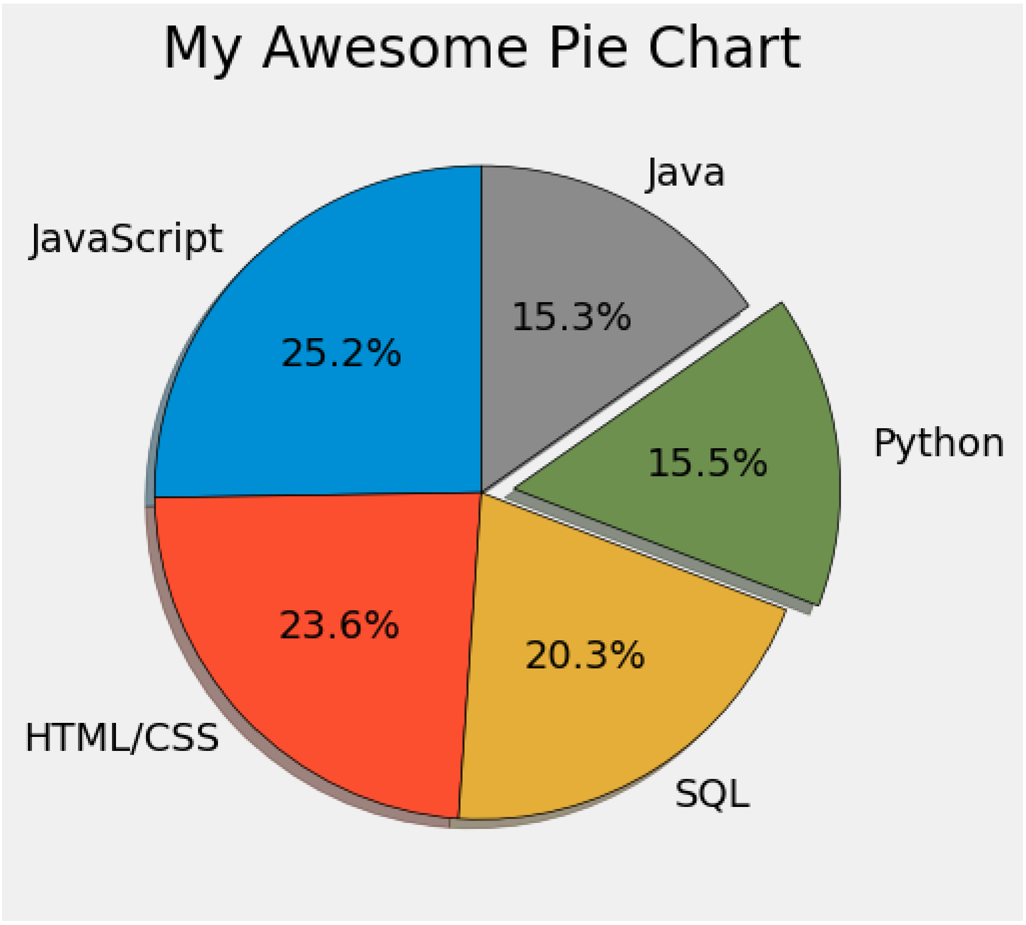
MatplotLib Pie Chart Implementation

In [ ]:
```
# Import Library
from matplotlib import pyplot as plt
```

In [ ]:
```
# for style there are several other styles as well
plt.style.use("fivethirtyeight")
```

In [ ]:
```
# Data
slices = [59219, 55466, 47544, 36443, 35917]
labels = ['JavaScript', 'HTML/CSS', 'SQL', 'Python', 'Java']
explode = [0, 0, 0, 0.1, 0]
```

In [ ]:
```
# Pie Plot function
plt.pie(slices, labels=labels, explode=explode, shadow=True,
        startangle=90, autopct='%1.1f%%',
        wedgeprops={'edgecolor': 'black'})
#Plot
plt.title("My Awesome Pie Chart")
plt.tight_layout()
plt.show()
```



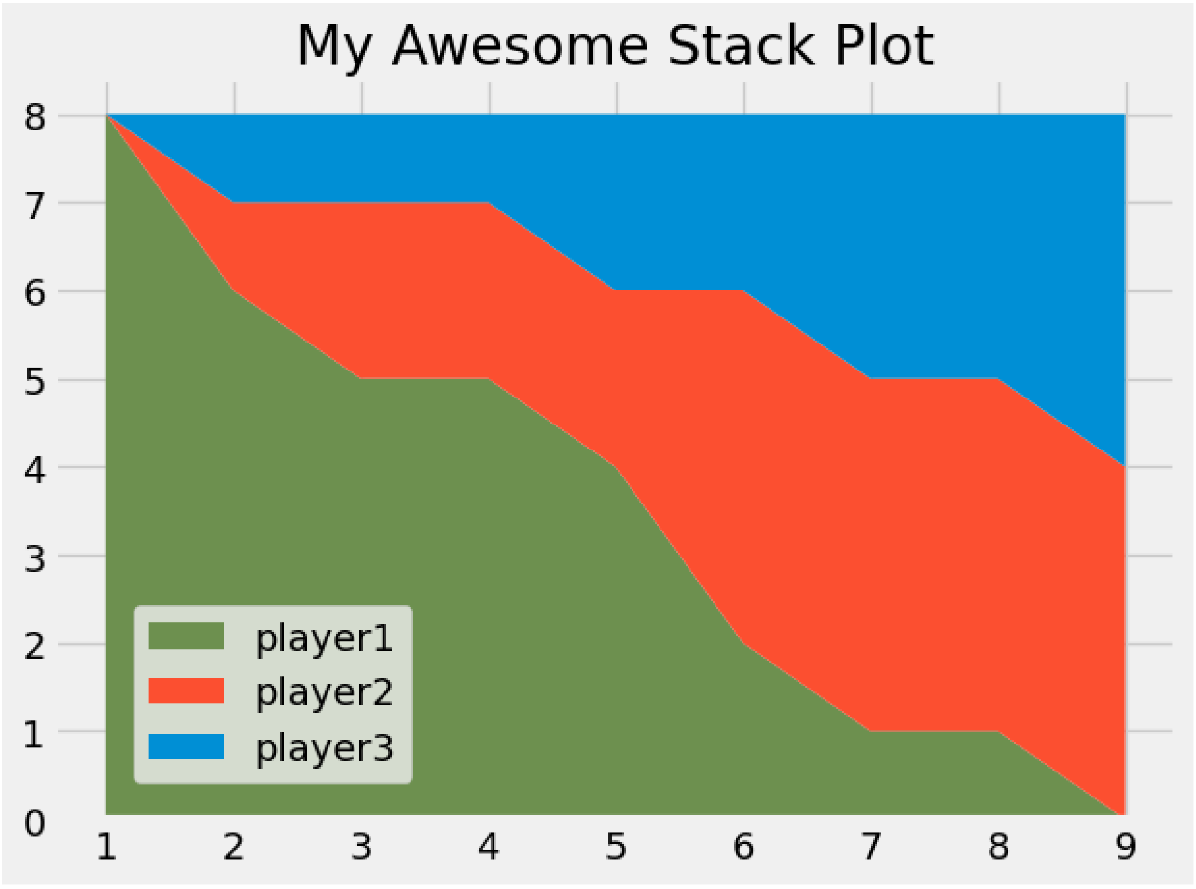
Implementation of Stack Plots

In [ ]:
```
# Import Library
from matplotlib import pyplot as plt
```

In [ ]:
```
# Style Sheet
plt.style.use("fivethirtyeight")
```

In [ ]:
```
# Data
minutes = [1, 2, 3, 4, 5, 6, 7, 8, 9]

player1 = [8, 6, 5, 5, 4, 2, 1, 1, 0]
player2 = [0, 1, 2, 2, 2, 4, 4, 4, 4]
player3 = [0, 1, 1, 1, 2, 2, 3, 3, 4]

labels = ['player1', 'player2', 'player3']
colors = ['#6d904f', '#fc4f30', '#008fd5']
```

In [ ]:
```
# Plotting
plt.stackplot(minutes, player1, player2, player3, labels=labels, colors=colors)
plt.legend(loc=(0.07, 0.05))

plt.title("My Awesome Stack Plot")
plt.tight_layout()
plt.show()
```



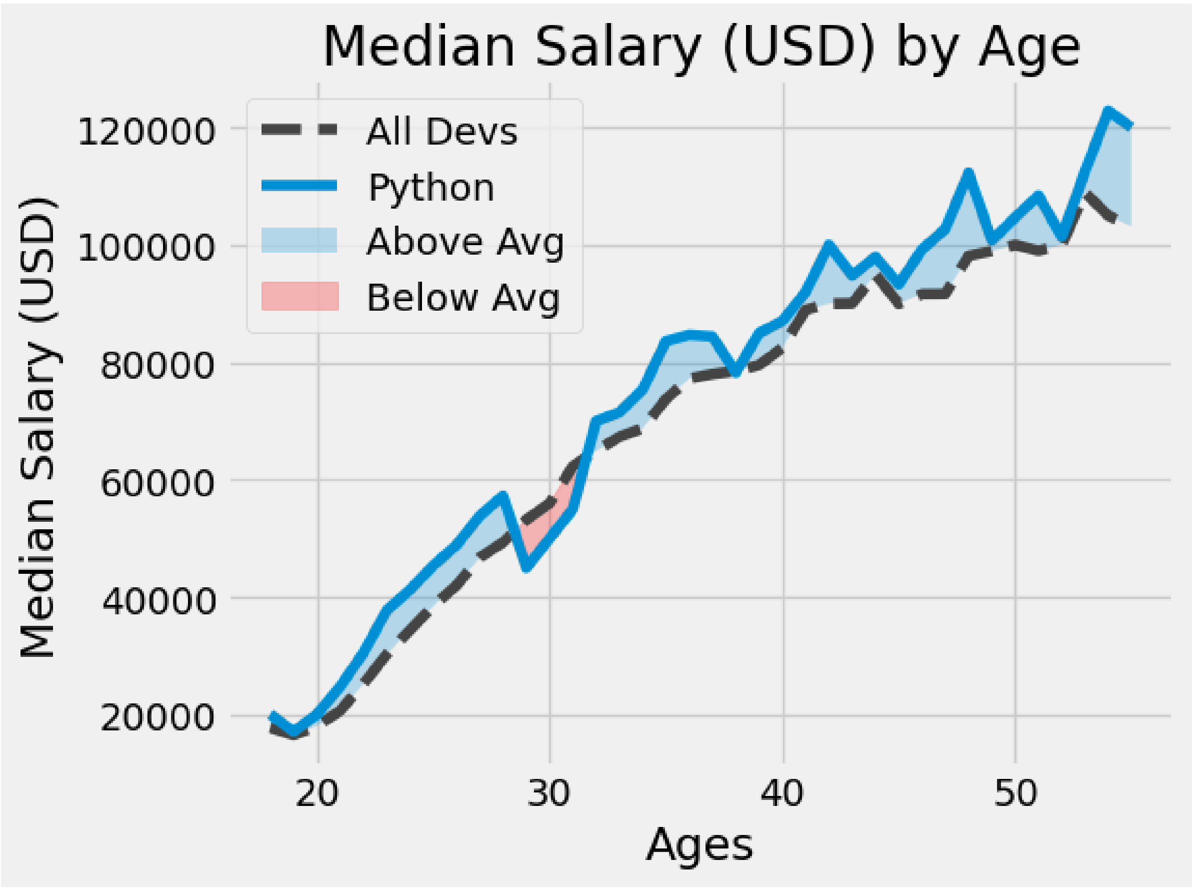
Implementation of Line Area Plot

In [ ]:
```
# Import Library
import pandas as pd
from matplotlib import pyplot as plt
```

In [ ]:
```
# Get Data
data = pd.read_csv('data.csv')
ages = data['Age']
dev_salaries = data['All_Devs']
```

```
py_salaries = data['Python']
js_salaries = data['JavaScript']
```

In [ ]:
```
# Plot
plt.plot(ages, dev_salaries, color='#444444',
         linestyle='--', label='All Devs')

plt.plot(ages, py_salaries, label='Python')
overall_median = 57287

plt.fill_between(ages, py_salaries, dev_salaries,
                 where=(py_salaries > dev_salaries),
                 interpolate=True, alpha=0.25, label='Above Avg')

plt.fill_between(ages, py_salaries, dev_salaries,
                 where=(py_salaries <= dev_salaries),
                 interpolate=True, color='red', alpha=0.25, label='Below Avg')

plt.legend()

plt.title('Median Salary (USD) by Age')
plt.xlabel('Ages')
plt.ylabel('Median Salary (USD)')

plt.tight_layout()

plt.show()
```



Implementation of Histogram

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

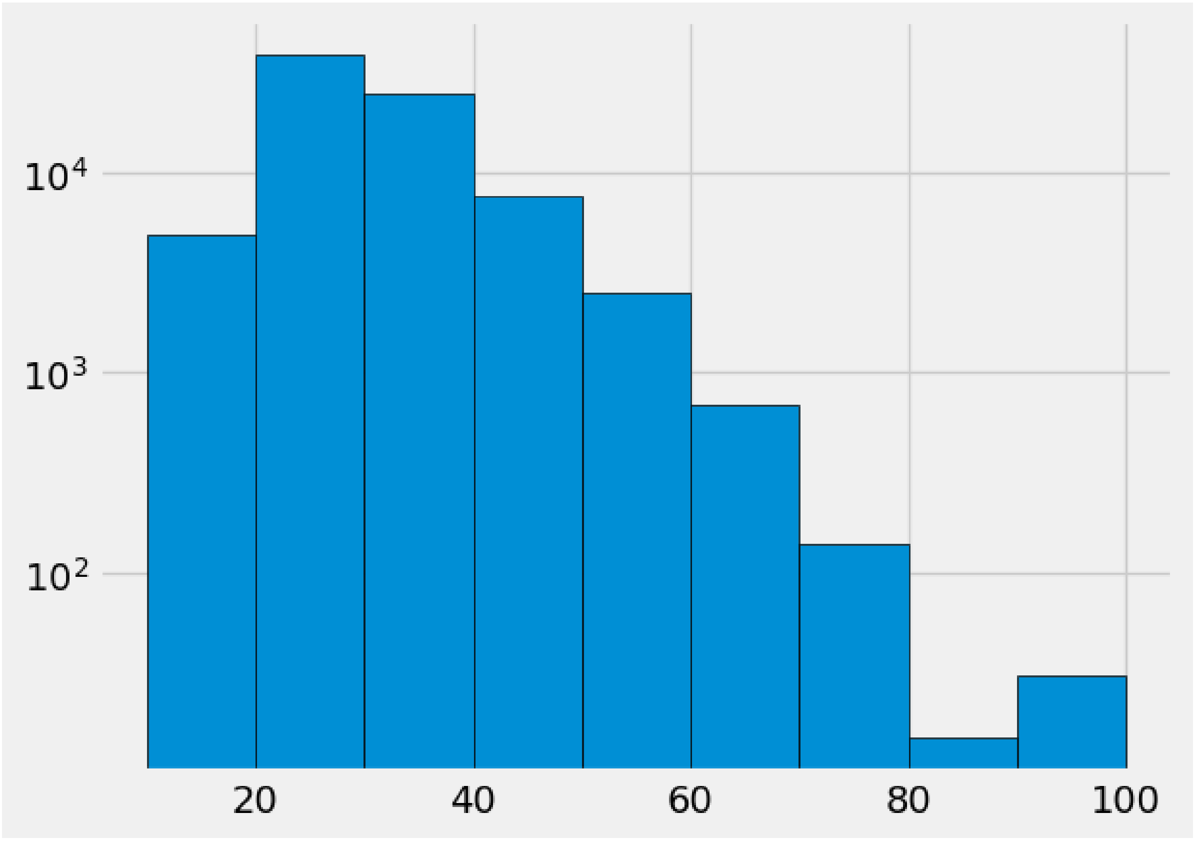
```
plt.style.use('fivethirtyeight')

data = pd.read_csv('data1.csv')
```

In [ ]:

```
ids = data['Responder_id']
ages = data['Age']
```

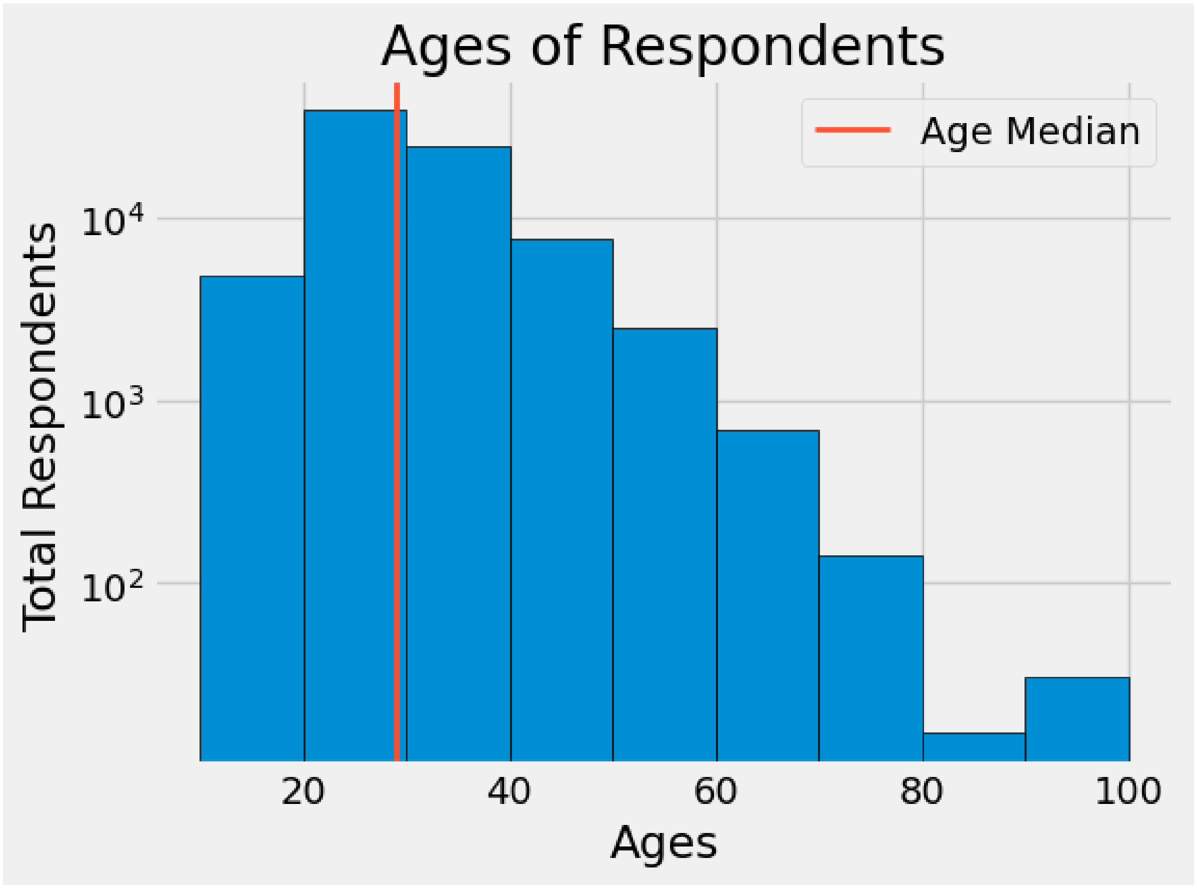
In [ ]:

```
bins = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100]

plt.hist(ages, bins=bins, edgecolor='black', log=True)

median_age = 29
color = '#fc4f30'
```



In [ ]:

```
plt.hist(ages, bins=bins, edgecolor='black', log=True)
plt.axvline(median_age, color=color, label='Age Median', linewidth=2)

plt.legend()

plt.title('Ages of Respondents')
plt.xlabel('Ages')
plt.ylabel('Total Respondents')

plt.tight_layout()

plt.show()
```

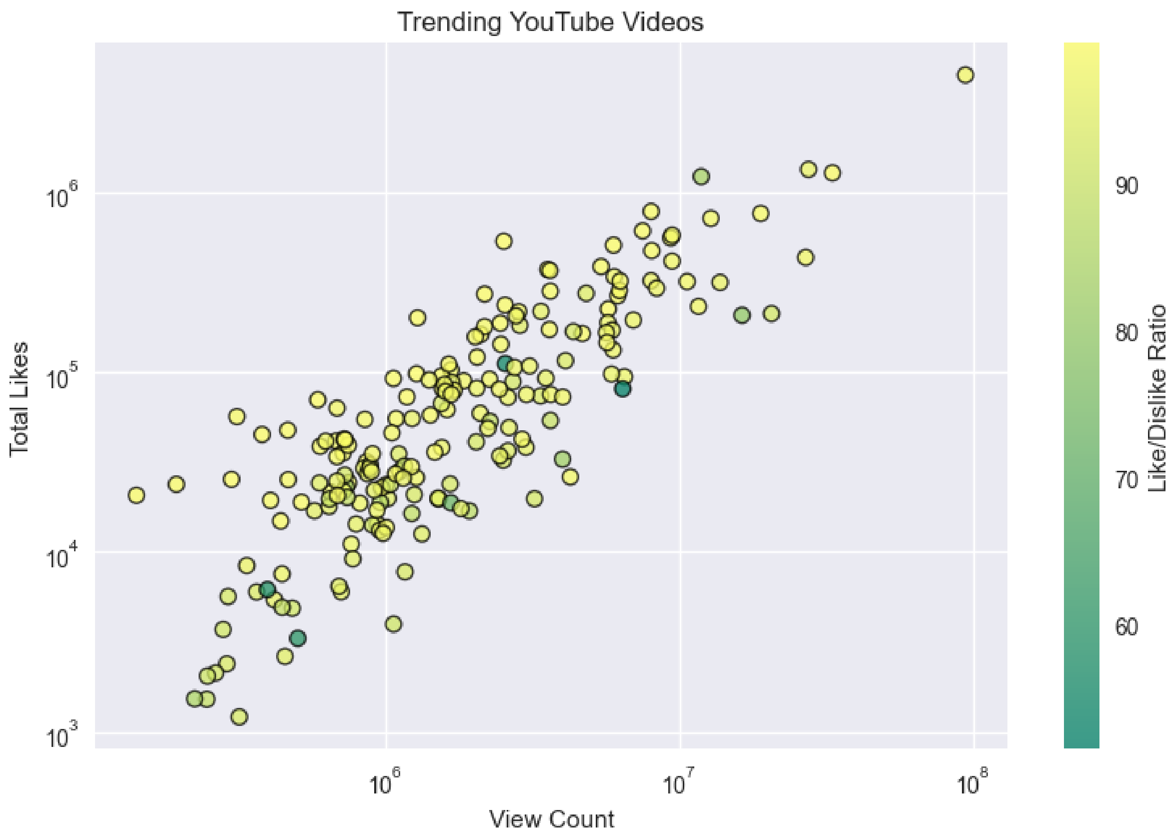


Implementation of Scatter Plot

In [ ]:
```
import pandas as pd
from matplotlib import pyplot as plt
plt.style.use('seaborn')

data = pd.read_excel('data5.xlsx')
```

```
C:\Users\anand\AppData\Local\Temp\ipykernel_14096\3893117574.py:3: MatplotlibDeprecationWarning: The seaborn styles shipped by Matplotlib are deprecated since 3.6, as they no longer correspond to the styles shipped by seaborn. However, they will remain available as 'seaborn-v0_8-<style>'. Alternatively, directly use the seaborn API instead.
  plt.style.use('seaborn')
```

In [ ]:
```
view_count = data['view_count']
likes = data['likes']
ratio = data['ratio']

plt.scatter(view_count, likes, c=ratio, cmap='summer',
            edgecolor='black', linewidth=1, alpha=0.75)
cbar = plt.colorbar()
cbar.set_label('Like/Dislike Ratio')

plt.xscale('log')
plt.yscale('log')

plt.title('Trending YouTube Videos')
plt.xlabel('View Count')
plt.ylabel('Total Likes')

plt.tight_layout()
plt.show()
```

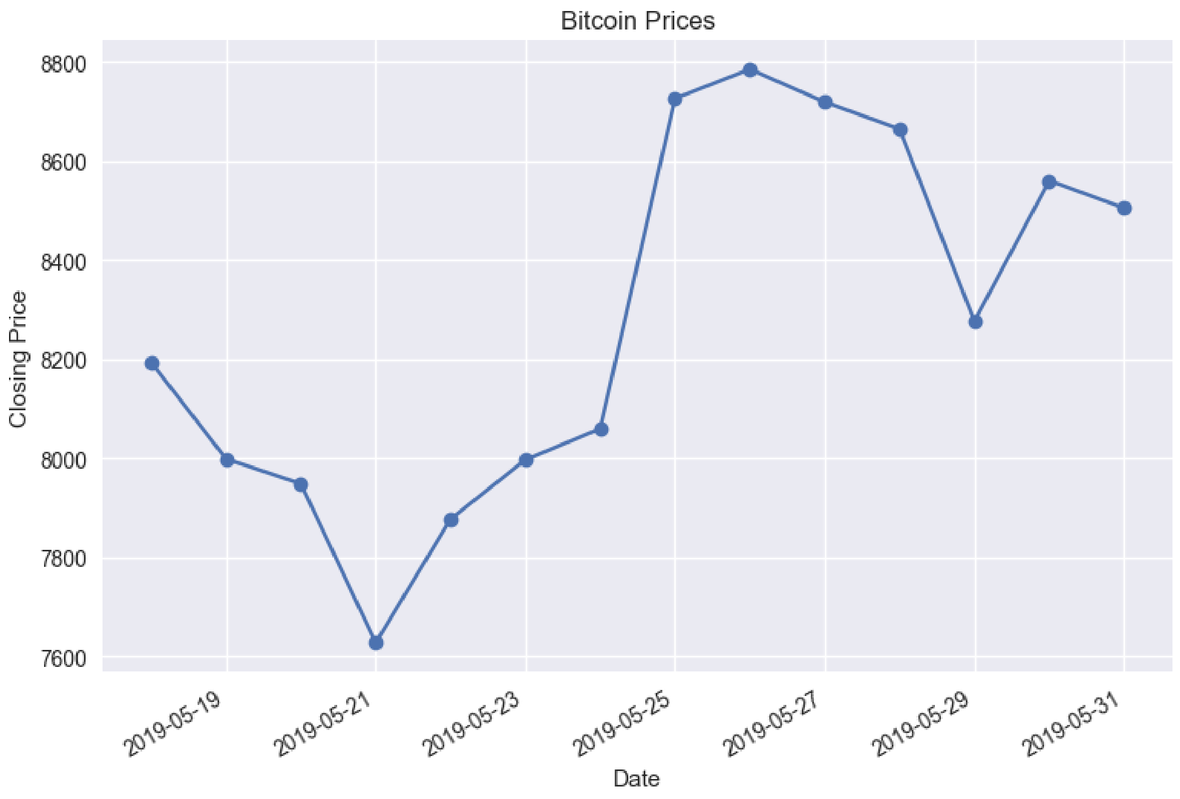


Plotting time series data

In [ ]:
```
import pandas as pd
from datetime import datetime, timedelta
from matplotlib import pyplot as plt
from matplotlib import dates as mpl_dates
```

In [ ]:
```
data = pd.read_excel('data6.xlsx')
```

In [ ]:
```
data['Date'] = pd.to_datetime(data['Date'])
data.sort_values('Date', inplace=True)

price_date = data['Date']
price_close = data['Close']

plt.plot_date(price_date, price_close, linestyle='solid')

plt.gcf().autofmt_xdate()

plt.title('Bitcoin Prices')
plt.xlabel('Date')
plt.ylabel('Closing Price')

plt.tight_layout()

plt.show()
```



Implement Real Time data plot using MATPLOTLIB

In [ ]:
```
import csv
import random
import time

x_value = 0
total_1 = 1000
total_2 = 1000

fieldnames = ["x_value", "total_1", "total_2"]


with open('data.csv', 'w') as csv_file:
    csv_writer = csv.DictWriter(csv_file, fieldnames=fieldnames)
    csv_writer.writeheader()

while True:

    with open('data.csv', 'a') as csv_file:
        csv_writer = csv.DictWriter(csv_file, fieldnames=fieldnames)

        info = {
            "x_value": x_value,
            "total_1": total_1,
            "total_2": total_2
        }

        csv_writer.writerow(info)
        print(x_value, total_1, total_2)

        x_value += 1
        total_1 = total_1 + random.randint(-6, 8)
        total_2 = total_2 + random.randint(-5, 6)
    time.sleep(1)
```

```
0 1000 1000
1 1008 995
2 1002 996
3 999 991
4 1001 990
5 996 993
6 999 988
7 994 992
8 994 987
9 989 991
10 985 989
11 990 994
12 988 989
13 989 986
14 984 984
15 990 986
16 986 986
17 986 981
18 990 986
19 994 986
20 989 992
21 991 998
22 985 1004
23 980 1008
24 983 1006
25 990 1006
26 990 1007
27 984 1004
28 980 1007
29 981 1013
30 985 1016
31 986 1021
32 987 1026
33 989 1021
34 984 1019
35 992 1023
36 986 1021
37 987 1021
38 986 1020
39 984 1017
```

```
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
Cell In[64], line 34
     31     total_1 = total_1 + random.randint(-6, 8)
     32     total_2 = total_2 + random.randint(-5, 6)
---> 34 time.sleep(1)

KeyboardInterrupt:
```

In [ ]:

```
import random
from itertools import count
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib.animation import FuncAnimation

plt.style.use('fivethirtyeight')

x_vals = []
y_vals = []
```

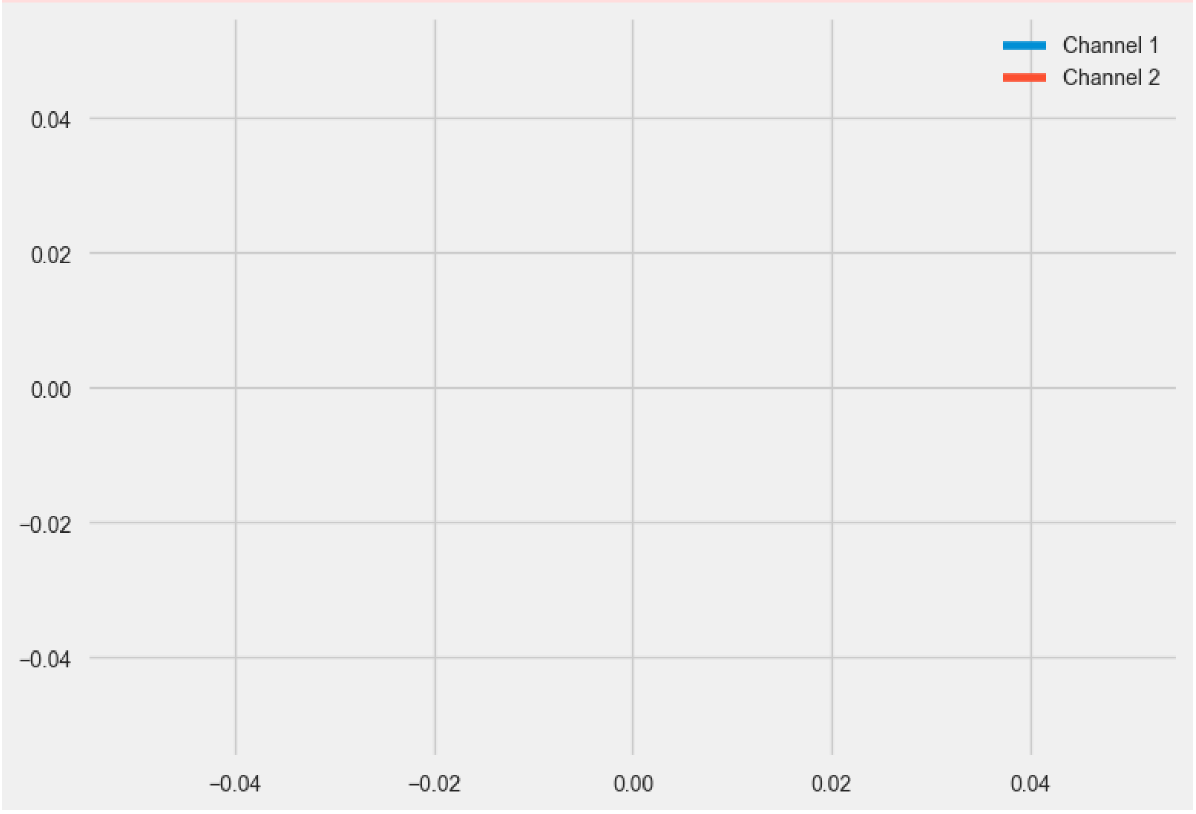
```python
index = count()


def animate(i):
    data = pd.read_csv('data.csv')
    x = data['x_value']
    y1 = data['total_1']
    y2 = data['total_2']

    plt.cla()

    plt.plot(x, y1, label='Channel 1')
    plt.plot(x, y2, label='Channel 2')

    plt.legend(loc='upper left')
    plt.tight_layout()


ani = FuncAnimation(plt.gcf(), animate, interval=1000)

plt.tight_layout()
plt.show()
```

```
C:\Users\anand\AppData\Local\Temp\ipykernel_14096\2077380327.py:30: UserWarning: fra
mes=None which we can infer the length of, did not pass an explicit *save_count* and
passed cache_frame_data=True.  To avoid a possibly unbounded cache, frame data cachi
ng has been disabled. To suppress this warning either pass `cache_frame_data=False`
or `save_count=MAX_FRAMES`.
  ani = FuncAnimation(plt.gcf(), animate, interval=1000)
c:\Users\anand\anaconda3\envs\dev1\Lib\site-packages\matplotlib\animation.py:884: Us
erWarning: Animation was deleted without rendering anything. This is most likely not
intended. To prevent deletion, assign the Animation to a variable, e.g. `anim`, that
exists until you output the Animation using `plt.show()` or `anim.save()`.
  warnings.warn(
<Figure size 800x550 with 0 Axes>
```

In [ ]:

```python
# Another way to do it without clearing the Axis
from itertools import count
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib.animation import FuncAnimation

plt.style.use('fivethirtyeight')

x_vals = []
y_vals = []

plt.plot([], [], label='Channel 1')
plt.plot([], [], label='Channel 2')


def animate(i):
    data = pd.read_csv('data.csv')
    x = data['x_value']
    y1 = data['total_1']
    y2 = data['total_2']

    ax = plt.gca()
    line1, line2 = ax.lines

    line1.set_data(x, y1)
    line2.set_data(x, y2)

    xlim_low, xlim_high = ax.get_xlim()
    ylim_low, ylim_high = ax.get_ylim()

    ax.set_xlim(xlim_low, (x.max() + 5))

    y1max = y1.max()
    y2max = y2.max()
    current_ymax = y1max if (y1max > y2max) else y2max

    y1min = y1.min()
    y2min = y2.min()
    current_ymin = y1min if (y1min < y2min) else y2min

    ax.set_ylim((current_ymin - 5), (current_ymax + 5))


ani = FuncAnimation(plt.gcf(), animate, interval=1000)

plt.legend()
plt.tight_layout()
plt.show()
```

```
C:\Users\anand\AppData\Local\Temp\ipykernel_14096\4154212094.py:44: UserWarning: fra
mes=None which we can infer the length of, did not pass an explicit *save_count* and
passed cache_frame_data=True.  To avoid a possibly unbounded cache, frame data cachi
ng has been disabled. To suppress this warning either pass `cache_frame_data=False`
or `save_count=MAX_FRAMES`.
  ani = FuncAnimation(plt.gcf(), animate, interval=1000)
c:\Users\anand\anaconda3\envs\dev1\Lib\site-packages\matplotlib\animation.py:884: Us
erWarning: Animation was deleted without rendering anything. This is most likely not
intended. To prevent deletion, assign the Animation to a variable, e.g. `anim`, that
exists until you output the Animation using `plt.show()` or `anim.save()`.
  warnings.warn(
```



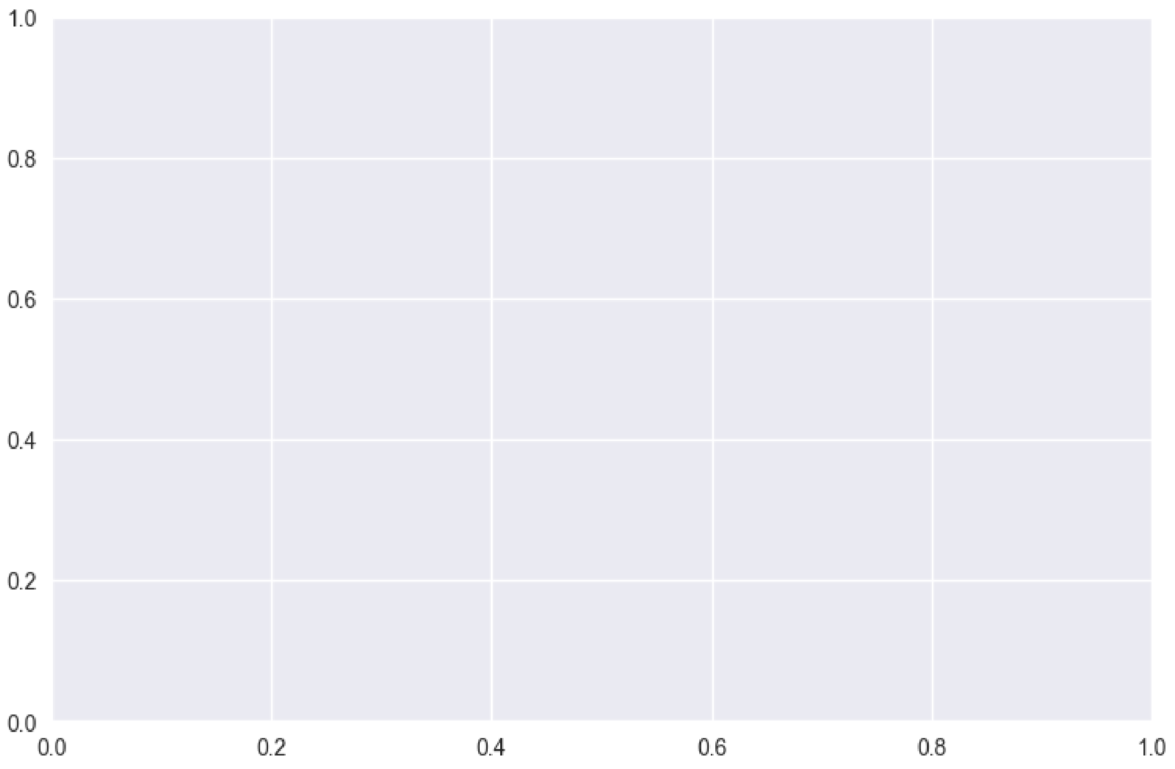
Using Subplots - MATPLOTLIB

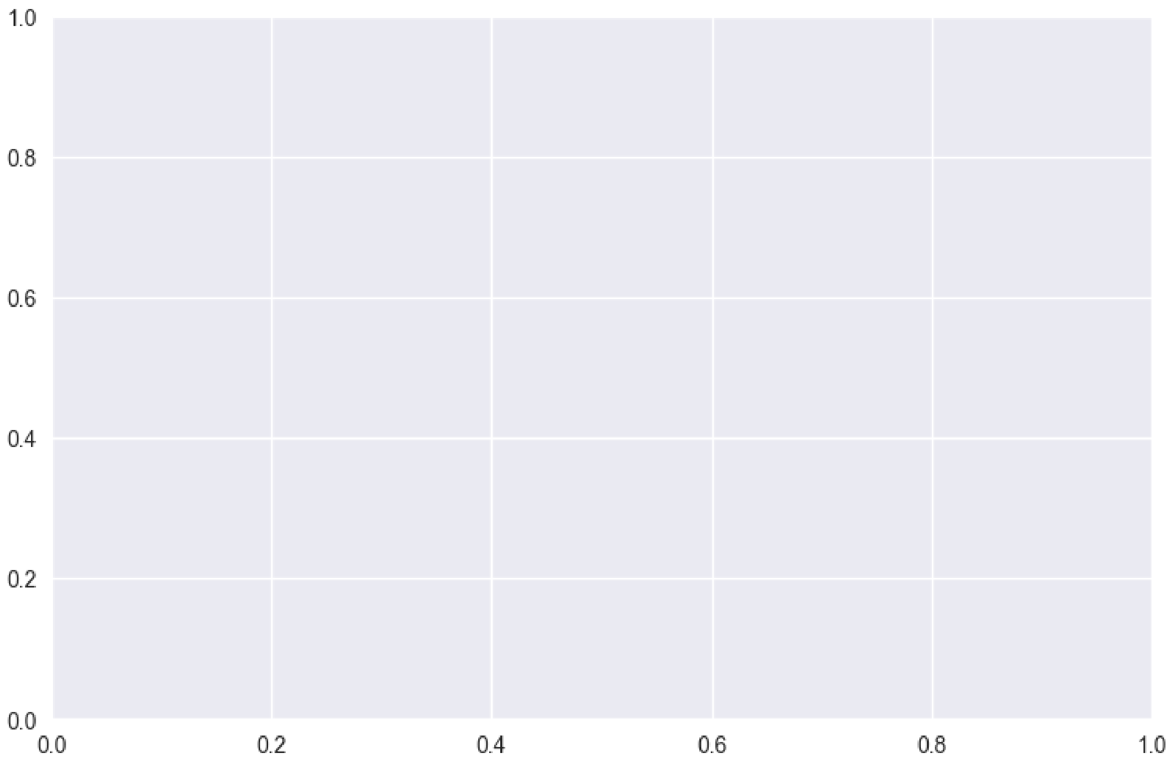
In [ ]:
```
import pandas as pd
from matplotlib import pyplot as plt
```

```
data = pd.read_excel('data7.xlsx')
```

In [ ]:
```
ages = data['Age']
dev_salaries = data['All_Devs']
py_salaries = data['Python']
js_salaries = data['JavaScript']
```

In [ ]:
```
fig1, ax1 = plt.subplots()
fig2, ax2 = plt.subplots()
```





In [ ]:
```
ax1.plot(ages, dev_salaries, color='#444444',
         linestyle='--', label='All Devs')

ax2.plot(ages, py_salaries, label='Python')
```

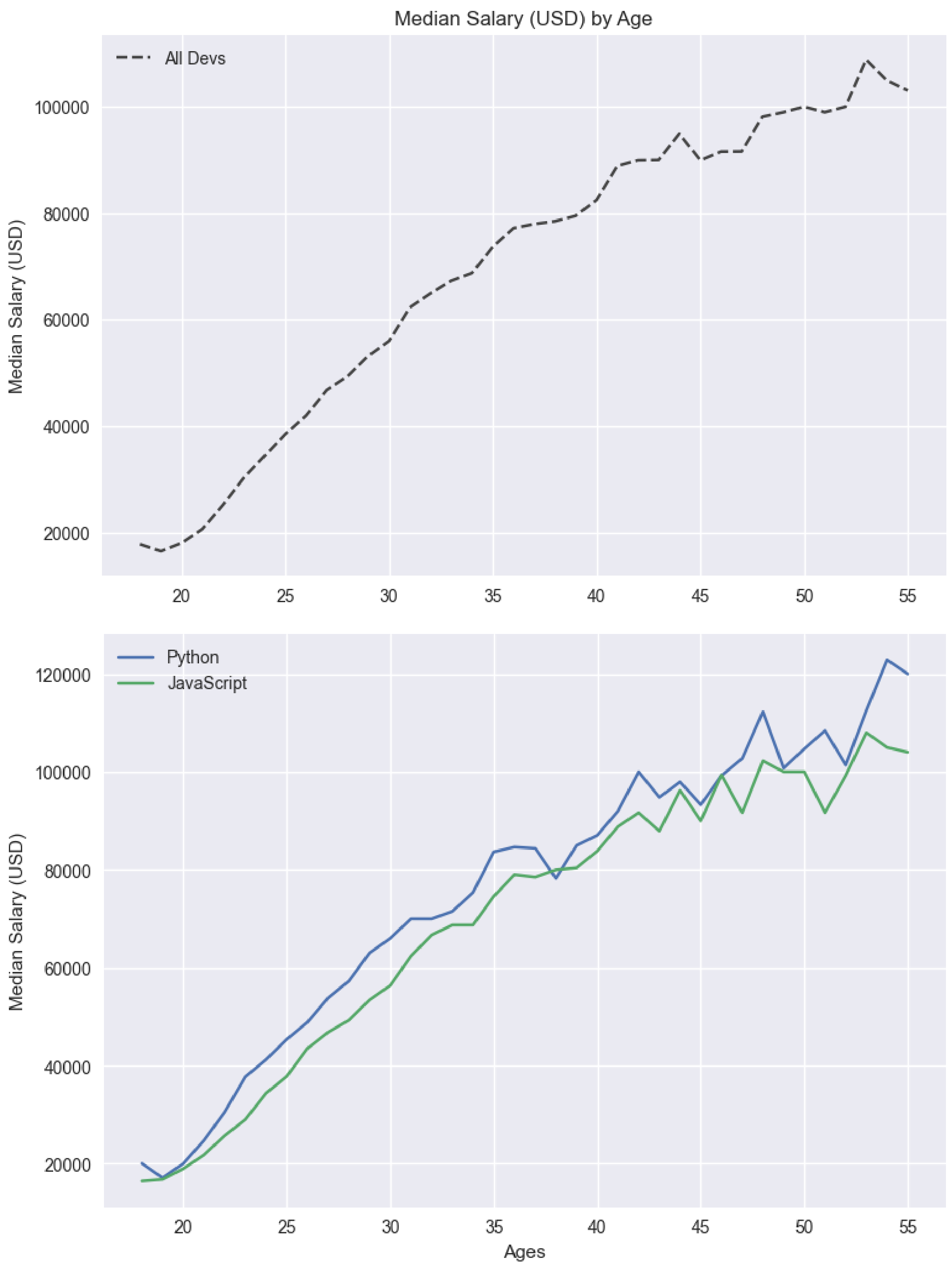
```
ax2.plot(ages, js_salaries, label='JavaScript')

ax1.legend()
ax1.set_title('Median Salary (USD) by Age')
ax1.set_ylabel('Median Salary (USD)')

ax2.legend()
ax2.set_xlabel('Ages')
ax2.set_ylabel('Median Salary (USD)')

plt.tight_layout()

plt.show()

fig1.savefig('fig1.png')
fig2.savefig('fig2.png')
```

```
<Figure size 800x550 with 0 Axes>
```

In [ ]:

```
import pandas as pd
from matplotlib import pyplot as plt

plt.style.use('seaborn')

data = pd.read_excel('data7.xlsx')
ages = data['Age']
dev_salaries = data['All_Devs']
py_salaries = data['Python']
js_salaries = data['JavaScript']

fig1, ax1 = plt.subplots()
fig2, ax2 = plt.subplots()

ax1.plot(ages, dev_salaries, color='#444444',
         linestyle='--', label='All Devs')

ax2.plot(ages, py_salaries, label='Python')
ax2.plot(ages, js_salaries, label='JavaScript')

ax1.legend()
ax1.set_title('Median Salary (USD) by Age')
ax1.set_ylabel('Median Salary (USD)')

ax2.legend()
ax2.set_xlabel('Ages')
ax2.set_ylabel('Median Salary (USD)')

plt.tight_layout()

plt.show()

fig1.savefig('fig1.png')
fig2.savefig('fig2.png')
```

```
C:\Users\anand\AppData\Local\Temp\ipykernel_14096\253559789.py:4: MatplotlibDeprecat
ionWarning: The seaborn styles shipped by Matplotlib are deprecated since 3.6, as th
ey no longer correspond to the styles shipped by seaborn. However, they will remain
available as 'seaborn-v0_8-<style>'. Alternatively, directly use the seaborn API ins
tead.
  plt.style.use('seaborn')
```


Median Salary (USD) by Age
All Devs
Median Salary (USD)
Python
JavaScript
Median Salary (USD)
Ages

In [ ]: