Import libraries

In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.patches as mpatches
import seaborn as sns
```

Read CSV and apply filtering

In [3]:
```
data = pd.read_csv("fatalities.csv")
```

Filter data which is only from palestine or israel

In [4]:
```
data = data[(data['citizenship']  == 'Palestinian') | (data['citizenship'] ==
```

Data Analysis and Cleansing

Exploratory data analysis

Get start elements of data

In [5]:
```
data.head()
```

Out[5]:

	name	date_of_event	age	citizenship	event_location	event_location_district	event_loca
0	'Abd a-Rahman Suleiman Muhammad Abu Daghash	24/09/2023	32.0	Palestinian	Nur Shams R.C.	Tulkarm	
1	Usayed Farhan Muhammad 'Ali Abu 'Ali	24/09/2023	21.0	Palestinian	Nur Shams R.C.	Tulkarm	
2	'Abdallah 'Imad Sa'ed Abu Hassan	22/09/2023	16.0	Palestinian	Kfar Dan	Jenin	
3	Durgham Muhammad Yihya al-Akhras	20/09/2023	19.0	Palestinian	'Aqbat Jaber R.C.	Jericho	
4	Raafat 'Omar Ahmad Khamaisah	19/09/2023	15.0	Palestinian	Jenin R.C.	Jenin	

Get last elements of data

In [6]:
```
data.tail()
```

Out[6]:

	name	date_of_event	age	citizenship	event_location	event_location_district	event_
11119	Binyamin Herling	19/10/2000	64.0	Israeli	Nablus	Nablus	
11120	Farid Musa 'Issa a-Nesasreh	17/10/2000	28.0	Palestinian	Beit Furik	Nablus	
11121	Hillel Lieberman	07/10/2000	36.0	Israeli	Nablus	Nablus	
11122	Fahed Mustafa 'Odeh Baker	07/10/2000	21.0	Palestinian	Bidya	Salfit	
11123	Wichlav Zalsevsky	02/10/2000	24.0	Israeli	Masha	Salfit	

Get info of data

In [7]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 11121 entries, 0 to 11123
Data columns (total 16 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   name                            11121 non-null  object
 1   date_of_event                   11121 non-null  object
 2   age                             10992 non-null  float64
 3   citizenship                     11121 non-null  object
 4   event_location                  11121 non-null  object
 5   event_location_district         11121 non-null  object
 6   event_location_region           11121 non-null  object
 7   date_of_death                   11121 non-null  object
 8   gender                          11101 non-null  object
 9   took_part_in_the_hostilities    9693 non-null   object
 10  place_of_residence              11053 non-null  object
 11  place_of_residence_district     11053 non-null  object
 12  type_of_injury                  10830 non-null  object
 13  ammunition                      5869 non-null   object
 14  killed_by                       11121 non-null  object
 15  notes                           10841 non-null  object
dtypes: float64(1), object(15)
memory usage: 1.4+ MB
```

Check how many null values avilable in data in each column

In [8]:
```
data.isna().sum()
```

Out[8]:
```
name                                0
date_of_event                       0
age                               129
citizenship                         0
event_location                      0
event_location_district             0
event_location_region               0
date_of_death                       0
gender                             20
took_part_in_the_hostilities     1428
place_of_residence                 68
place_of_residence_district        68
type_of_injury                    291
ammunition                       5252
killed_by                           0
notes                             280
dtype: int64
```

Check number of duplicates in data

In [9]:
```
data.duplicated().sum()
```

Out[9]:
```
7
```

Data cleaning

Let's get rid of duplicate entries

In [10]:
```
data.drop_duplicates(keep='first',inplace=True)
```

Fill null values

In [11]:
```
data['age'].fillna(data['age'].median(), inplace=True)
data['gender'].fillna(data['gender'].mode()[0],inplace=True)
data['place_of_residence'].fillna(data['place_of_residence'].mode()[0],inplace
data['place_of_residence_district'].fillna(data['place_of_residence_district']
data['citizenship'].fillna(data['citizenship'].mode()[0], inplace=True)
data['type_of_injury'].fillna('Unknown',inplace=True)
data['took_part_in_the_hostilities'].fillna(data['took_part_in_the_hostilities
data['ammunition'].fillna('Unknown',inplace=True)
data['notes'].fillna('',inplace=True)
```

Convert columns of data to correct format for better data analysis

In [12]:
```
data['date_of_event'] = data['date_of_event'].astype('datetime64[ns]')
data['date_of_death'] = data['date_of_death'].astype('datetime64[ns]')
data['age'] = data['age'].astype(int)
data['gender'] = data['gender'].astype(str)
data['place_of_residence'] = data['place_of_residence'].astype(str)
data['place_of_residence_district'] = data['place_of_residence_district'].asty
data['type_of_injury'] = data['type_of_injury'].astype(str)
data['took_part_in_the_hostilities'] = data['took_part_in_the_hostilities'].as
data['ammunition'] = data['ammunition'].astype(str)
data['notes'] = data['notes'].astype(str)
```

```
C:\Users\shafi\AppData\Local\Temp\ipykernel_13368\705076602.py:1: UserWarnin
g: Parsing dates in %d/%m/%Y format when dayfirst=False (the default) was sp
ecified. Pass `dayfirst=True` or specify a format to silence this warning.
  data['date_of_event'] = data['date_of_event'].astype('datetime64[ns]')
C:\Users\shafi\AppData\Local\Temp\ipykernel_13368\705076602.py:2: UserWarnin
g: Parsing dates in %d/%m/%Y format when dayfirst=False (the default) was sp
ecified. Pass `dayfirst=True` or specify a format to silence this warning.
  data['date_of_death'] = data['date_of_death'].astype('datetime64[ns]')
```

Get description of data and some important statistical measures

In [13]:
```
data.describe()
```

Out[13]:

	date_of_event	age	date_of_death
count	11114	11114.000000	11114
mean	2009-11-01 00:45:51.988483072	26.704427	2009-11-03 23:56:37.876551936
min	2000-10-02 00:00:00	1.000000	2000-10-02 00:00:00
25%	2004-05-18 00:00:00	19.000000	2004-05-19 00:00:00
50%	2008-12-29 00:00:00	23.000000	2008-12-30 00:00:00
75%	2014-07-26 00:00:00	31.000000	2014-07-26 00:00:00
max	2023-09-24 00:00:00	112.000000	2023-09-24 00:00:00
std	NaN	13.711165	NaN

Sns styles

Set figure size to some default value

In [14]:
```
sns.set(rc={'figure.figsize':(16, 7)})
```

Set Color palette, axes and grid color

In [15]:
```
sns.set(palette='pastel')
sns.set_style({"axes.facecolor": "#dcdcdc", "grid.color": "#f5f5f5"})
```

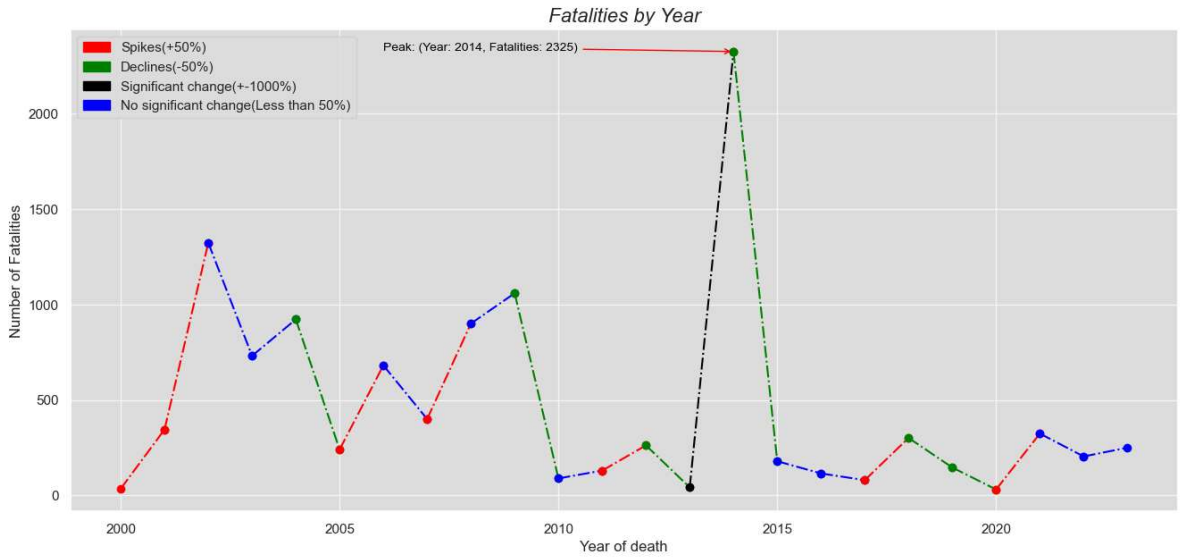
Q-1

Explore the dataset and identify the trends in fatalities over time.

Identify any significant changes, spikes, or declines in the number of fatalities

Fatalities trend over time

In [16]:
```
# Get counts of deaths based on year and find percentage change of current and
fatalities_count_by_year = data["date_of_death"].dt.year.value_counts().sort_i
percentage_change_based_on_previous = fatalities_count_by_year.pct_change()

# Threshhold variables and array for graph lines and colours
spike_threshhold = 0.5
decline_threshhold = -0.5
significant_change_threshhold_spike = 10
significant_change_threshhold_decline = -significant_change_threshhold_spike
lines_and_its_colours = []

# Add lines and their colours in array based on threshhold
for i in range(1, len(percentage_change_based_on_previous)):
    if percentage_change_based_on_previous.iloc[i] >= significant_change_thres
          percentage_change_based_on_previous.iloc[i] <= significant_change_th
        lines_and_its_colours.append((fatalities_count_by_year.index[i - 1], '
    elif percentage_change_based_on_previous.iloc[i] >= spike_threshhold:
        lines_and_its_colours.append((fatalities_count_by_year.index[i - 1], '
    elif percentage_change_based_on_previous.iloc[i] <= decline_threshhold:
        lines_and_its_colours.append((fatalities_count_by_year.index[i - 1], '
    else:
        lines_and_its_colours.append((fatalities_count_by_year.index[i - 1], '

# Change figure size and add lines to graph
for i in range(0, len(lines_and_its_colours)):
    line_start_year, line_color = lines_and_its_colours[i]
    line_end_year = line_start_year + 1
    start_value, end_value = fatalities_count_by_year.loc[line_start_year], fa
    plt.plot([line_start_year, line_end_year], [start_value, end_value], color

# Get the peak point and add arrow to it. Add labels and titles
peak_y = fatalities_count_by_year.values[np.argmax(fatalities_count_by_year.va
peak_x = fatalities_count_by_year.index[np.argmax(fatalities_count_by_year.val

plt.annotate(f'Peak: (Year: {peak_x}, Fatalities: {peak_y})', xy=(peak_x, peak
             arrowprops=dict(arrowstyle='->', color='red'), fontsize=10, colo

plt.xlabel('Year of death', fontsize=12)
plt.ylabel('Number of Fatalities', fontsize=12)
plt.title('Fatalities by Year', fontsize=16, fontstyle='italic')

# Add grid lines to graph and also patches to the legend and then show the gra
plt.grid(True)
redPatch = mpatches.Patch(color='red', label=f'Spikes(+{int(spike_threshhold *
greenPatch = mpatches.Patch(color='green', label=f'Declines({int(decline_thres
blackPatch = mpatches.Patch(color='black', label=f'Significant change(+-{int(s
bluePatch = mpatches.Patch(color='blue', label=f'No significant change(Less th
plt.legend(handles=[redPatch, greenPatch, blackPatch, bluePatch], loc='upper l
plt.show()
```



As shown in graph, We have spikes on years with red color and decline with green colour. Blue color shows less than 50% decline or spike according to previous data point. Black color shows huge spike or decline which is10X with respect to previous data point.

This shows that we have spikes in year of 2001, 2002, 2006, 2008, 2012, 2018, 2021 where increase in fatalities is 50% greater than previous year. We have declines in year of 2005, 2010, 2013, 2015, 2019, 2020 where decline in fatalities is 50% greater than previous year.

