Final Project Submission

Please fill out:

- Student name: Alice Kimani
- Student pace: Part time
- Scheduled project review date/time:
- Instructor name: Fidelis Wanalwenge

Your company is expanding in to new industries to diversify its portfolio. Specifically, they are interested in purchasing and operating airplanes for commercial and private enterprises, but do not know anything about the potential risks of aircraft. You are charged with determining which aircraft are the lowest risk for the company to start this new business endeavor. You must then translate your findings into actionable insights that the head of the new aviation division can use to help decide which aircraft to purchase.

In [4]:
```
#create the environment for analysis
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
```

In [5]:
```
#import the data that we will be working on
df = pd.read_csv('data/Aviation_Data.csv', low_memory= False)
```

In [6]:
```
#data preview
df.head()
```

Out[6]:

	Event.Id	Investigation.Type	Accident.Number	Event.Date	Location	C
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	

5 rows × 31 columns

In [7]:
```
#Checking the structure of our data, number of rows vs Number of columns
df.shape
```

Out[7]: (90348, 31)

DATA CLEANING

In [8]:
```
#Investigating the variables in the column header
df.columns
```

Out[8]:
```
Index(['Event.Id', 'Investigation.Type', 'Accident.Number', 'Event.Dat
e',
       '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.Descrip
tion',
       'Schedule', 'Purpose.of.flight', 'Air.carrier', 'Total.Fatal.Inju
ries',
       'Total.Serious.Injuries', 'Total.Minor.Injuries', 'Total.Uninjure
d',
       'Weather.Condition', 'Broad.phase.of.flight', 'Report.Status',
       'Publication.Date'],
      dtype='object')
```

In [9]:
```
#Investigates data type and null values in the data frame
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 90348 entries, 0 to 90347
Data columns (total 31 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Event.Id                88889 non-null  object
 1   Investigation.Type      90348 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            50132 non-null  object
 9   Airport.Name            52704 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     87507 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             81793 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           82505 non-null  object
 30  Publication.Date        73659 non-null  object
dtypes: float64(5), object(26)
memory usage: 21.4+ MB
```

In [10]:

```
#showcasing the number of missing values in each column variable
df.isna().sum()
```

Out[10]:
```
Event.Id                   1459
Investigation.Type            0
Accident.Number            1459
Event.Date                 1459
Location                   1511
Country                    1685
Latitude                  55966
Longitude                 55975
Airport.Code              40216
Airport.Name              37644
Injury.Severity            2459
Aircraft.damage            4653
Aircraft.Category         58061
Registration.Number        2841
Make                       1522
Model                      1551
Amateur.Built              1561
Number.of.Engines          7543
Engine.Type                8555
FAR.Description           58325
Schedule                  77766
Purpose.of.flight          7651
Air.carrier               73700
Total.Fatal.Injuries      12860
Total.Serious.Injuries    13969
Total.Minor.Injuries      13392
Total.Uninjured            7371
Weather.Condition          5951
Broad.phase.of.flight     28624
Report.Status              7843
Publication.Date          16689
dtype: int64
```

In [11]:
```
#Listing the specific columns with missing values
missing_cols = df.columns[df.isnull().any()]
print(missing_cols)
```

```
Index(['Event.Id', '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.Uninjure
d',
       'Weather.Condition', 'Broad.phase.of.flight', 'Report.Status',
       'Publication.Date'],
      dtype='object')
```

The expectation is that the aircrafts will be for both private and commercial use. We would like to know the potential risk of the airplanes, Lowest risk for the company to start the business and determining which aircraft to purchase. With this in mind we need to narrow down our columns to only those that will enable us analyse our data objectively.

IMPORTANT COLUMNS Event.Date-Helps us identify safety trends overtime Injury.Severity-Key indicator of the magnitude of injury Aircraft.damage-Shows the extent of physical impact to the aircraft after an accident Aircraft.Category-

Distinguishes aircraft types(airplane, helicopter etc)-we need to figure out the risk profile for different categories Make-Helps to compare different manufactures interms of design, training, maintenance and other factors Model-Pinpoints specific aircraft models with higher/lower incident rates NUmber.of.Engines-More engines can mean more safety or even redundancy Engine.Type-Different types of Engines have different performance and safety records Purpose.of.flight- Distinguishes the prpose of flight either commercial, private etc Total.Fatal Injuries-Measures the worst outcome(best for identifying high risk aircraft) Total.Serious.Injuries-Quantifies accident severity without being fatal, indicates significant safety risk Total.Minor.Injuries-Useful to identify aircrafts involved in lower severity incidents otal.Uninjured- Shows how often passengers survived ncidents unhurt.(airraft resilience) Weather.Condition- Helps to show how external factors affect aircraft operations

In [12]:
```
#Narrowing down our data to only the columns that will enable us make the
important_columns = [
    'Event.Date',
    'Injury.Severity',
    'Aircraft.damage',
    'Aircraft.Category',
    'Make',
    'Model',
    'Number.of.Engines',
    'Engine.Type',
    'Purpose.of.flight',
    'Total.Fatal.Injuries',
    'Total.Serious.Injuries',
    'Total.Minor.Injuries',
    'Total.Uninjured',
    'Weather.Condition',
]

df_filtered = df[important_columns]
```

We want to have a glimpse of the parameters in our important_columns, just to have an idea of the breakdown of the cells

In [13]:
```
#obtaining an outlook of our data we run the print
print(df_filtered)
```

```
      Event.Date Injury.Severity Aircraft.damage Aircraft.Category  \
0      1948-10-24        Fatal(2)       Destroyed               NaN
1      1962-07-19        Fatal(4)       Destroyed               NaN
2      1974-08-30        Fatal(3)       Destroyed               NaN
3      1977-06-19        Fatal(2)       Destroyed               NaN
4      1979-08-02        Fatal(1)       Destroyed               NaN
...          ...             ...             ...               ...
90343  2022-12-26           Minor             NaN               NaN
90344  2022-12-26             NaN             NaN               NaN
90345  2022-12-26       Non-Fatal     Substantial          Airplane
90346  2022-12-26             NaN             NaN               NaN
90347  2022-12-29           Minor             NaN               NaN

                            Make      Model  Number.of.Engines  \
0                        Stinson      108-3                1.0
1                          Piper   PA24-180                1.0
2                         Cessna       172M                1.0
3                       Rockwell        112                1.0
4                         Cessna        501                NaN
...                          ...        ...                ...
90343                      PIPER  PA-28-151                NaN
90344                   BELLANCA       7ECA                NaN
90345  AMERICAN CHAMPION AIRCRAFT      8GCBC                1.0
90346                     CESSNA       210N                NaN
90347                      PIPER  PA-24-260                NaN

         Engine.Type Purpose.of.flight  Total.Fatal.Injuries  \
0      Reciprocating          Personal                   2.0
1      Reciprocating          Personal                   4.0
2      Reciprocating          Personal                   3.0
3      Reciprocating          Personal                   2.0
4                NaN          Personal                   1.0
...              ...               ...                   ...
90343            NaN          Personal                   0.0
90344            NaN               NaN                   0.0
90345            NaN          Personal                   0.0
90346            NaN          Personal                   0.0
90347            NaN          Personal                   0.0

       Total.Serious.Injuries  Total.Minor.Injuries  Total.Uninjured  \
0                         0.0                   0.0              0.0
1                         0.0                   0.0              0.0
2                         NaN                   NaN              NaN
3                         0.0                   0.0              0.0
4                         2.0                   NaN              0.0
...                       ...                   ...              ...
90343                     1.0                   0.0              0.0
90344                     0.0                   0.0              0.0
90345                     0.0                   0.0              1.0
90346                     0.0                   0.0              0.0
90347                     1.0                   0.0              1.0

     Weather.Condition
0                  UNK
1                  UNK
2                  IMC
3                  IMC
4                  VMC
...                ...
90343              NaN
```

```
90344              NaN
90345              VMC
90346              NaN
90347              NaN

[90348 rows x 14 columns]
```

Inorder to clean our important_columns we need to know the representation of null values, this will help up know which method of filling is convinient for the different data types.

In [14]:
```
#obtaining an outlook of our data we run the print

print(df_filtered)
```

```
      Event.Date Injury.Severity Aircraft.damage Aircraft.Category  \
0      1948-10-24        Fatal(2)       Destroyed               NaN
1      1962-07-19        Fatal(4)       Destroyed               NaN
2      1974-08-30        Fatal(3)       Destroyed               NaN
3      1977-06-19        Fatal(2)       Destroyed               NaN
4      1979-08-02        Fatal(1)       Destroyed               NaN
...           ...             ...             ...               ...
90343  2022-12-26           Minor             NaN               NaN
90344  2022-12-26             NaN             NaN               NaN
90345  2022-12-26       Non-Fatal     Substantial          Airplane
90346  2022-12-26             NaN             NaN               NaN
90347  2022-12-29           Minor             NaN               NaN

                          Make      Model  Number.of.Engines  \
0                      Stinson      108-3                1.0
1                        Piper   PA24-180                1.0
2                       Cessna       172M                1.0
3                     Rockwell        112                1.0
4                       Cessna        501                NaN
...                        ...        ...                ...
90343                    PIPER  PA-28-151                NaN
90344                 BELLANCA       7ECA                NaN
90345  AMERICAN CHAMPION AIRCRAFT    8GCBC                1.0
90346                   CESSNA       210N                NaN
90347                    PIPER  PA-24-260                NaN

         Engine.Type Purpose.of.flight  Total.Fatal.Injuries  \
0      Reciprocating          Personal                   2.0
1      Reciprocating          Personal                   4.0
2      Reciprocating          Personal                   3.0
3      Reciprocating          Personal                   2.0
4                NaN          Personal                   1.0
...              ...               ...                   ...
90343            NaN          Personal                   0.0
90344            NaN               NaN                   0.0
90345            NaN          Personal                   0.0
90346            NaN          Personal                   0.0
90347            NaN          Personal                   0.0

       Total.Serious.Injuries  Total.Minor.Injuries  Total.Uninjured  \
0                         0.0                   0.0              0.0
1                         0.0                   0.0              0.0
2                         NaN                   NaN              NaN
3                         0.0                   0.0              0.0
4                         2.0                   NaN              0.0
...                       ...                   ...              ...
90343                     1.0                   0.0              0.0
90344                     0.0                   0.0              0.0
90345                     0.0                   0.0              1.0
90346                     0.0                   0.0              0.0
90347                     1.0                   0.0              1.0

      Weather.Condition
0                   UNK
1                   UNK
2                   IMC
3                   IMC
4                   VMC
...                 ...
90343               NaN
```

```
90344              NaN
90345              VMC
90346              NaN
90347              NaN

[90348 rows x 14 columns]
```

Depending on whether the data type we use different methods of filing

In [15]:
```
#data cleaning by filling the missing values
df_filtered = df[important_columns].copy()

#for categorical columns we fill with 'Unknown'
categorical_cols = [
    'Injury.Severity','Aircraft.damage','Aircraft.Category','Make','Model
]
df_filtered.loc[:,categorical_cols]= df_filtered[categorical_cols].fillna

#Assuming missing numeric value in injury has not been reported we fill w
injury_cols = [
    'Total.Serious.Injuries','Total.Minor.Injuries','Total.Fatal.Injuries
]
df_filtered.loc[:,injury_cols] = df_filtered[injury_cols].fillna(0)

#Fill missing number of engines as 1
df_filtered['Number.of.Engines'] = df_filtered['Number.of.Engines'].filln

#Filling the missing values in our date
df_filtered['Event.Date'] = df_filtered['Event.Date'].fillna(pd.to_dateti
```

In [16]:
```
#Confirmation of success of filling missing values
df_filtered.isnull().sum()
```

Out[16]:
```
Event.Date                0
Injury.Severity           0
Aircraft.damage           0
Aircraft.Category         0
Make                      0
Model                     0
Number.of.Engines         0
Engine.Type               0
Purpose.of.flight         0
Total.Fatal.Injuries      0
Total.Serious.Injuries    0
Total.Minor.Injuries      0
Total.Uninjured           0
Weather.Condition         0
dtype: int64
```

ANALYSIS

Getting into specifics Calculate total injuries(fatal+serious+minor) Group by aircraft make/model and calculate Number of incidents Average injuries %with major damage damage in poor weather safety profile by aircraft

In [17]:
```
#Create a total injuries column
df_filtered['Total.Injuries'] = (
    df_filtered['Total.Fatal.Injuries']+
```

```
    df_filtered['Total.Serious.Injuries']+
    df_filtered['Total.Minor.Injuries']
)
```

In [18]:
```
#Group by make and model to calculate risk metrics
risk_df = df_filtered.groupby(['Make', 'Model']).agg(
    Total_Incidents=('Event.Date', 'count'),
    Avg_Injuries=('Total.Injuries', 'mean'),
    Major_Damage_Percentage=('Aircraft.damage', lambda x: (x == 'Destroye
    Bad_Weather_Percentage=('Weather.Condition',lambda x: (x =='Instrumen
).reset_index()
```

In [19]:
```
#Define a function to normalize a column between 0 and 1
def normalize(col):
    return (col - col.min()) / (col.max() - col.min())

#Apply normalization to each metric
risk_df['Injury_Score'] = normalize(risk_df['Avg_Injuries'])
risk_df['Damage_Score'] = normalize(risk_df['Major_Damage_Percentage'])
risk_df['Weather_Score'] = normalize(risk_df['Bad_Weather_Percentage'])

#Combine into a single Risk Score(lower is safer)
risk_df['Risk_Score'] = risk_df[['Injury_Score','Damage_Score','Weather_S
```

In [20]:
```
#Sort by Risk Score
safe_aircraft = risk_df.sort_values('Risk_Score').reset_index(drop=True)
```

In [21]:
```
#Display top 10 Safest aircraft (lowest risk score)
safe_aircraft[['Make','Model','Total_Incidents','Avg_Injuries','Major_Dam
```

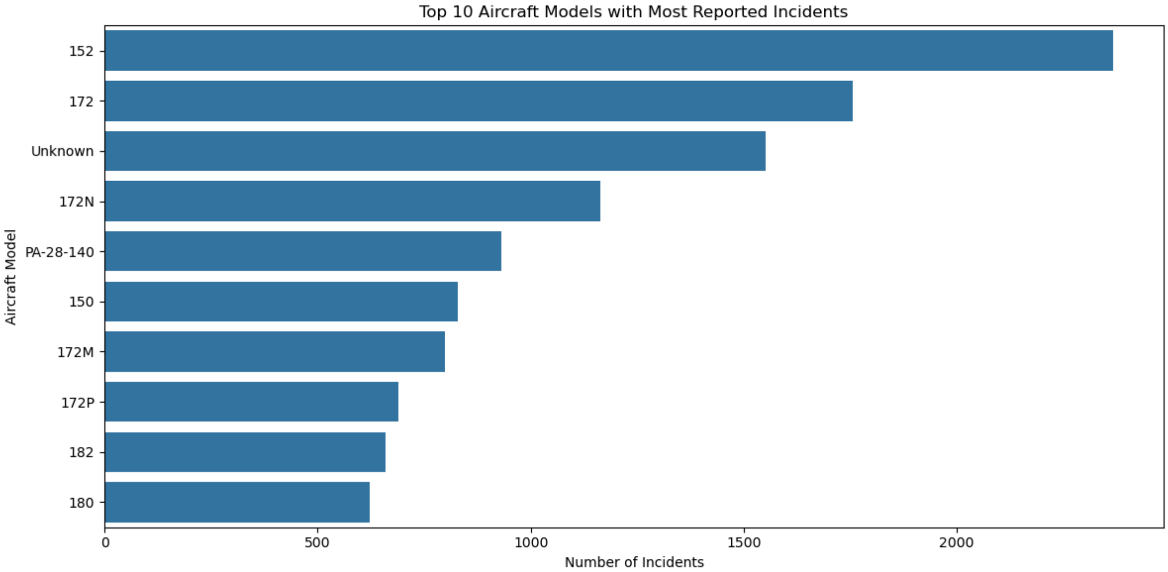
Out[21]:

	Make	Model	Total_Incidents	Avg_Injuries	Major_Damage_Percent
0	Macphee	RV6	1	0.0	
1	FISHER HAROLD R	320	1	0.0	
2	Manarin/johnson	Lancair IVP	1	0.0	
3	Malott	Mustang II	1	0.0	
4	MURRAY FRANK H	DA5B	1	0.0	
5	MULHOLLAND ROBERT A	VANS RV-7	1	0.0	
6	MUELLER MICHAEL WALTER	CHALLENGER II CW SPC	1	0.0	
7	MUDRY	CAP10	1	0.0	
8	MUDGE RAY	KIT FOX 5	1	0.0	
9	Byron/sorrell	SNS-2	1	0.0	

VISUALIZATION

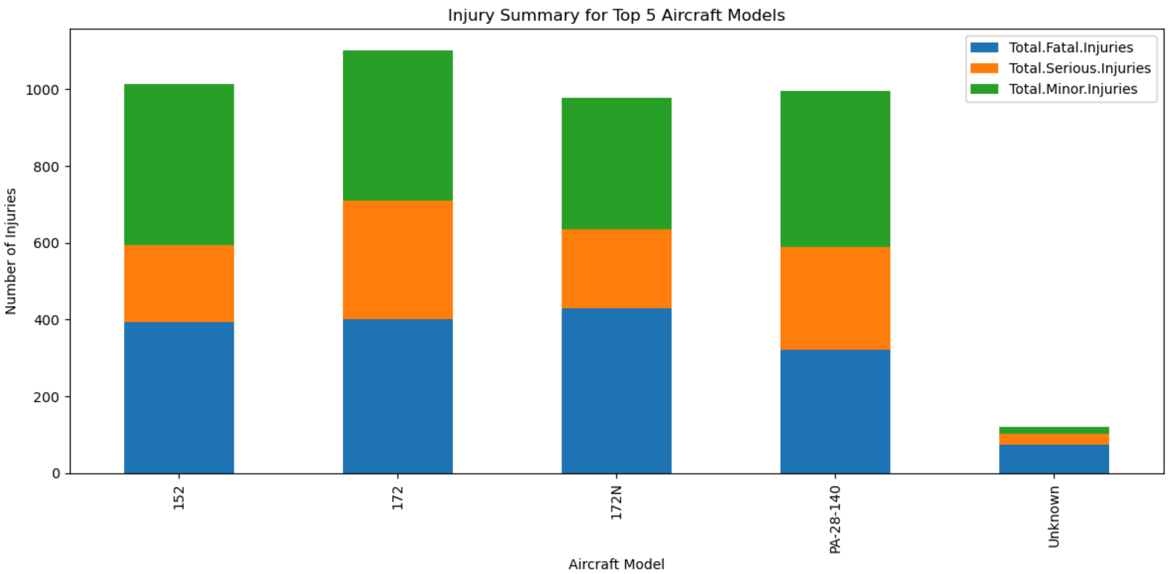
In [22]:
```
#Aircraft Models with Most Accidents
plt.figure(figsize=(12,6))
```

```
sns.countplot(data=df_filtered, y='Model', order=df_filtered['Model'].val
plt.title('Top 10 Aircraft Models with Most Reported Incidents')
plt.xlabel('Number of Incidents')
plt.ylabel('Aircraft Model')
plt.tight_layout()
plt.show()
```



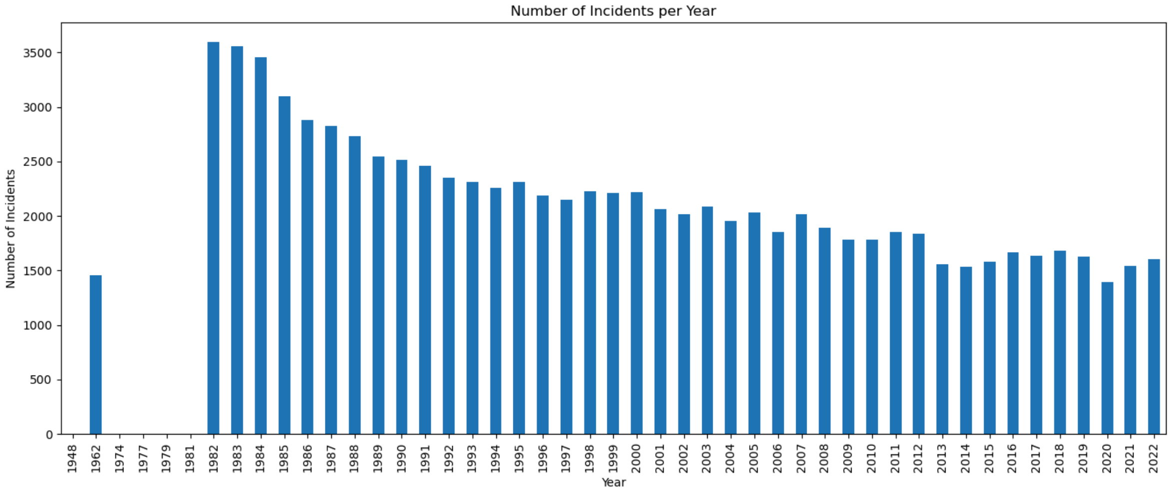
Findings: From the visualization these are the aircrafts that have had the most number of accidents. We definitely roll them out of our recommendation.

In [23]:
```
#Fatal vs Serious vs Minor Injuries by Model
top_models = df_filtered['Model'].value_counts().head(5).index
injury_summary = df_filtered[df_filtered['Model'].isin(top_models)][['Mod
injury_summary = injury_summary.groupby('Model').sum()

injury_summary.plot(kind='bar', stacked=True, figsize=(12,6))
plt.title('Injury Summary for Top 5 Aircraft Models')
plt.ylabel('Number of Injuries')
plt.xlabel('Aircraft Model')
plt.tight_layout()
plt.show()
```



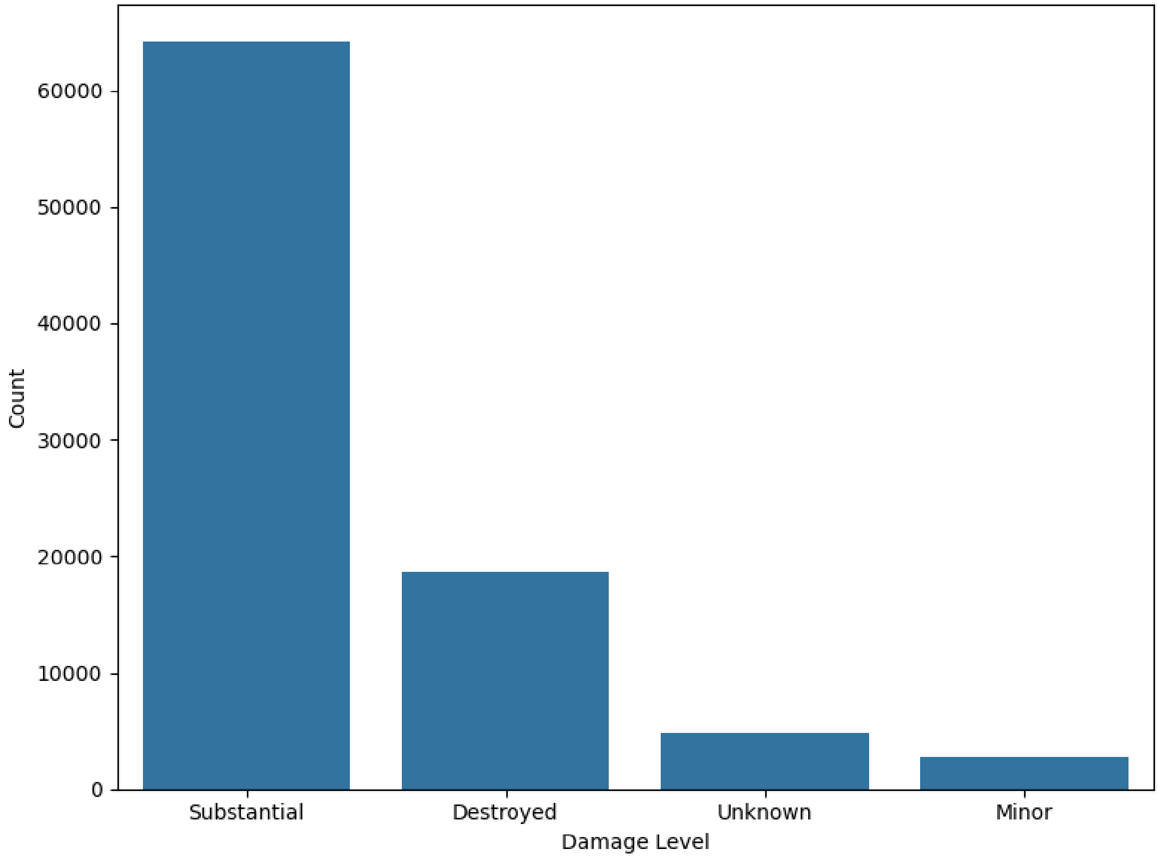
Findings: Out of the top aircraft models with most incidents we would want to understand the severity of the injuries. From the data a huge percentage of the injuries were either serious or fatal meaning the survival rate was minimal to zero

In [24]:
```
#Accident trend over time
df_filtered['Event.Date'] = pd.to_datetime(df_filtered['Event.Date'], err

plt.figure(figsize=(14,6))
df_filtered['Event.Date'].dt.year.value_counts().sort_index().plot(kind='
plt.title('Number of Incidents per Year')
plt.xlabel('Year')
plt.ylabel('Number of Incidents')
plt.tight_layout()
plt.show()
```



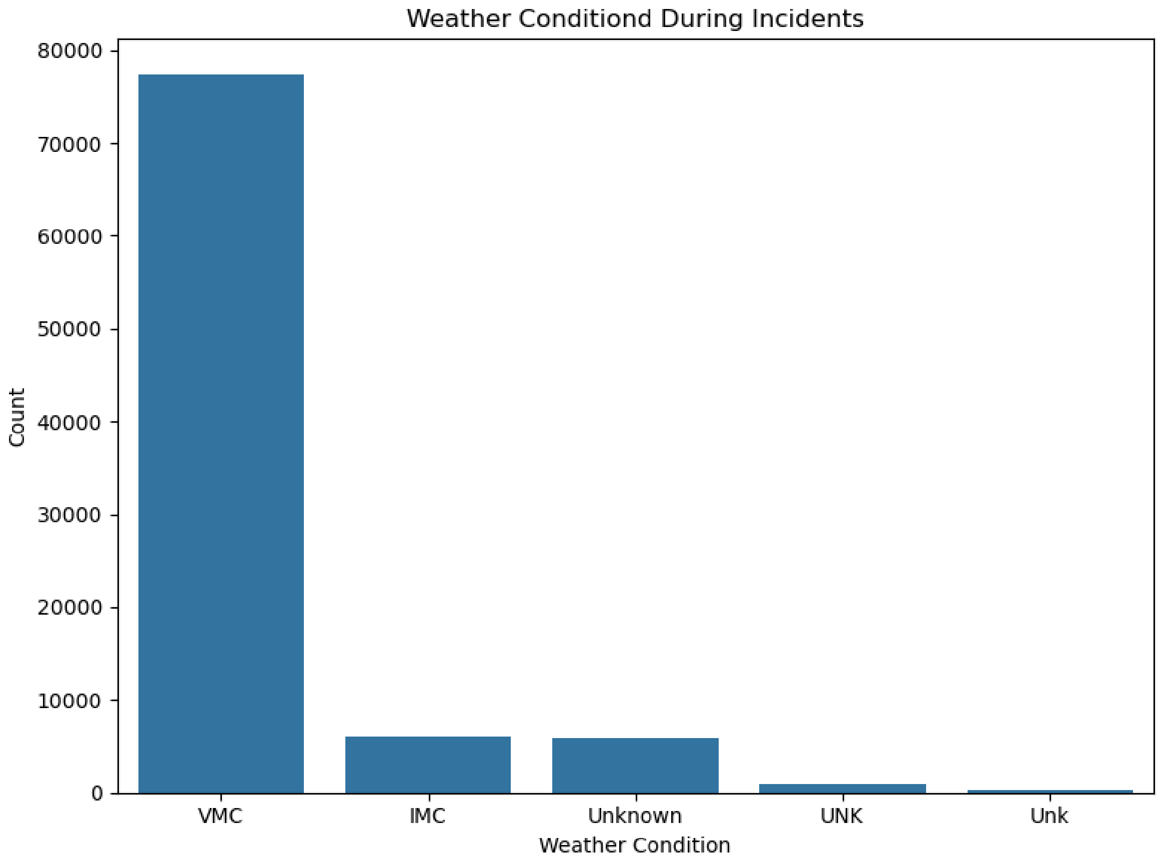
Findings: Over the years the number of incidents have reduced significantly. This may be due to technnological advancements or other factors that have improved overtime from a high of approximately 3,700 incidences in 1982 to approximately 1600 in 202. There has been an upward trend on the number of incidents but from the previous years trend there is a likelihood of the incidents reducing. It is an investment risk worth taking.

In [25]:
```
#Aircraft Damage Level
plt.figure(figsize=(8,6))
sns.countplot(data=df_filtered, x='Aircraft.damage', order=df_filtered['A
plt.xlabel('Damage Level')
plt.ylabel('Count')
plt.tight_layout()
plt.show()
```



Findings: Out of the reported incidents, approximately 65,000 of them the impact of damage is Quite substantial, around 18,000 were destroyed, 4,000 unknown and arounf 3,000 had minor damage. That means when an incident happens chances of incurring a major impact is high

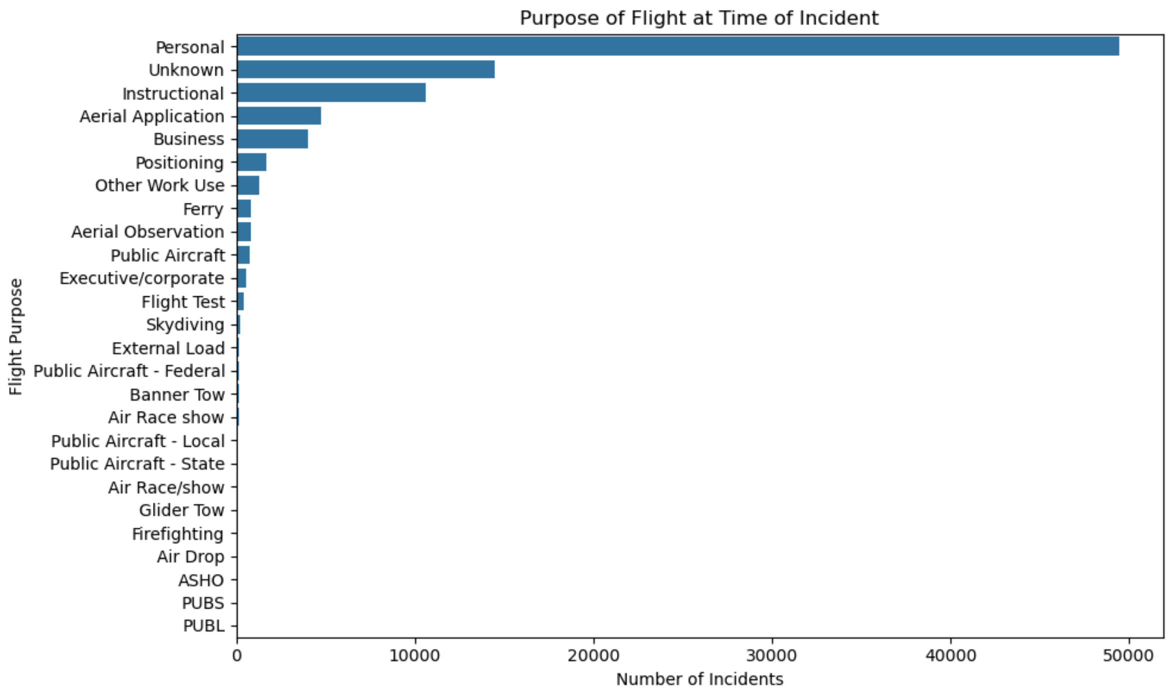
In [26]:
```
#Weather conditions during accidents
plt.figure(figsize=(8,6))
sns.countplot(data=df_filtered, x='Weather.Condition', order=df_filtered[
plt.title('Weather Conditiond During Incidents')
plt.xlabel('Weather Condition')
plt.ylabel('Count')
plt.tight_layout()
plt.show()
```



VMC – Visual Meteorological Conditions(Good visibility, no need for instruments to fly) IMC – Instrument Meteorological Conditions(Low visibility, clouds, fog, storms, heavy rain, etc) The incidents that happened during VMC re slightly above 76,000 and those that happened during IMC are slightly above 5,000. We would expect that we would have more incidents during IMC and not during VMC. What that means weather conditions does not significantly impact on the occurence of incidents.

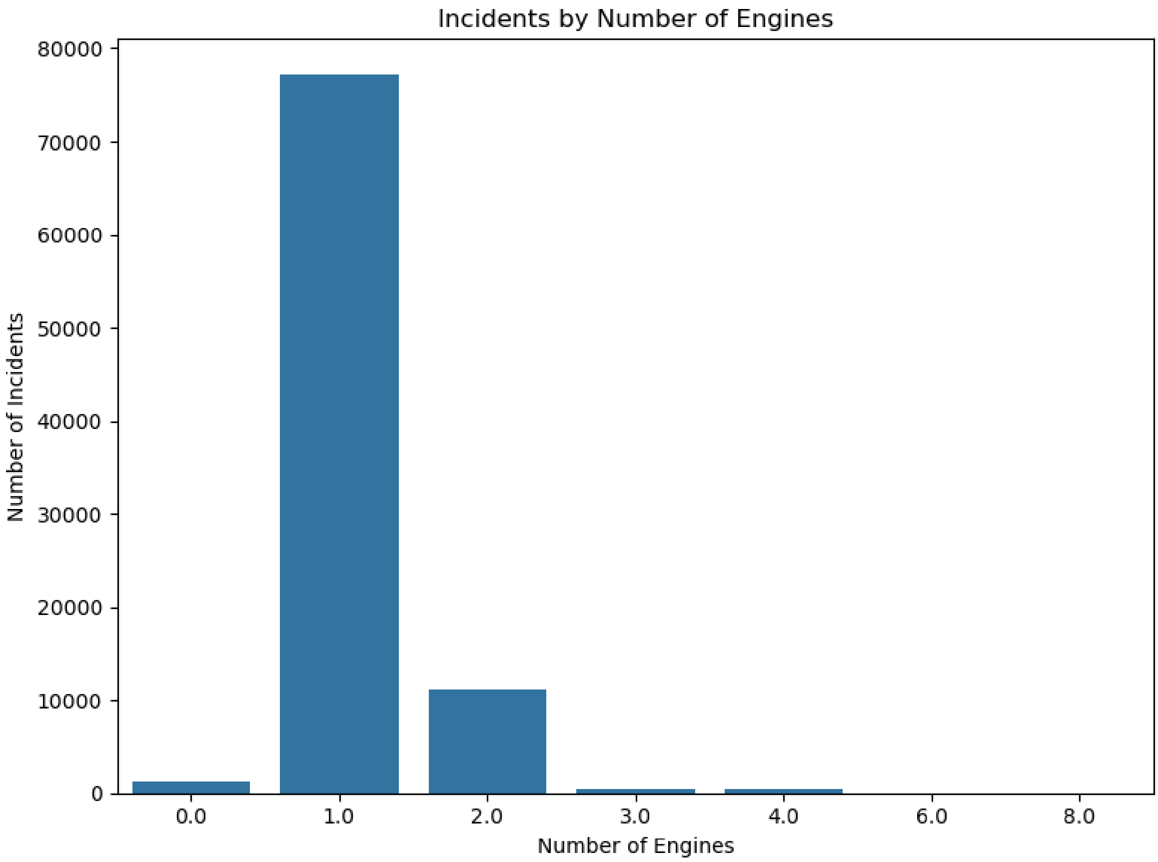
In [27]:

```
#purpose of flight
plt.figure(figsize=(10,6))
sns.countplot(data=df_filtered, y='Purpose.of.flight', order=df_filtered[
plt.title('Purpose of Flight at Time of Incident')
plt.xlabel('Number of Incidents')
plt.ylabel('Flight Purpose')
plt.tight_layout()
plt.show()
```



Findings: Personal aircrafts are more prone to accidents. Out of the reported incidents around 48,000 are from personal aircrafts Business aircrafts have experienced around 4,500 incidents. As a business advisor the investor can invest more in business aircraft .

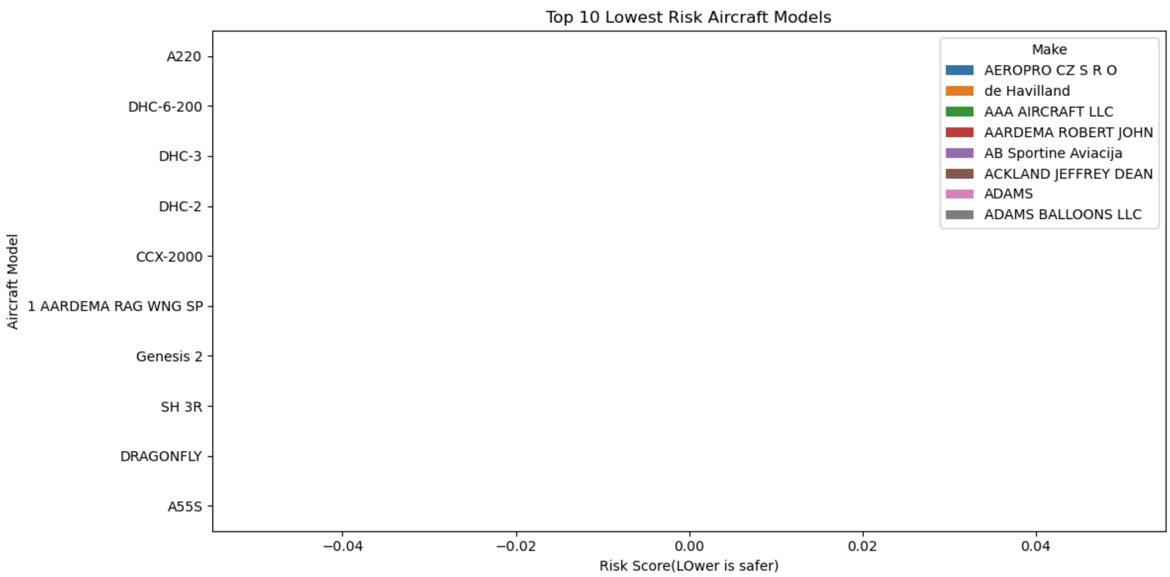
In [28]:
```
#engine count vs Incident Frequency
plt.figure(figsize=(8,6))
sns.countplot(data=df_filtered, x='Number.of.Engines')
plt.title('Incidents by Number of Engines')
plt.xlabel('Number of Engines')
plt.ylabel('Number of Incidents')
plt.tight_layout()
plt.show()
```



Findings: One engine aircrafts are prone to incidents. Around 76,000 of incidents reported are from one-engine aircrafts. Those with 3 or 4 engines reported around 1,000 incidents and those with 6 to 8 engines reported no case. What that implies is the more the engines the less the risk

In [29]:

```
#Top 10 Lowest risk aircraft models
plt.figure(figsize=(12,6))
sns.barplot(data=safe_aircraft.sort_values(by='Risk_Score').head(10), x='
plt.title('Top 10 Lowest Risk Aircraft Models')
plt.xlabel('Risk Score(LOwer is safer)')
plt.ylabel('Aircraft Model')
plt.tight_layout()
plt.show()
```



Findings: In terms of aircraft make the top 5 recommendations are AEROPRO CZ S R O, de havilland etc where as the best aircraft model isA55S, DRAGON FLY onwards. The makes and model are less prone to accidents and would definately recommend

In [ ]: