident count exceeding 350, far surpassing other locations.This car attributed to large air traffic as air traffic has been seen to be directly proportional to the number c accidents.

locations with least accidents

```
In [38]: plt.figure(figsize=(12, 6))

# Get the top 10 most frequent locations for accidents
location_with_least_accidents = df_cleaned['Location'].value_counts().nsma

# Plot the top 10 locations
sns.barplot(x=location_with_least_accidents.index, y=location_with_least_a
plt.title('Locations with the least number of   Accidents')
plt.xlabel('Location')
plt.ylabel('Accident Count')
plt.xticks(rotation=90)
plt.tight_layout()
plt.show()
```

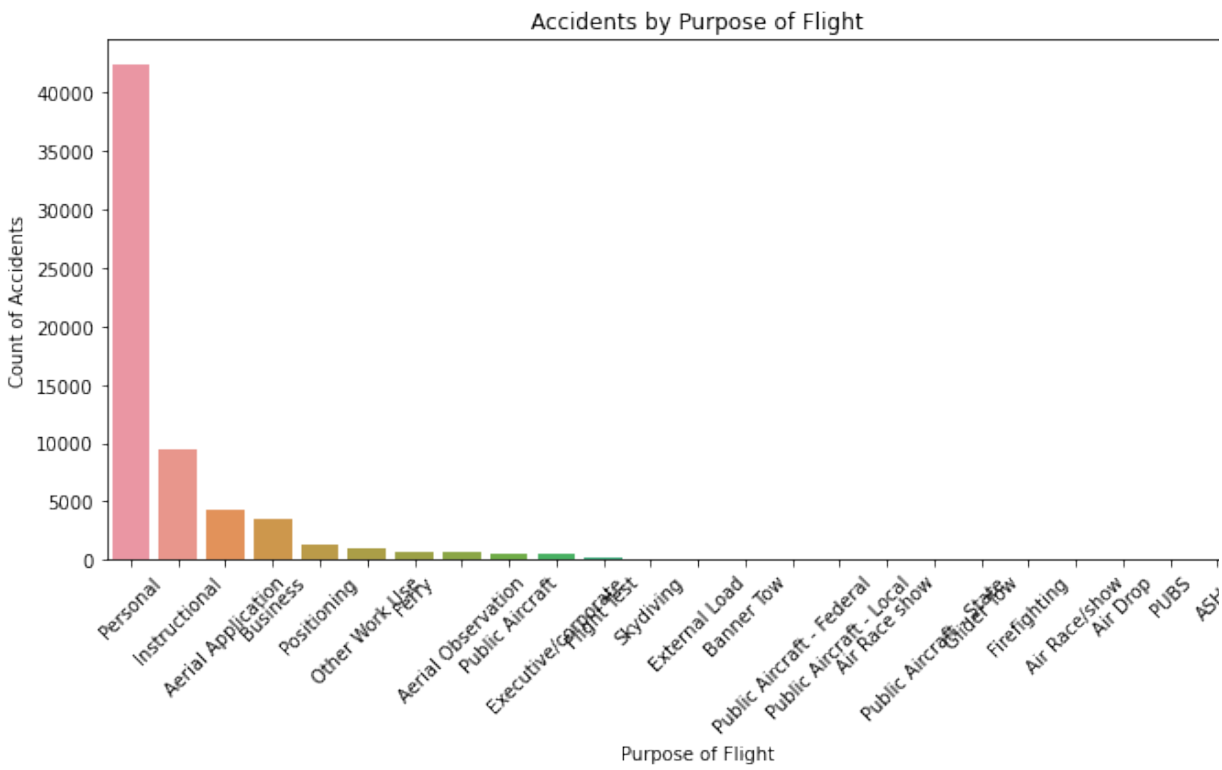


Above are the areas with the least number of incidents.This may signal a good opportunity for inv in terms of safety.It may also indicate good weather condition,low air traffic or advancement in technology.Good weather and advancement in technology offers promising opportunity for invest traffic however directly correlates with the economic activity of an area and low air traffic may ind economic activities.

purpose of flight

what type of flights by purpose are mostly involved in accidents

In [39]:
```
plt.figure(figsize=(10, 6))
sns.countplot(data=df_cleaned, x='Purpose Of Flight', order=df_cleaned['Pu
plt.title('Accidents by Purpose of Flight')
plt.xlabel('Purpose of Flight')
plt.ylabel('Count of Accidents')
plt.xticks(rotation=45)  # Rotate x labels
plt.tight_layout()
plt.show()
```

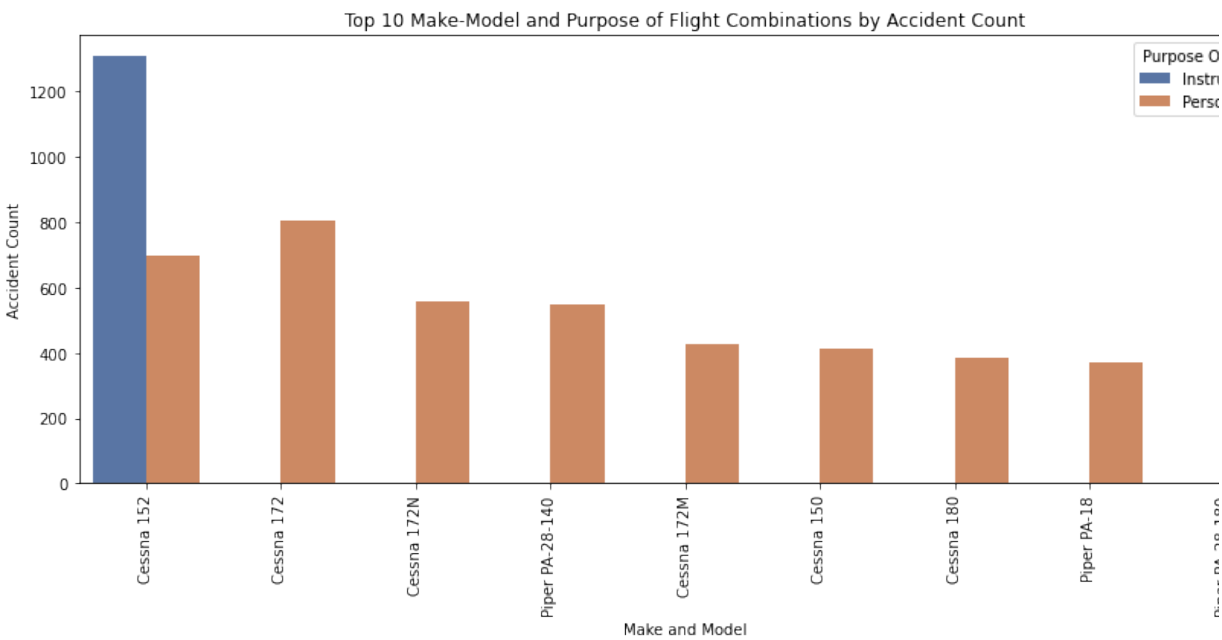


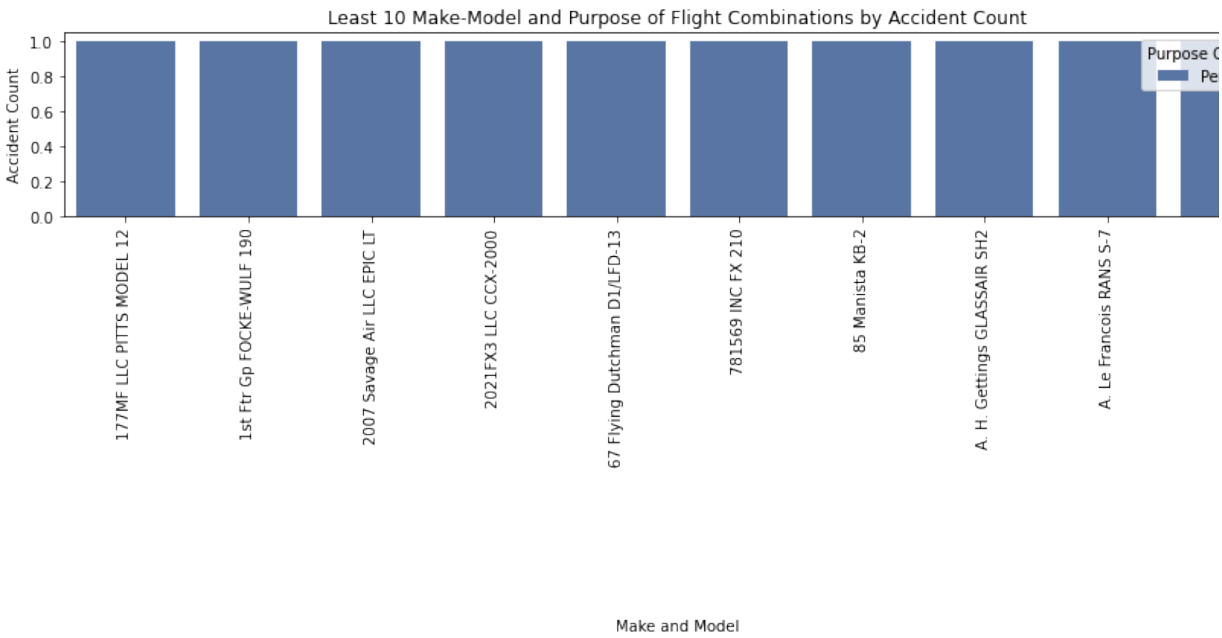
The "Personal" category is overwhelmingly the largest, with over 40,000 accidents. It suggests th personal flights account for a significant majority of all accidents in the dataset.The second-highe category is "Instructional", with fewer than 10,000 accidents, followed by "Aerial Application", whi similar count. This indicates that accidents occur relatively frequently in training and agricultural s or similar tasks. Business and Positioning flights contribute a smaller but noticeable share of acc

Make-model and purpose of flight

what is the distribution of accidents based on the make,model and purpose of the aircraft. Which safest make and model based on the purpose of flight to invest in.

In [40]:
```
# Group by 'Make', 'Model', and 'Purpose of Flight' to count accidents
df_cleaned['Make_Model'] = df_cleaned['Make'] + ' ' + df_cleaned['Model']
make_model_purpose_counts = df_cleaned.groupby(['Make_Model', 'Purpose Of

# Top 10 combinations
top_10_make_model_purpose = make_model_purpose_counts.nlargest(10, 'Accid

# Least 10 combinations
least_10_make_model_purpose = make_model_purpose_counts.nsmallest(10, 'Ac

# Plot the Top 10 Make-Model and Purpose of Flight combinations
plt.figure(figsize=(12, 6))
sns.barplot(data=top_10_make_model_purpose, x='Make_Model', y='Accident_C
plt.title('Top 10 Make-Model and Purpose of Flight Combinations by Accide
plt.xlabel('Make and Model')
plt.ylabel('Accident Count')
plt.xticks(rotation=90)
plt.tight_layout()
plt.show()

# Plot the Least 10 Make-Model and Purpose of Flight combinations
plt.figure(figsize=(12, 6))
sns.barplot(data=least_10_make_model_purpose, x='Make_Model', y='Accident_
plt.title('Least 10 Make-Model and Purpose of Flight Combinations by Acci
plt.xlabel('Make and Model')
plt.ylabel('Accident Count')
plt.xticks(rotation=90)
plt.tight_layout()
plt.show()
```





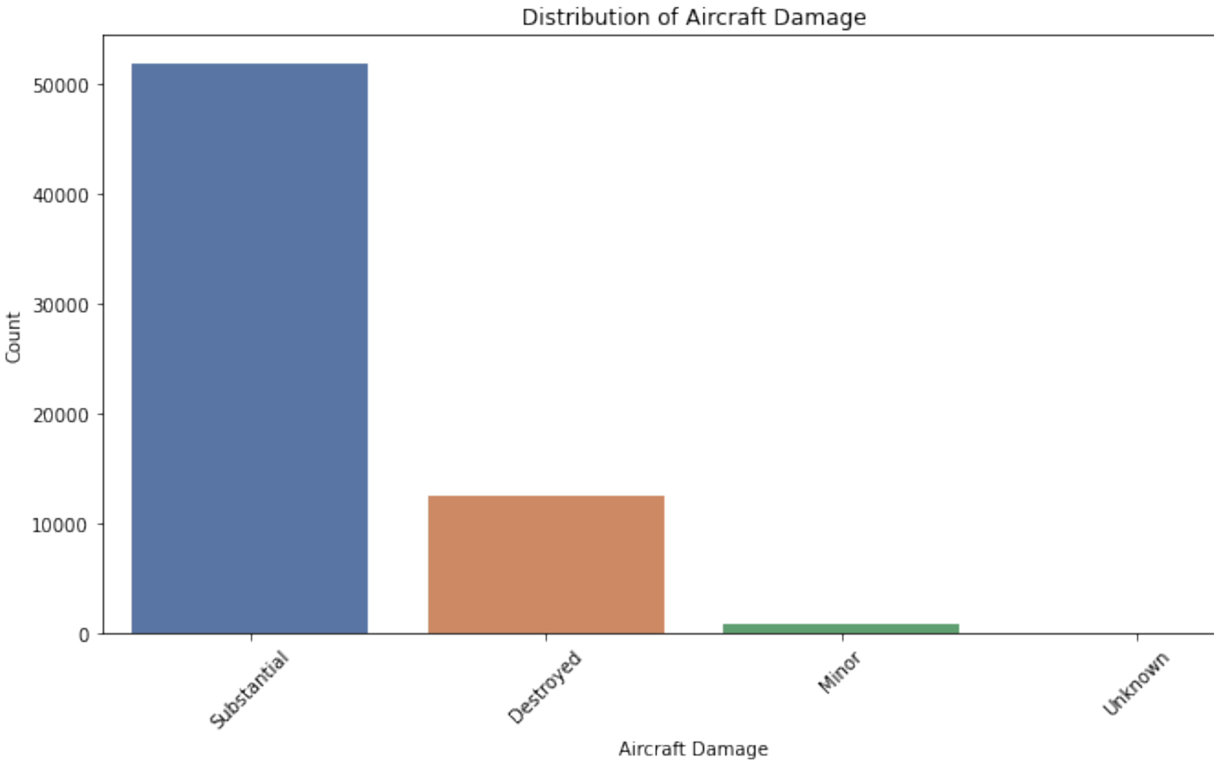
Cessna 152 used for instructional purpose is the most prone to accidents.Cessna aircrafts used
personal travels dominates the list of the ten most aircrafts with accidents followed by Piper.

Aircraft Damage

how damaged are the aircrafts incase of an accident?

In [41]:

```
plt.figure(figsize=(10, 6))
sns.countplot(data=df_cleaned, x='Aircraft Damage', order=df_cleaned['Air
plt.title('Distribution of Aircraft Damage')
plt.xlabel('Aircraft Damage')
plt.ylabel('Count')
plt.xticks(rotation=45)  # Rotate x labels
plt.tight_layout()
plt.show()
```

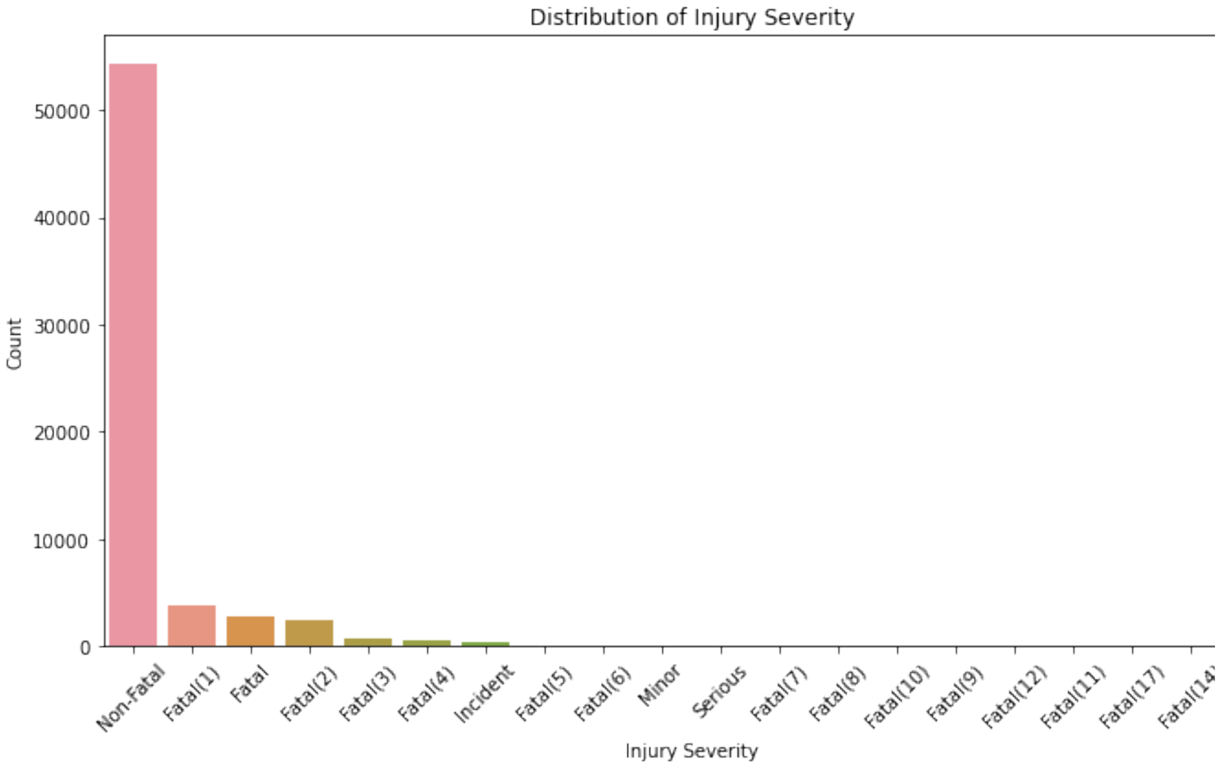


The chart is dominated by the "Substantial" category, with over 50,000 instances, indicating most accidents result in significant but repairable aircraft damage. The second most common outcome "Destroyed," with around 10,000–15,000 incidents, representing cases where aircraft were irrepa damaged. "Minor" damage is rare, showing only a small number of accidents with superficial or e fixable damage.

Distribution of Injury Severity

how fatal were the injuries?

In [42]:
```
# Plot 1: Distribution of Injury Severity
plt.figure(figsize=(10, 6))
sns.countplot(data=df_cleaned, x='Injury Severity', order=df_cleaned['Inj
plt.title('Distribution of Injury Severity')
plt.xlabel('Injury Severity')
plt.ylabel('Count')
plt.xticks(rotation=45)  # Rotate x labels
plt.tight_layout()
plt.show()
```

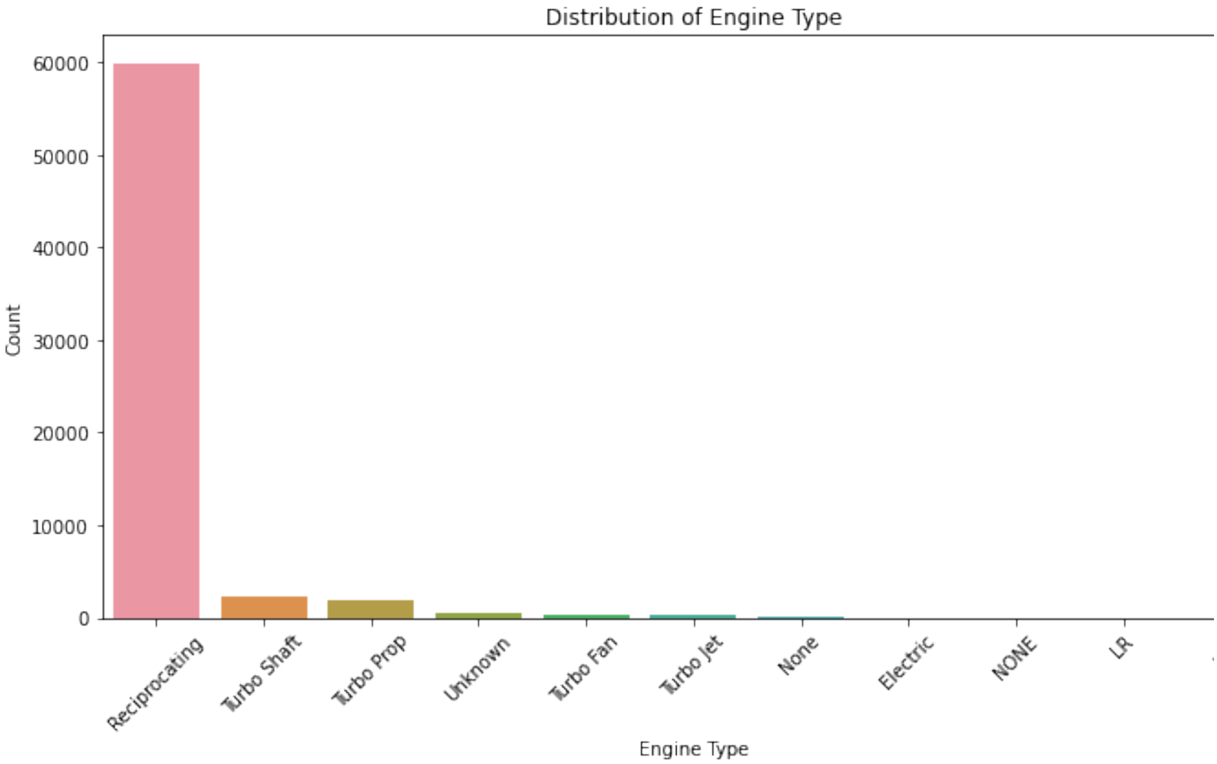


Most injuries by severity are non-fatal.

Distribution of Engine Type

which engine types are most and least involved in accidents?

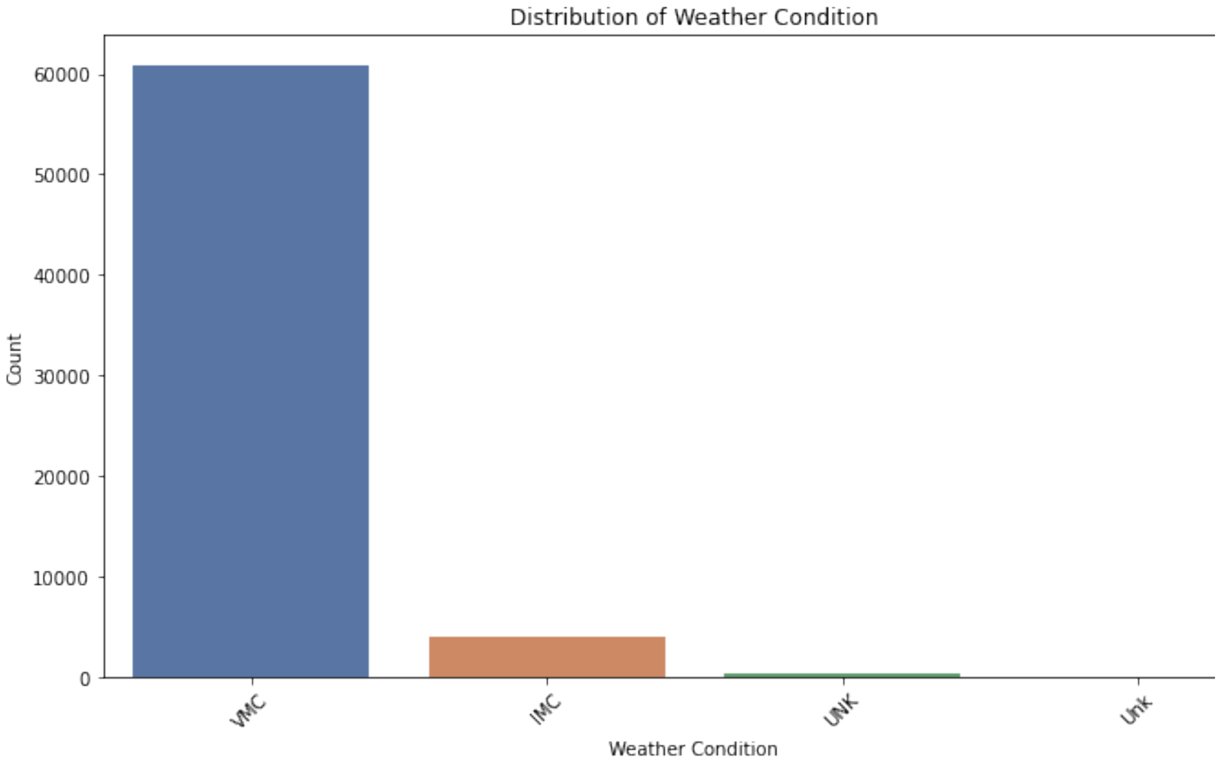
In [47]:
```
plt.figure(figsize=(10, 6))
sns.countplot(data=df_cleaned, x='Engine Type', order=df_cleaned['Engine
plt.title('Distribution of Engine Type')
plt.xlabel('Engine Type')
plt.ylabel('Count')
plt.xticks(rotation=45)  # Rotate x labels
plt.tight_layout()
plt.show()
```



Reciprocating engines are the most prone to accidents.Reciprocating engines are used for small such as Cessna .The small aircrafts areused for personal purposes.This explains why Cessna ha highest number of accidents. Turbo jet engines are used in commercial and business aircracts ar low number of accidents.

Distribution of Accidents by Weather Condition

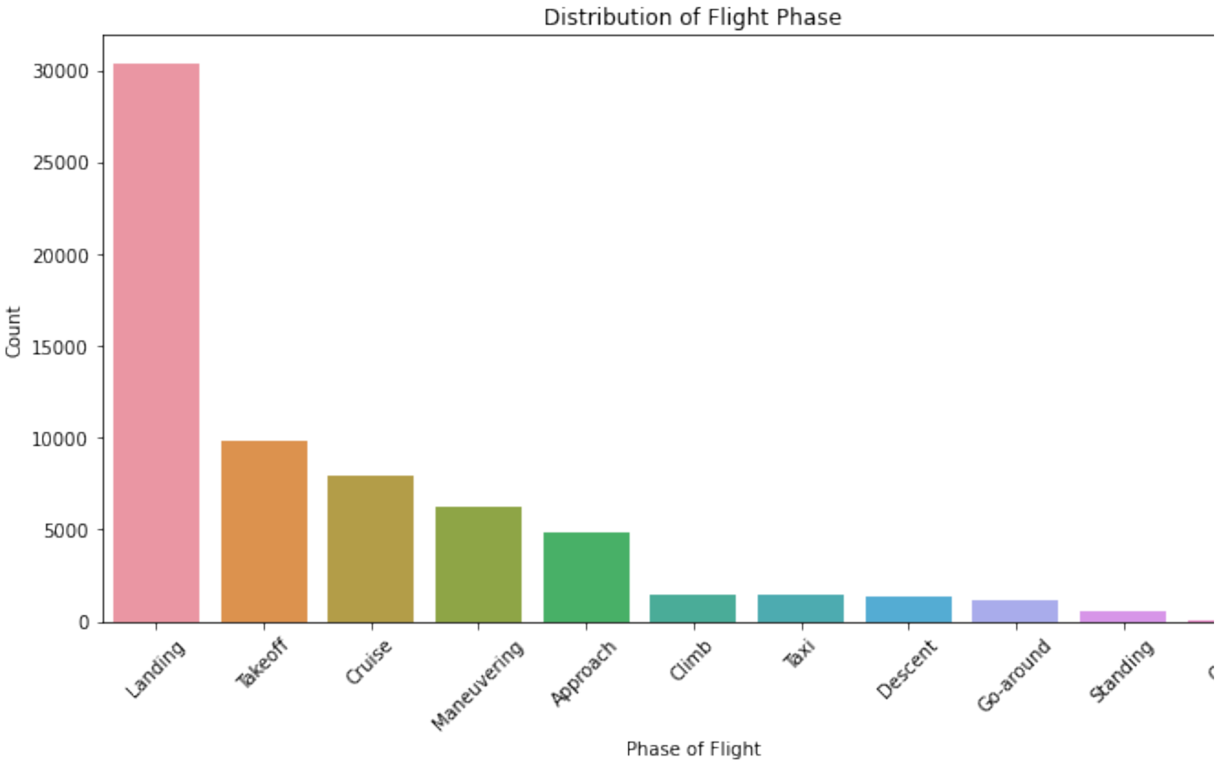
In [44]:
```
plt.figure(figsize=(10, 6))
sns.countplot(data=df_cleaned, x='Weather Condition', order=df_cleaned['We
plt.title('Distribution of Weather Condition')
plt.xlabel('Weather Condition')
plt.ylabel('Count')
plt.xticks(rotation=45)  # Rotate x labels
plt.tight_layout()
plt.show()
```



Weather condition directly affects the number of accidents.This is confirmed by our analysis of th location where Alaska has the most accidents and experiences bad weather.

Distribution of Broad Phase of Flight

In [45]:
```
plt.figure(figsize=(10, 6))
sns.countplot(data=df_cleaned, x='Broad Phase Of Flight', order=df_cleane
plt.title('Distribution of Flight Phase')
plt.xlabel('Phase of Flight')
plt.ylabel('Count')
plt.xticks(rotation=45)  # Rotate x labels
plt.tight_layout()
plt.show()
```



The above plot shows that most accidents occur during landing and take off followed by cruise ar maneuvering.The high number of accidents during landing may be attributed to several factors a them proximity to obstacles,decision making pressure,environmental conditions among other fac

Conclusion

The analysis has given us insights into the risk and opportunities in the aviation industry.Based o analysis of the accidents i would recommend that Datan Africa invest in the aviation industry.

```
In [ ]:
```