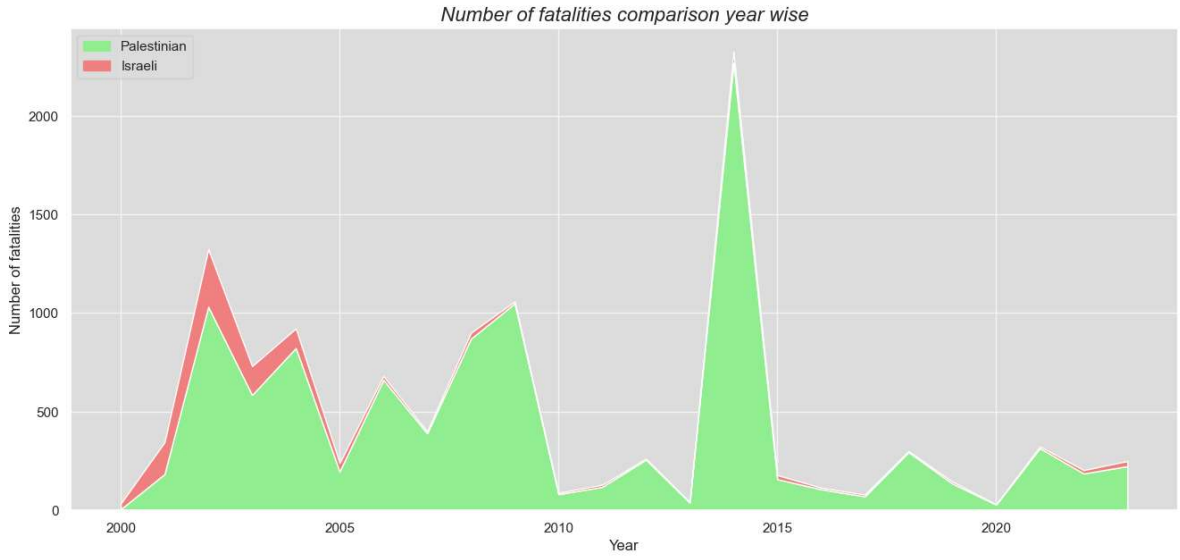
We have less than 50% change(w.r.t previous year) in fatalities in year of 2003, 2004, 2007, 2009, 2011, 2016, 2017, 2022, 2023.

We have more than 10X change in year of **2014** w.r.t 2013. It is peak value with **2325** fatalities.

Number of fatalities comparison by citizenship year wise

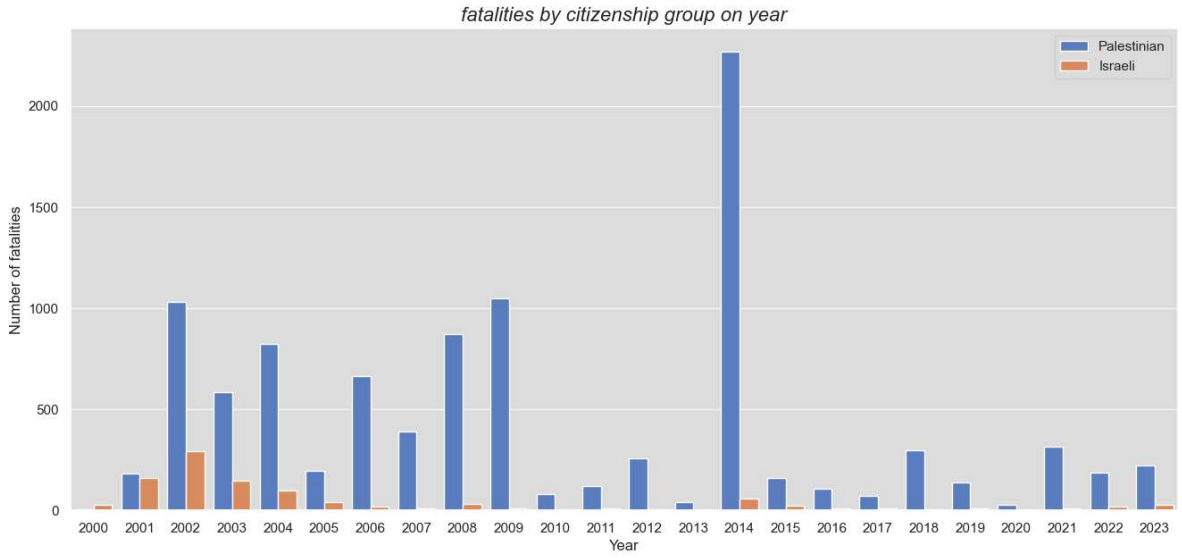
In [17]:
```
comparisonData = data.groupby([data["date_of_death"].dt.year, "citizenship"])[
unstackedComparisonData = comparisonData.unstack()
unstackedComparisonData['Israeli']
unstackedComparisonData['Palestinian']
unstackedComparisonData.index
plt.stackplot(unstackedComparisonData.index, unstackedComparisonData['Palestir
              unstackedComparisonData['Israeli'], colors =['lightgreen', 'ligh
plt.xlabel('Year',fontsize=12)
plt.ylabel('Number of fatalities',fontsize=12)
plt.title('Number of fatalities comparison year wise',fontsize=16, fontstyle='
PalestinianPatch = mpatches.Patch(color='lightgreen', label='Palestinian')
IsraeliPatch = mpatches.Patch(color='lightcoral', label='Israeli')
plt.legend(handles=[PalestinianPatch, IsraeliPatch], loc='upper left')
```

Out[17]:
```
<matplotlib.legend.Legend at 0x275a4834990>
```



Here in above image we can see that palestine fatalities over time is very large and dense as compared to israel fatalities over time. This can be more clear in the below image where we compare each year fatalities citizenship wise using bar plot.

In [18]:
```
sns.countplot(x=data['date_of_death'].dt.year, hue=data['citizenship'], palett
plt.legend(loc='upper right')
plt.xlabel('Year',fontsize=12)
plt.ylabel('Number of fatalities',fontsize=12)
plt.title('fatalities by citizenship group on year', fontsize = 16, fontstyle=
plt.show()
```



As the above image showing that Israel have very less fatalities over time. This also shows that in **2014** there is huge increase in fatalities where palestine fatalities are very high while israel fatalities are very low. Also in other years like in 2009 there are huge palestine fatalities while israel have near to no fatality. Also in other years israel fatalities are minor which shows israel brutality over the palestinian people.

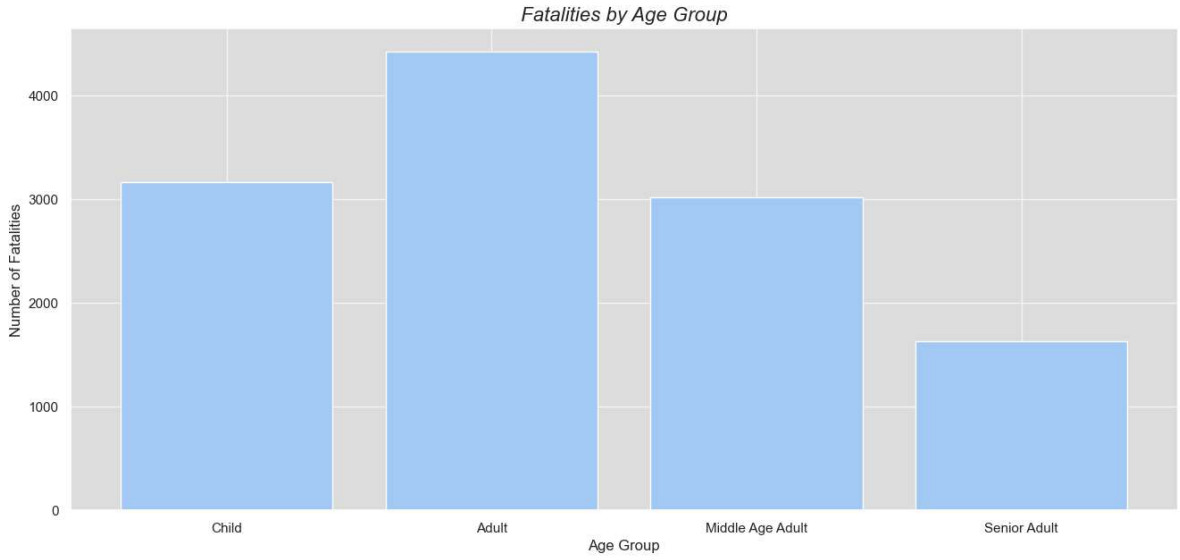
Q-2

Conduct an analysis by examining the age, gender, and citizenship of the individuals killed.

Determine if there are any notable patterns or disparities in the data.

Analysis By Age(Age Groups)

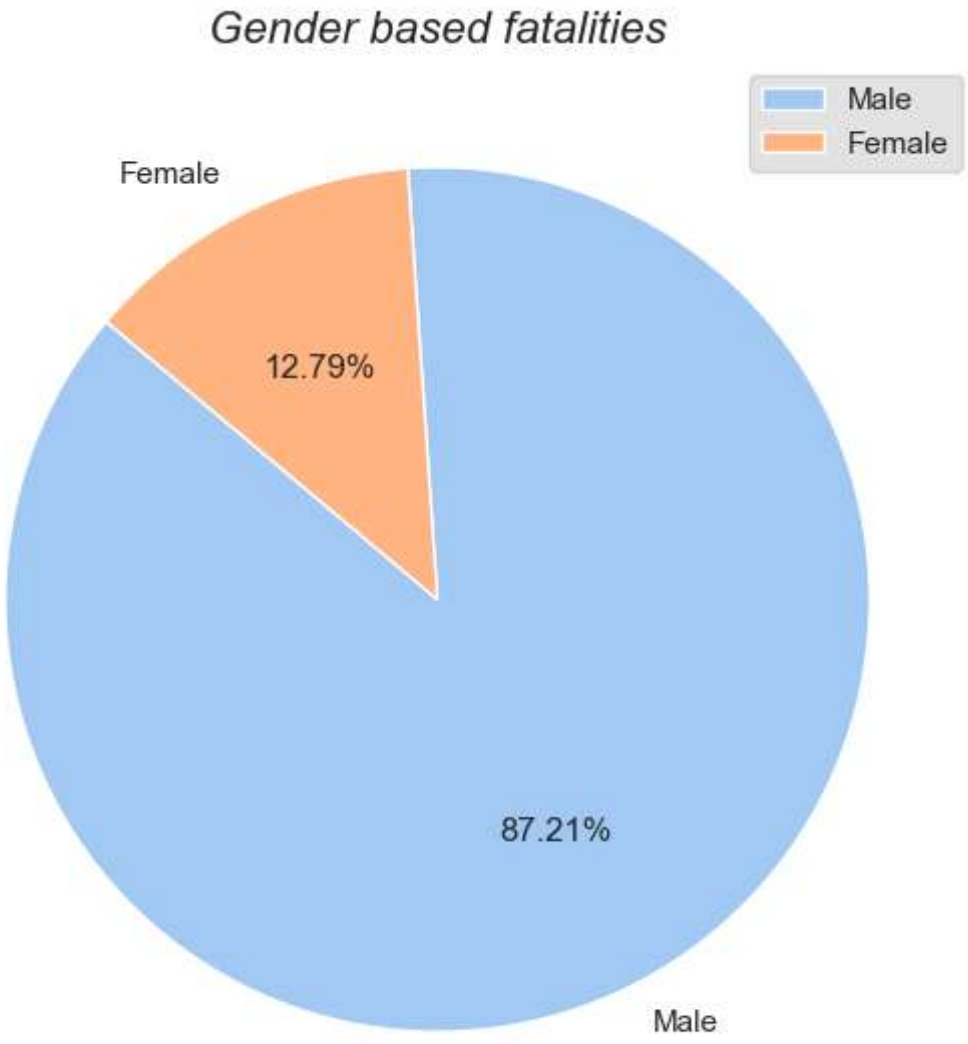
In [19]:
```
# Get count of deaths based on age of victim
age_value_counts_sotred = data['age'].value_counts().sort_index()

# Bar chart based on age groups
# Age groups based on age
child = age_value_counts_sotred.loc[0:19].sum()
adult = age_value_counts_sotred.loc[19:26].sum()
middleAgeAdult = age_value_counts_sotred.loc[26:41].sum()
seniorAdult = age_value_counts_sotred.loc[40:].sum()

# plt.figure(figsize=(16, 7))
plt.bar(['Child', 'Adult', 'Middle Age Adult', 'Senior Adult'],
        [child, adult, middleAgeAdult, seniorAdult])
plt.xlabel('Age Group', fontsize=12)
plt.ylabel('Number of Fatalities',fontsize=12)
plt.title('Fatalities by Age Group',fontsize=16,fontstyle='italic')
plt.show()
```



It is clear in this image that there are more number of fatalities in adult category then it's child who was victim and senior adult are very less than other categories.

Analysis By Gender

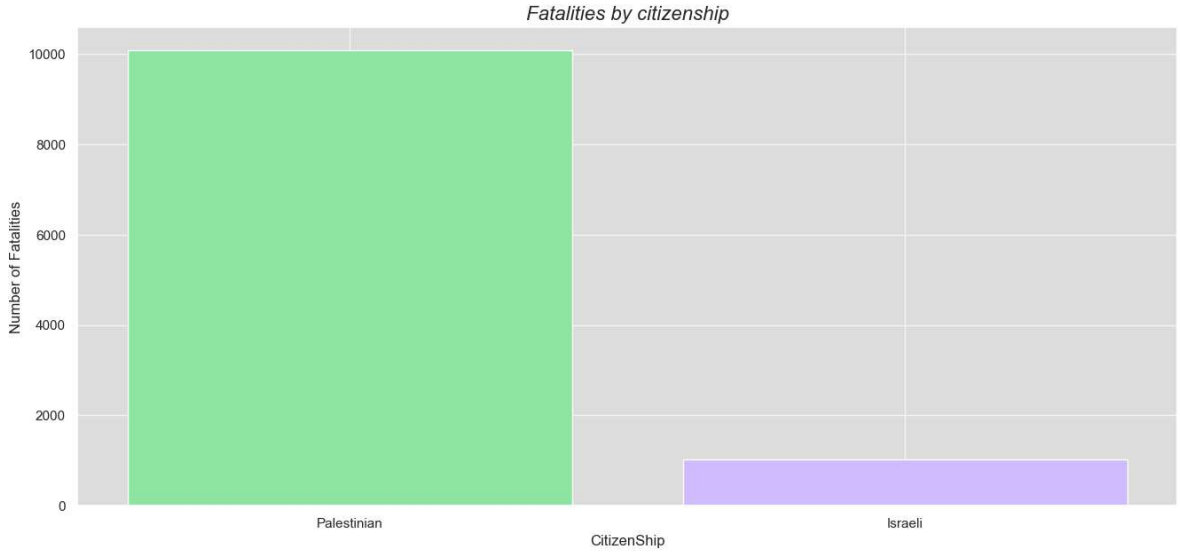
In [20]:
```
# Pie chart based on gender
gender_value_counts = data['gender'].value_counts()
# plt.figure(figsize=(7, 7))
plt.pie(gender_value_counts, labels=gender_value_counts.index.map(lambda value
                                                                  else 'Female
plt.title('Gender based fatalities',fontsize=16,fontstyle='italic')
plt.legend()
plt.show()
```



This graph shows that there are males who are more killed with 87.21% fatalities percentage while females percentage is 12.79% which is 7X less than males.

Analysis By Citizenship

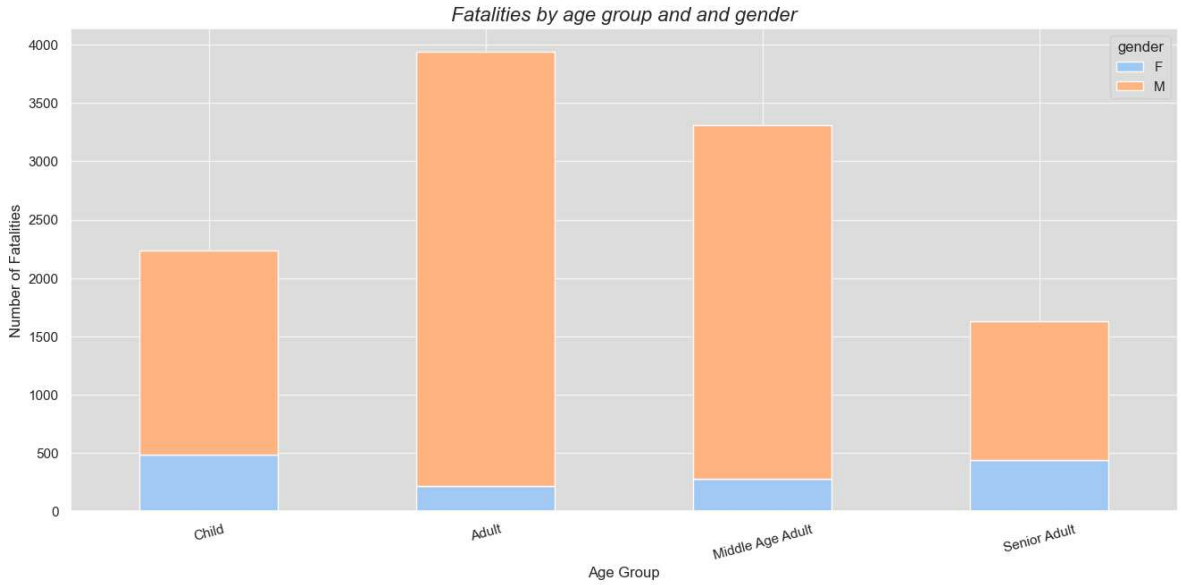
In [21]:
```
# Bar plot based on citizendhip
citizenship_value_counts = data['citizenship'].value_counts()
# plt.figure(figsize=(16, 7))s
plt.bar(citizenship_value_counts.index, citizenship_value_counts.values, color
plt.xlabel('CitizenShip', fontsize=12)
plt.ylabel('Number of Fatalities',fontsize=12)
plt.title('Fatalities by citizenship',fontsize=16,fontstyle='italic')
plt.xticks(rotation=0)
plt.show()
```



If we make comparison based on citizenship it is clear that palestine fatalities are much greater as compared to israel fatalities.

Analysis By Age(Age Groups) and Gender

In [22]:
```
# Stacked plots
df = pd.DataFrame(data) # Copy data in df to avoid manipulation in real data
# Age bins and groups distribution
age_bins = [0, 18, 25, 40, 120]
age_labels = ['Child', 'Adult', "Middle Age Adult", 'Senior Adult']
df['age_group'] = pd.cut(df['age'], bins=age_bins, labels=age_labels, right=Fa

# Stack bar plot between age group and gender
pivot_df = df.pivot_table(index='age_group'
                          , columns='gender', values='age', aggfunc='count')
pivot_df.plot(kind='bar', stacked=True)

plt.xlabel('Age Group', fontsize=12)
plt.ylabel('Number of Fatalities',fontsize=12)
plt.title('Fatalities by age group and and gender',fontsize=16,fontstyle='ital
plt.xticks(rotation=15)
plt.show()
```

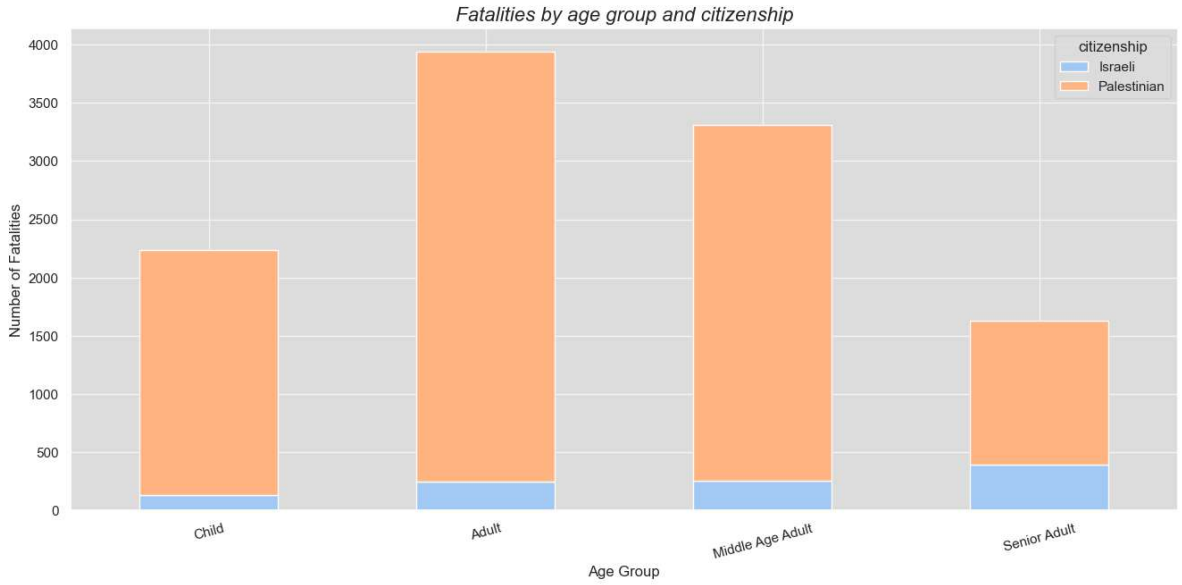


If we compare fatalities based on age group and gender, we can see that male adults fatalities are maximum overall.

- Females fatalities are maximum in child age category while minimum in adult age category.
- Male fatalities are maximum in adult age category while minimum in senior adult.

Analysis By Age(Age Groups) and Citizenship

In [23]:
```
# Stacked plots
df = pd.DataFrame(data) # Copy data in df to avoid manipulation in real data
# Age bins and groups distribution
age_bins = [0, 18, 25, 40, 120]
age_labels = ['Child', 'Adult', "Middle Age Adult", 'Senior Adult']
df['age_group'] = pd.cut(df['age'], bins=age_bins, labels=age_labels, right=Fa

# Stack bar plot between age group and citizenship
pivot_df = df.pivot_table(index='age_group'
                          , columns='citizenship', values='age', aggfunc='cour
pivot_df.plot(kind='bar', stacked=True)
plt.xlabel('Age Group', fontsize=12)
plt.ylabel('Number of Fatalities',fontsize=12)
plt.title('Fatalities by age group and citizenship',fontsize=16,fontstyle='ita
plt.xticks(rotation=15)
plt.show()
```



If we perform comparison based on age group and citizenship, we can see that palestinian adults fatalities are maximum overall.

- Palestine fatalities are maximum in adult age category while minimum in senior adult age category.
- Israel fatalities are maximum in senior adult while minimum in child age category.
- We can compare that palestinian children killed are much higher that israel senior adult killing which is huge proof of israel brutality on palestinian people.

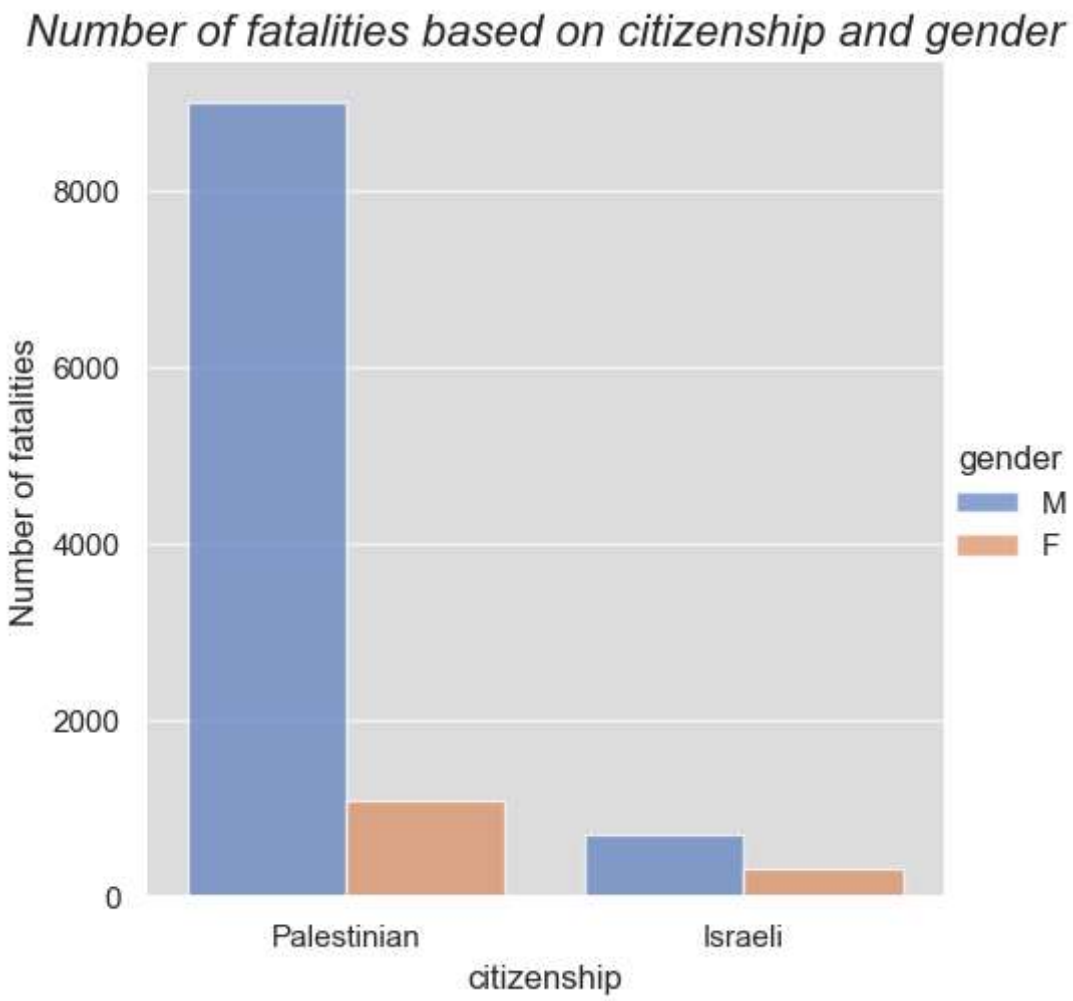
> The maximum killing here are palestinian adults which shows israel plan to wipe out the whole generation of palestinian people to decrease their moral and destroy their future.

Analysis By Gender and Citizenship

In [24]:
```
sns.catplot(
    data=data, kind="count",
    x="citizenship", hue="gender",
    palette="muted", alpha=.7, height=5
).despine(left=True)
plt.ylabel("Number of fatalities", fontsize=12)
plt.title("Number of fatalities based on citizenship and gender", fontsize=16,
plt.figure(figsize=(10,10))
```

```
C:\Users\shafi\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarn
ing: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```

Out[24]:
```
<Figure size 1000x1000 with 0 Axes>
```




```
<Figure size 1000x1000 with 0 Axes>
```

This comparison shows fatalities based on citizenship and gender.

- This shows that maximum fatilities are for palestinian males and minumun for israeli females.
- It shows a big difference between israel and palestine fatalities.

> Here we can get that even palestinian female fatalities are greater that the israel males fatalities.

In [25]:
```
data.groupby(['citizenship', 'gender']).size()
```

Out[25]:
```
citizenship  gender
Israeli      F          331
             M          698
Palestinian  F         1091
             M         8994
dtype: int64
```

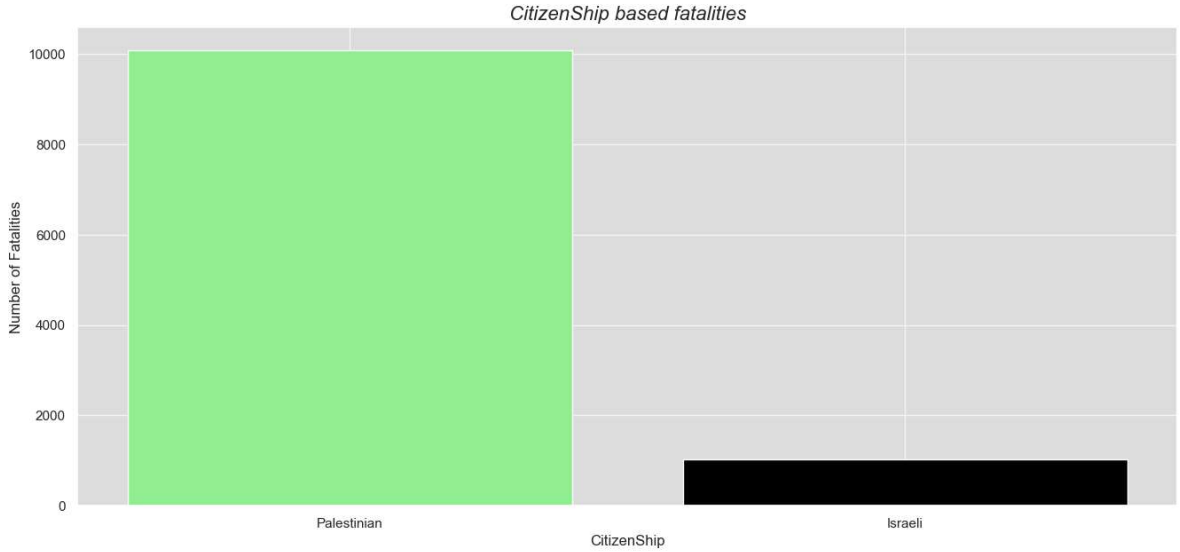
> Here we can see that all israel fatalities are 1029 while only palestinian females fatalities are 1091 which is greate than israel all fatalities.
> This shows the brutal attack of israel on palestinian people.

Q-3

Visualize the distribution of fatalities and identify areas that have experienced higher levels of violence.

Fatalities based on citizenship

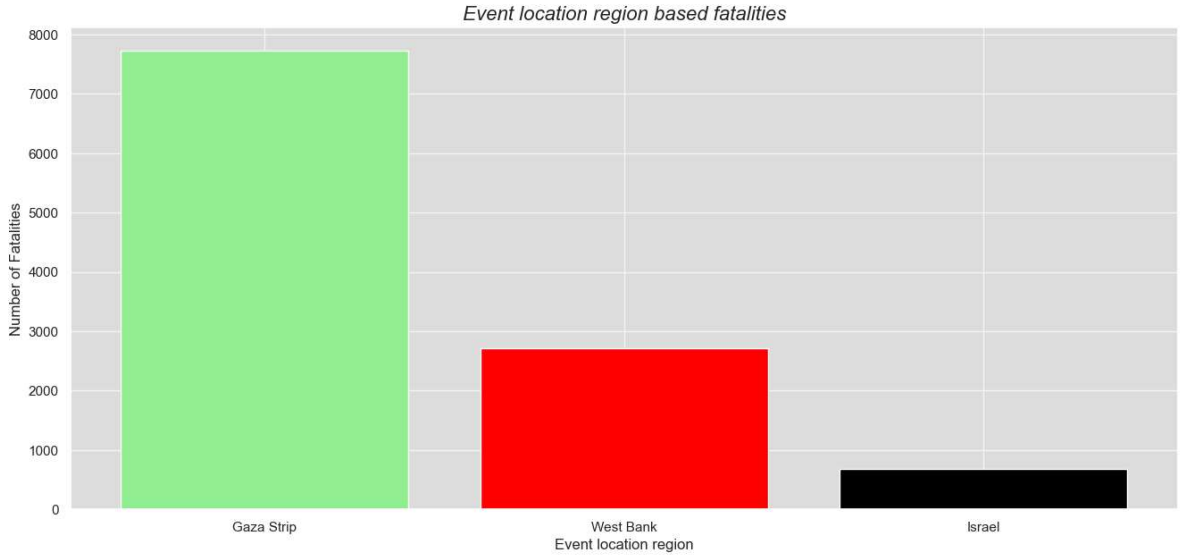
In [26]:
```
# Bar plot based on citizenship value counts
citizenship_value_counts = data['citizenship'].value_counts()
plt.figure(figsize=(16, 7))
plt.bar(citizenship_value_counts.index, citizenship_value_counts.head(5).value
plt.xlabel('CitizenShip',fontsize=12)
plt.ylabel('Number of Fatalities',fontsize=12)
plt.title('CitizenShip based fatalities',fontsize=16,fontstyle='italic')
plt.xticks(rotation=0)
plt.show()
```



This graph shows that palestine fatalities are higher.

Region based fatalities

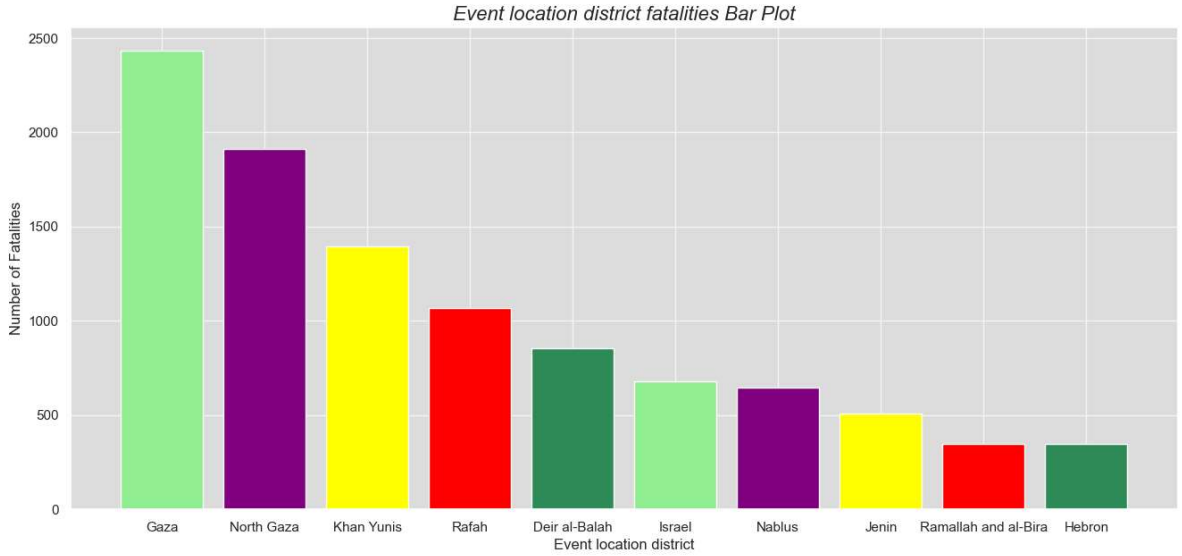
In [27]:
```
# Bar plot based on event location region value counts
event_location_region_value_counts = data['event_location_region'].value_count
plt.figure(figsize=(16, 7))
plt.bar(event_location_region_value_counts.index, event_location_region_value_
plt.xlabel('Event location region',fontsize=12)
plt.ylabel('Number of Fatalities',fontsize=12)
plt.title('Event location region based fatalities',fontsize=16,fontstyle='ital
plt.xticks(rotation=0)
plt.show()
```



This graph shows that Gaza strip region is experiencing higher level of violence while West Bank is on second for violence and israel is facing minimum violence.

District based fatalities

In [28]:
```
# Bar plot based on top 10 event location district value counts
event_location_district_value_counts = data['event_location_district'].value_c
plt.figure(figsize=(16, 7))
plt.bar(event_location_district_value_counts.head(10).index, event_location_di
        color=['lightgreen', 'purple', 'yellow', 'red', 'seagreen'])
plt.xlabel('Event location district',fontsize=12)
plt.ylabel('Number of Fatalities',fontsize=12)
plt.title('Event location district fatalities Bar Plot',fontsize=16,fontstyle=
plt.xticks(rotation=0)
plt.show()
```

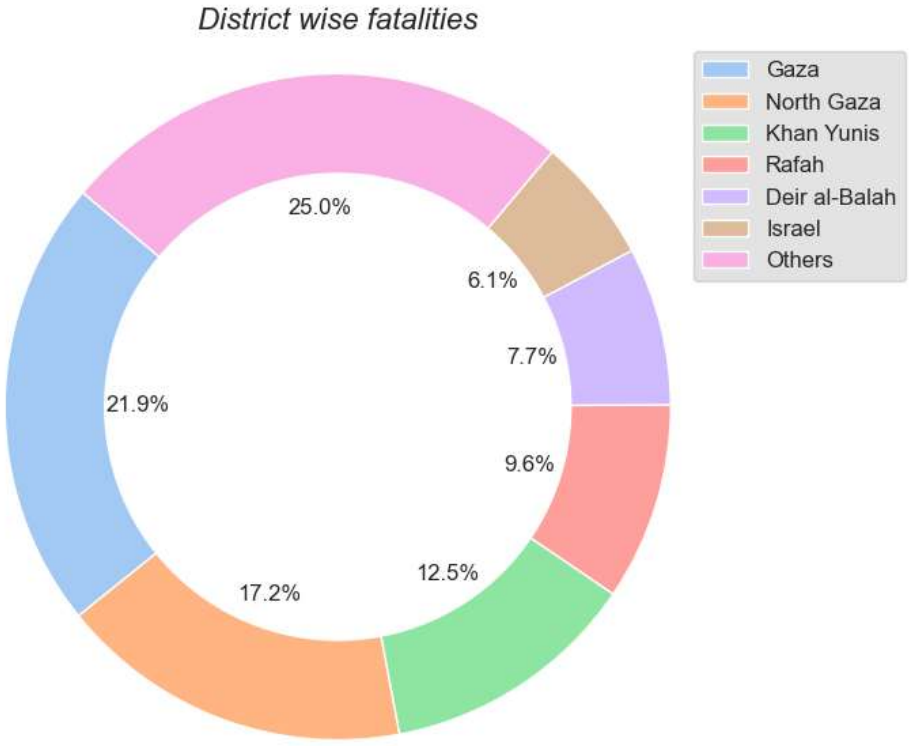


This graph shows that Gaza district is facing higher level of violence.

- North Gaza, Khan Yunis, Rafah and Deir-al-Balah is after this while Israel is on sixth number in facing violence.
- After them Nablus, Jenin, Ramallah and al-bira, and Herbon are facing violence.
- Some other district also facing violence but violence intensity in these are less.

  District showing here are facing higher level of violence.
  This will be more clear in below graph.

In [29]:
```
top_10_district = event_location_district_value_counts[:6]
others_count = event_location_district_value_counts[6:].sum()
others_series = pd.Series({'Others': others_count})
updated_district_counts = pd.concat([top_10_district, others_series])
plt.figure(figsize=(11, 7))
wedges, texts, autotexts = plt.pie(updated_district_counts, labels=updated_dis
plt.title('District wise fatalities', fontsize=16, fontstyle='italic')
for text in texts:
    text.set_visible(False)
plt.legend(wedges, updated_district_counts.index, loc='upper right', fontsize=
centre_circle = plt.Circle((0, 0), 0.7, color='white', linewidth=1)
fig = plt.gcf()
fig.gca().add_artist(centre_circle)
plt.axis('equal')
plt.show()
```



As the graph showing Gaza is facing higher level of violence with 21.9% while israel is facing 6.1% violence.
All district (other that showing in graph) combinally facing 25% violence.

Wordcloud of the notes in data

In [45]:
```
!pip install wordcloud
from wordcloud import WordCloud
```

```
Requirement already satisfied: wordcloud in c:\users\shafi\anaconda3\lib\sit
e-packages (1.9.2)
Requirement already satisfied: numpy>=1.6.1 in c:\users\shafi\anaconda3\lib
\site-packages (from wordcloud) (1.24.3)
Requirement already satisfied: pillow in c:\users\shafi\anaconda3\lib\site-p
ackages (from wordcloud) (9.4.0)
Requirement already satisfied: matplotlib in c:\users\shafi\anaconda3\lib\si
te-packages (from wordcloud) (3.7.2)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\shafi\anaconda3
\lib\site-packages (from matplotlib->wordcloud) (1.0.5)
Requirement already satisfied: cycler>=0.10 in c:\users\shafi\anaconda3\lib
\site-packages (from matplotlib->wordcloud) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\shafi\anaconda3
\lib\site-packages (from matplotlib->wordcloud) (4.25.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\shafi\anaconda3
\lib\site-packages (from matplotlib->wordcloud) (1.4.4)
Requirement already satisfied: packaging>=20.0 in c:\users\shafi\anaconda3\l
ib\site-packages (from matplotlib->wordcloud) (23.1)
Requirement already satisfied: pyparsing<3.1,>=2.3.1 in c:\users\shafi\anaco
nda3\lib\site-packages (from matplotlib->wordcloud) (3.0.9)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\shafi\anacon
da3\lib\site-packages (from matplotlib->wordcloud) (2.8.2)
Requirement already satisfied: six>=1.5 in c:\users\shafi\anaconda3\lib\site
-packages (from python-dateutil>=2.7->matplotlib->wordcloud) (1.16.0)
```

In [46]:
```
text = ''.join(df['notes'].dropna())
wordcloud = WordCloud(background_color='white', width=1366, height=768).genera
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.show()
```

As we're seeing wordcloud of the notes given on fatalities, we can easily observe that

- Most used word is Killed.
- After this, We can see some other famous words like soldier, shot, gunfire, Killed together, Exchange, friend, died, injured, home, house, military branch, Hamas military, Israel security, etc..

> This all shows huge words related to death and attacks. This all showing tense situation in the area of palestine and israel.

Q-4

Examine the types of injuries inflicted on individuals. Identify the most common types of

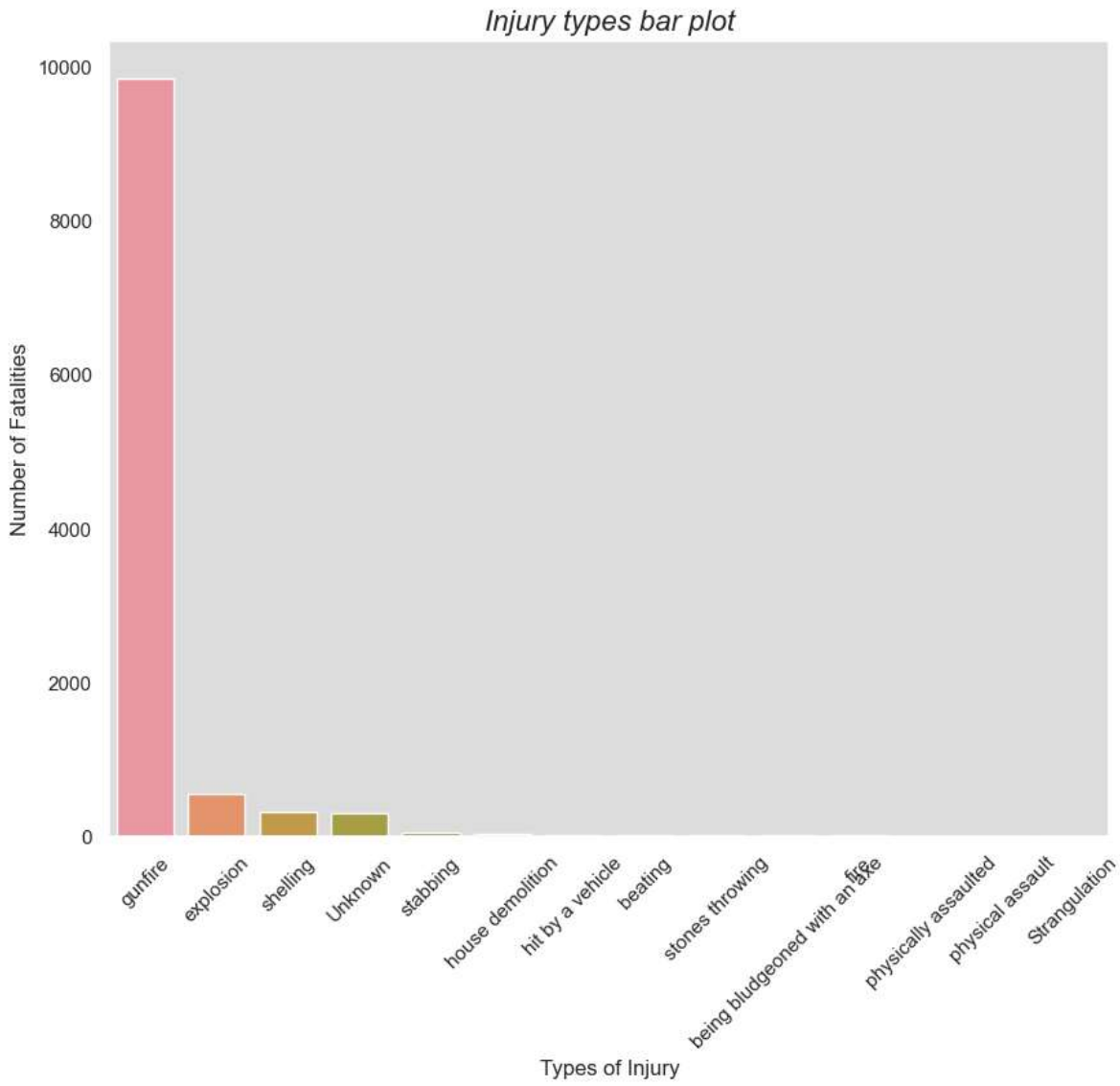
injuries and assess their severity

In [31]:
```
from pandas.api.types import CategoricalDtype
```

Bar Plot Describing The Type Of Injuries

In [32]:

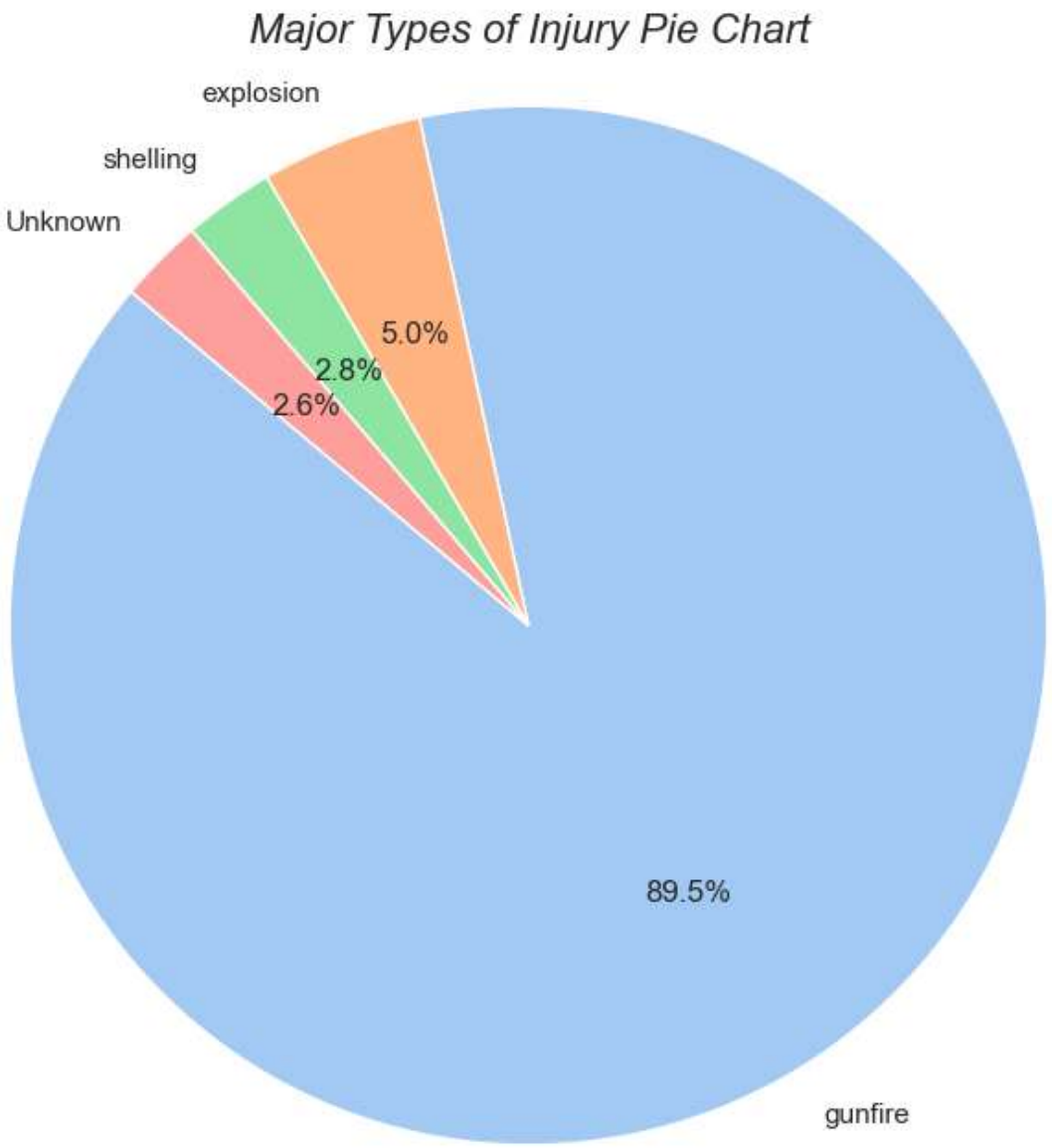
```
import seaborn as sns
import matplotlib.pyplot as plt



injury_types = data['type_of_injury']
injury_count = injury_types.value_counts()


plt.figure(figsize=(10, 8))
sns.barplot(x=injury_count.index, y=injury_count.values)
plt.xlabel('Types of Injury', fontsize=12)
plt.ylabel('Number of Fatalities', fontsize=12)
plt.title('Injury types bar plot', fontsize=16, fontstyle='italic')
plt.xticks(rotation=45)
plt.grid(axis='y')

plt.show()
```



Pie Chart Describing the Type of Injuries

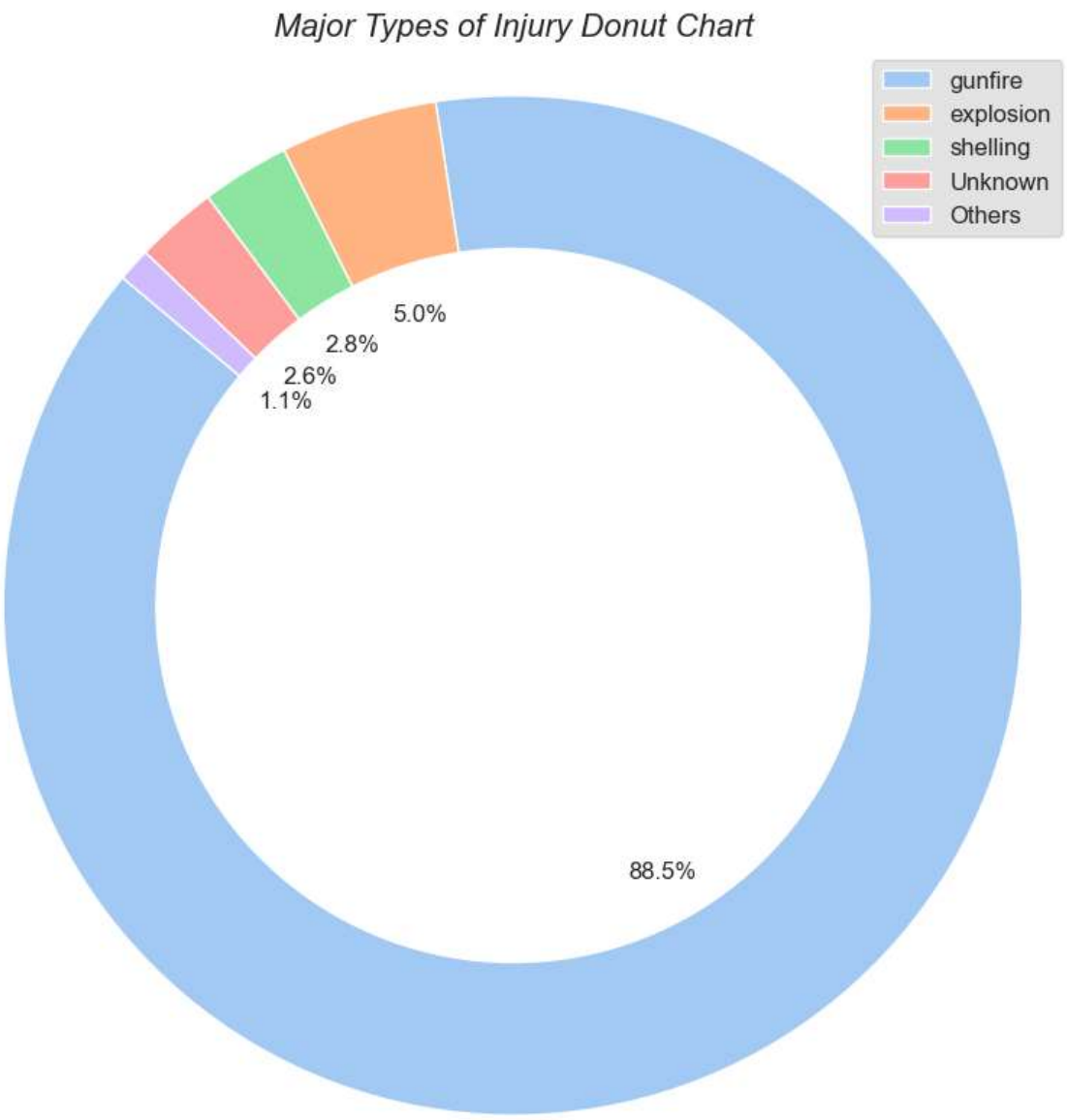
In [33]:

```
top_5_injuries = injury_count.head(4)


plt.figure(figsize=(8, 8))
plt.pie(top_5_injuries, labels=top_5_injuries.index, autopct='%1.1f%%', starta
plt.title('Major Types of Injury Pie Chart', fontsize=16, fontstyle='italic')
plt.axis('equal')
plt.show()
```



Donut Chart Describing Types of Injuries

In [34]:
```
import seaborn as sns
import matplotlib.pyplot as plt

top_5_injuries = injury_count.head(4)
other_count = injury_count[4:].sum()


updated_counts = pd.concat([top_5_injuries, pd.Series({'Others': other_count})

plt.figure(figsize=(10, 10))
wedges, texts, autotexts = plt.pie(updated_counts, labels=updated_counts.index
plt.title('Major Types of Injury Donut Chart', fontsize=16, fontstyle='italic'

for text in texts:
    text.set_visible(False)

plt.legend(wedges, updated_counts.index, loc='upper right', fontsize='medium')

centre_circle = plt.Circle((0, 0), 0.5, color='white', linewidth=0.8)
fig = plt.gcf()
fig.gca().add_artist(centre_circle)

plt.axis('equal')
plt.show()
```



The above Charts show the different types of injuries that occur in the Israel-Palestine conflict. The most common type of injury is gunfire, which is almost 88.5% of all injuries. Explosion is the second most common type of injury about 5% of all injuries.

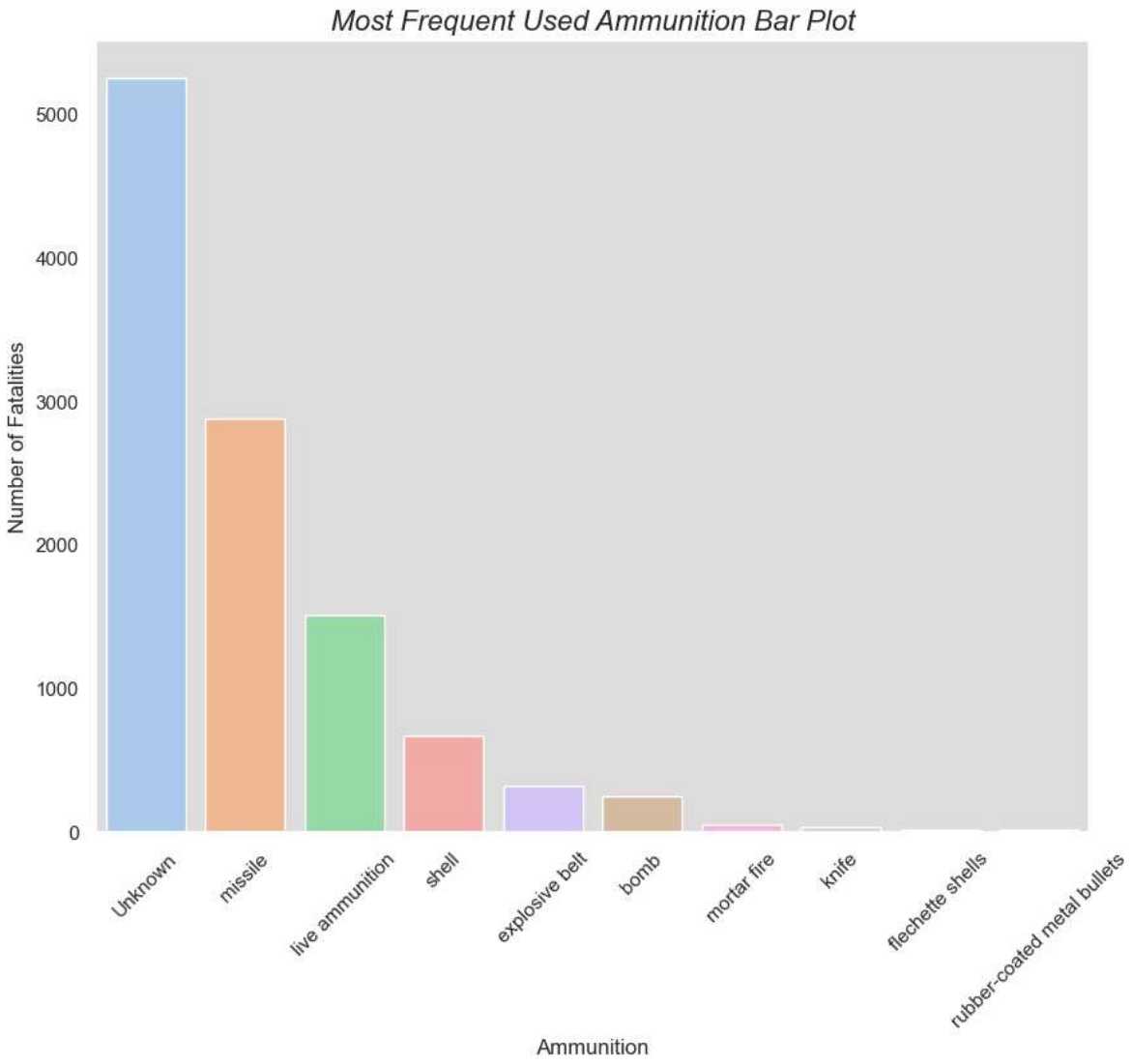
Based on this data, we can conclude that the most common weapon used in the Israel-Palestine conflict is the gun. Explosives are also a used as a weapon common.

Q-5

Analyze the ammunition and means by which the individuals were killed.

Determine most frequently used weapons or methods and evaluate their impact

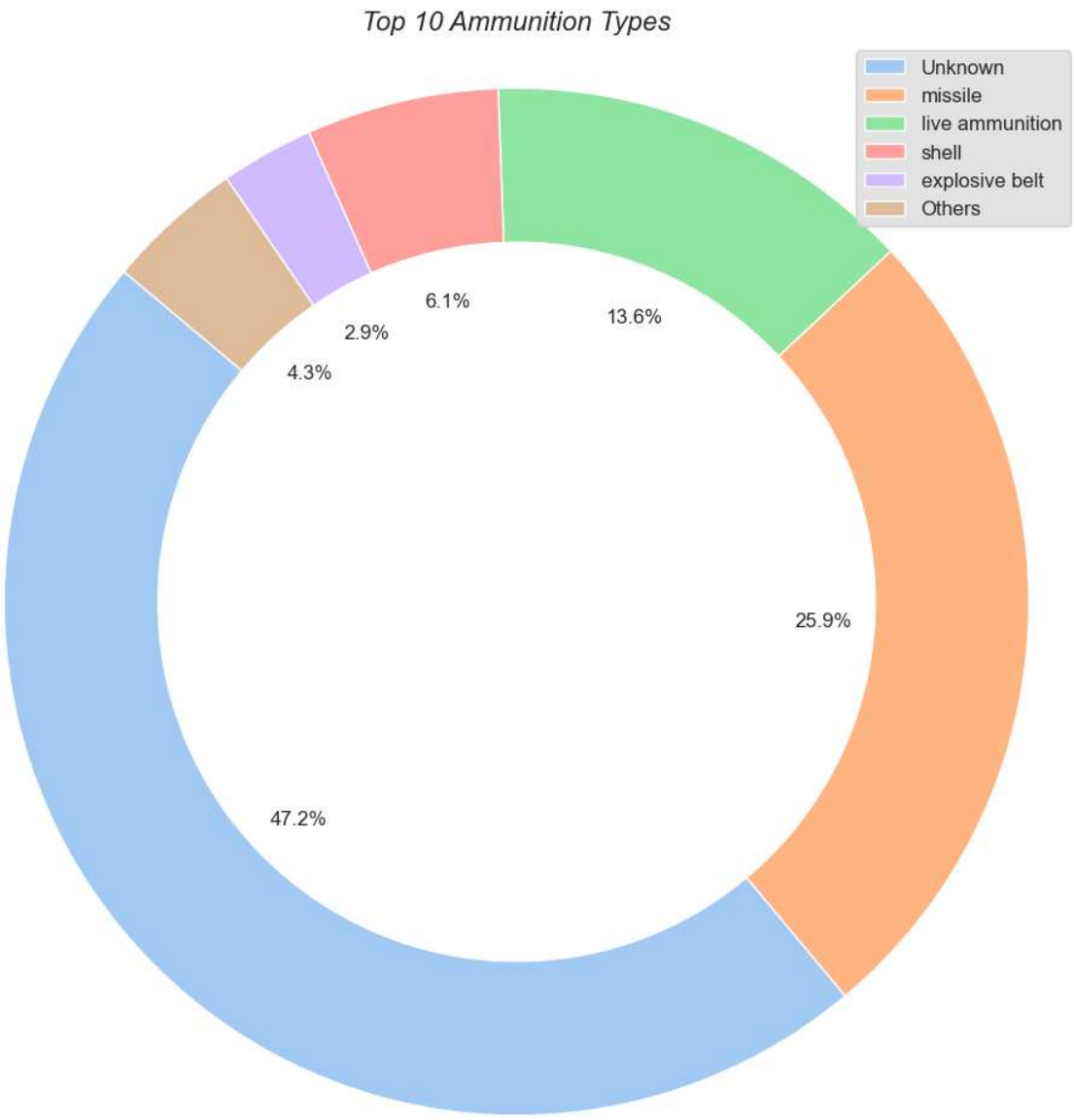
In [35]:
```
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd
from pandas.api.types import CategoricalDtype


ammunition_frequency = data['ammunition'].value_counts().head(10)

plt.figure(figsize=(10, 8))
sns.barplot(x=ammunition_frequency.index, y=ammunition_frequency.values)
plt.xlabel('Ammunition', fontsize=12)
plt.ylabel('Number of Fatalities', fontsize=12)
plt.title('Most Frequent Used Ammunition Bar Plot', fontsize=16, fontstyle='it
plt.xticks(rotation=45)
plt.grid(axis='y')
plt.show()

plt.clf()
```




```
<Figure size 1600x700 with 0 Axes>
```

The Above Bar Plot Analyze The Most Frequently used Type of Ammunation used in the conflict The Mostly Used weapon is not known however some other common weapons Missile live ammunition and shell After Analyzing above Chart We Conclude that various amunitions were employed, including missiles, live ammunition, and shells Which is Causing Harm

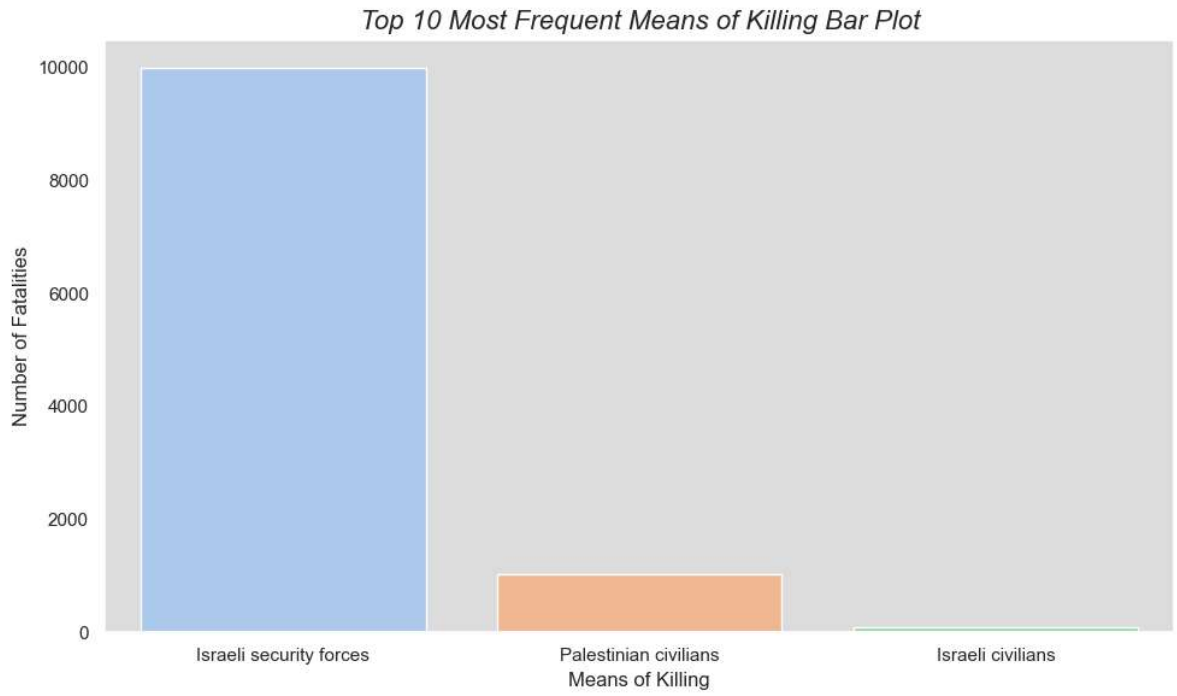
In [36]:
```
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd

ammunition_frequency = data['ammunition'].value_counts()

top_5_ammunition = ammunition_frequency[:5]

others_count = ammunition_frequency[5:].sum()


others_series = pd.Series({'Others': others_count})
updated_ammunition_counts = pd.concat([top_5_ammunition, others_series])

plt.figure(figsize=(12, 12))
wedges, texts, autotexts = plt.pie(updated_ammunition_counts, labels=updated_a
plt.title('Top 10 Ammunition Types', fontsize=16, fontstyle='italic')


for text in texts:
    text.set_visible(False)


plt.legend(wedges, updated_ammunition_counts.index, loc='best', fontsize='medi

centre_circle = plt.Circle((0, 0), 0.7, color='white', linewidth=0.8)
fig = plt.gcf()
fig.gca().add_artist(centre_circle)

plt.axis('equal')
plt.show()
```



The Donut Chart is Explaing the Most Frequently Used Type of Ammunation in conflict After Analyzing the Above Graph we Conclude That Mostly Used Ammunation is Unknown and in known weapons Missiles along with live ammunation is used for Fatilities

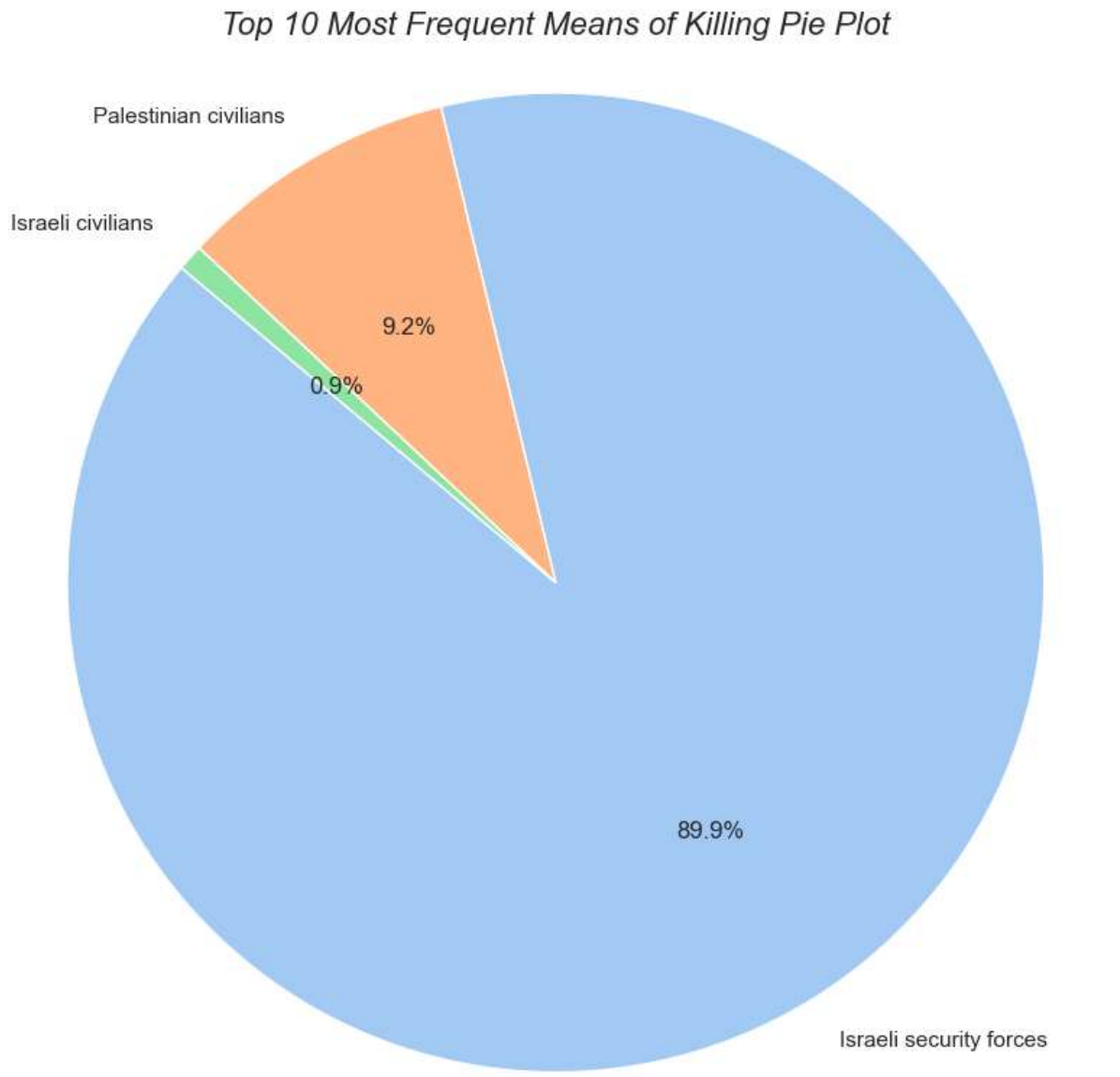
In [37]:

```
means_of_killing_frequency = data['killed_by'].value_counts().head(10)

plt.figure(figsize=(10, 6))
sns.barplot(x=means_of_killing_frequency.index, y=means_of_killing_frequency.v
plt.xlabel('Means of Killing', fontsize=12)
plt.ylabel('Number of Fatalities', fontsize=12)
plt.title('Top 10 Most Frequent Means of Killing Bar Plot', fontsize=16, fonts
plt.xticks()
plt.grid(axis='y')
plt.tight_layout()
plt.show()
```



The Above Bar Plot Explains the Killing of the people in conflict It is mostly done by the Israeli Security Forces some of them Is done by the Local civilians of the Palestine and few were Killed Israieli Civilians

In [38]:
```
plt.figure(figsize=(8, 8))
labels = means_of_killing_frequency.index
sizes = means_of_killing_frequency.values
colors = sns.color_palette('pastel')[0:len(labels)]


plt.pie(sizes, labels=labels, colors=colors, autopct='%1.1f%%', startangle=140
plt.title('Top 10 Most Frequent Means of Killing Pie Plot', fontsize=16, fonts

plt.axis('equal')
plt.tight_layout()
plt.show()
```



The Above Pie Plot Explains the Killing of the people in conflict Most of the causulities are done by the Israeli Security Forces Some Of them Is Done by the local civilians of the palestine

Q-6

Create profiles of the victims based on the available data such as age, gender, citizenship,

place of residence. Identify common characteristics among the victims.

In [39]:
```
import warnings
warnings.filterwarnings('ignore')
```

In [40]:

```
victim_profiles = data.groupby(['age', 'gender', 'citizenship', 'place_of_resi
victim_profiles.columns = ['Age', 'Gender', 'Citizenship', 'Place of Residence



fig = plt.figure(figsize=(12, 8))

ax1 = fig.add_subplot(2, 2, 1)
ax2 = fig.add_subplot(2, 2, 2)
ax3 = fig.add_subplot(2, 2, 3)
ax4 = fig.add_subplot(2, 2, 4)




sns.set(style="whitegrid")


ax1.bar(victim_profiles['Age'], victim_profiles['Frequency'], color='skyblue')
ax1.set_title('Age Distribution among Victims')
ax1.set_xlabel('Age')
ax1.set_ylabel('Number of People in 1000')

sns.barplot(x='Gender', y='Frequency', data=victim_profiles, ax=ax2, color='li
ax2.set_title('Gender Distribution among Victims')
ax2.set_xlabel('Gender')
ax2.set_ylabel('Number of People in 1000')
ax2.xaxis.grid(False)


sns.barplot(x='Citizenship', y='Frequency', data=victim_profiles, ax=ax3, colo
ax3.set_title('Citizenship Distribution among Victims')
ax3.set_xlabel('Citizenship')
ax3.set_ylabel('Number of People in 1000')
ax3.tick_params(axis='x', rotation=45)
ax3.xaxis.grid(False)

grouped_residence = victim_profiles.groupby('Place of Residence')['Frequency']


top_10_residence = grouped_residence.nlargest(10, 'Frequency')
ax4.bar(top_10_residence['Place of Residence'], top_10_residence['Frequency'],
ax4.set_title('Top 10 Place of Residence Distribution among Victims')
ax4.set_xlabel('Place of Residence')
ax4.set_ylabel('Number of People')
ax4.tick_params(axis='x', rotation=45)


plt.tight_layout()




plt.show()
```

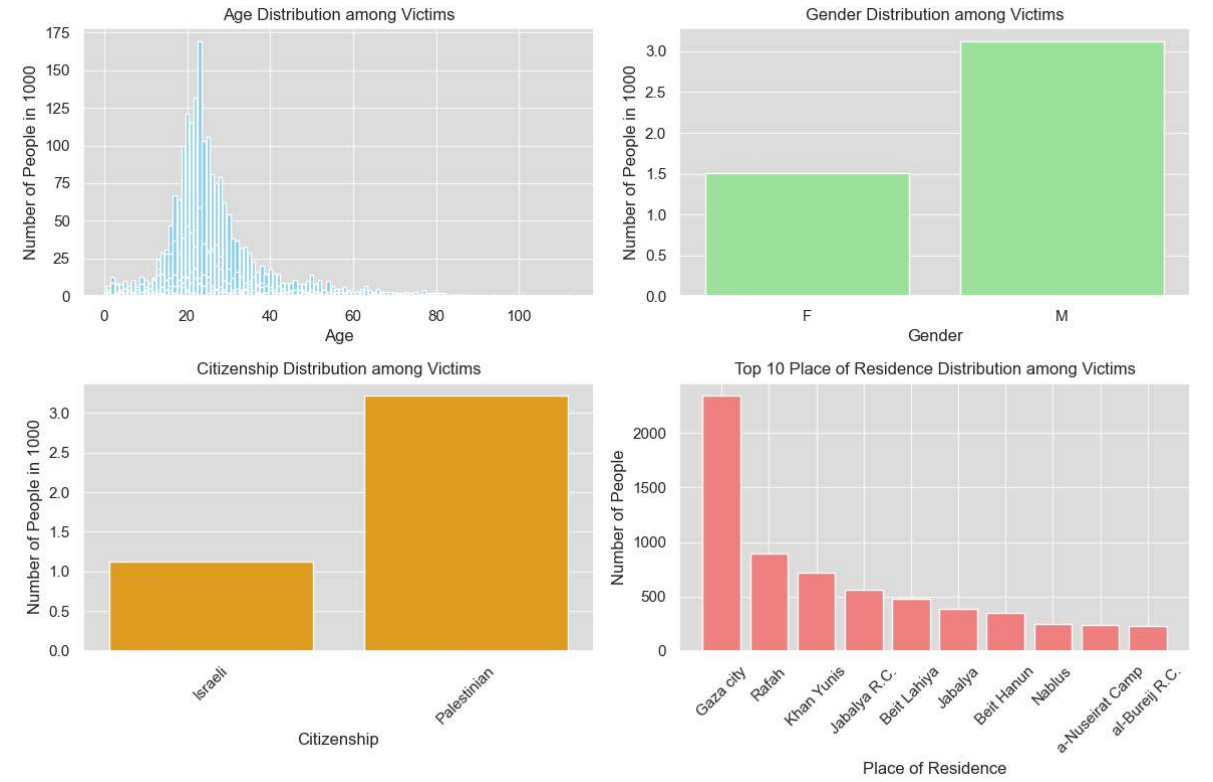


FIG-1

The Fig 1 Shows The Chart Consist of the Citizenship Distribution Among the Victims based on the Age

The Age is Divided into Bins from 0-19,20-39,40-59,60,69 and above 80

The Analysis Above Depicts mostly young generation include Childs , Teens and Adults Are Targeted

FIG-2

The Fig 2 Shows

Gender Distribtuion Among The Victims The Analysis in Fig 2 Depicts that mostly males are Killed in the confilct as compared to women Which are almost half as compared to man casualities

FIG-3 The Fig 3 Shows CitizenShip Among the Victims After Performing Data Analysis The Results Conclude That the Palestine peple are mostly killed which some people israel are killed in the conflict The palestanian are fatatilities almost 3 times more as compared to the irael people

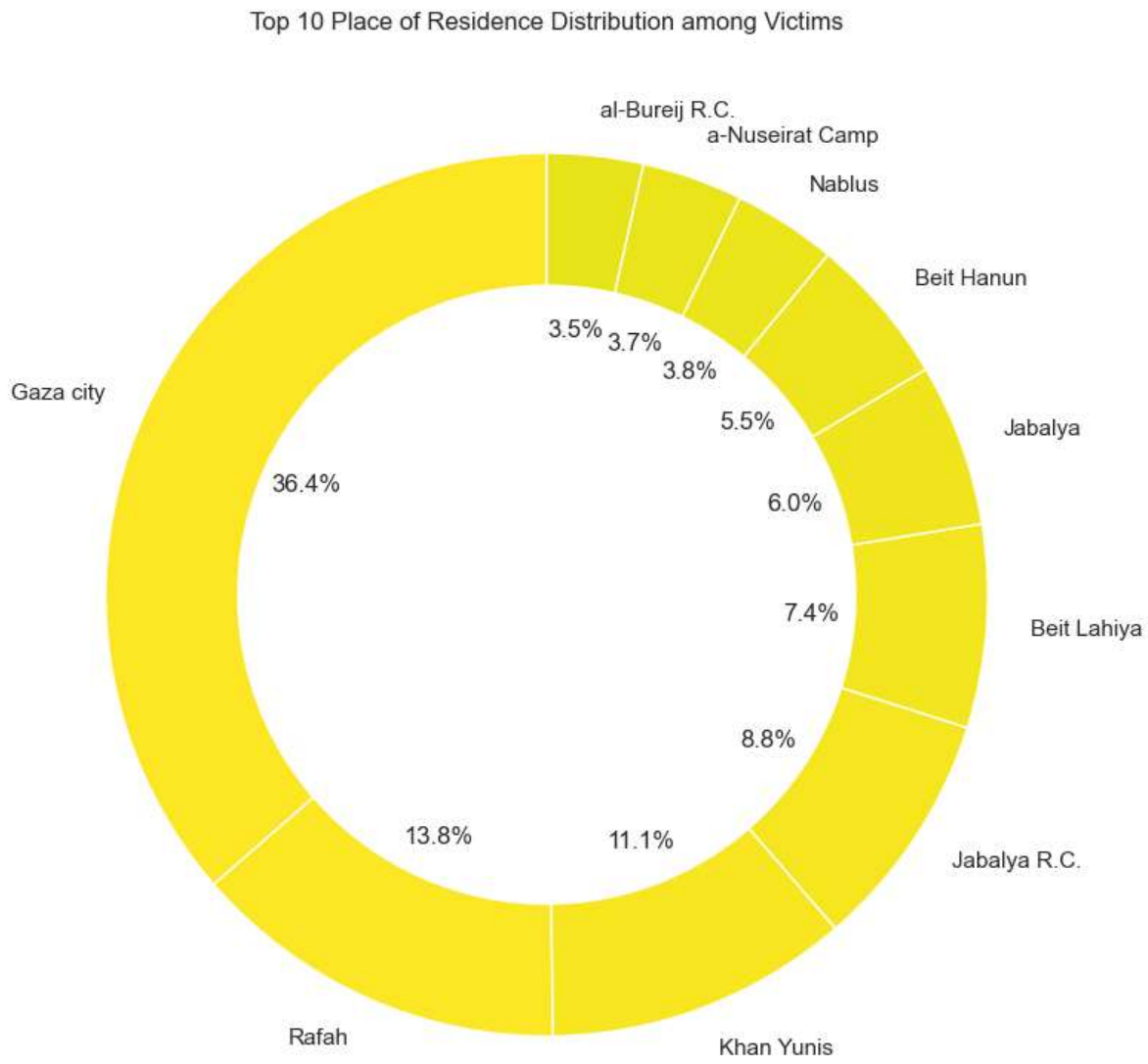
FIG-4

The Fig 4 Shows place of residence among the Victims majority of the People Belong From Gaza and Rafah City and some other cities

In [41]:
```
import warnings
warnings.filterwarnings('ignore')
```

In [42]:

```
plt.figure(figsize=(8, 8))
sizes = top_10_residence['Frequency']
labels = top_10_residence['Place of Residence']
colors = plt.cm.viridis_r(range(len(labels)))

plt.pie(sizes, labels=labels, colors=colors, wedgeprops=dict(width=0.3), autop
plt.title('Top 10 Place of Residence Distribution among Victims')


centre_circle = plt.Circle((0, 0), 0.5, color='white', edgecolor='black', line
plt.gca().add_artist(centre_circle)
sns.color_palette("pastel")



plt.axis('equal')
plt.tight_layout()
plt.show()
```



The Above Donut Chart Depicts the top 10 places of residence distribution among the victims The Data Depicts The Most Of the Victims Belong From Gaza City and Some Majority from Rafah and other areas of Palestine

In [ ]: