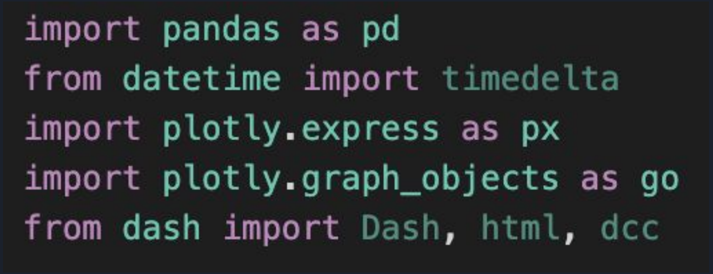
Analyzing Netflix Data

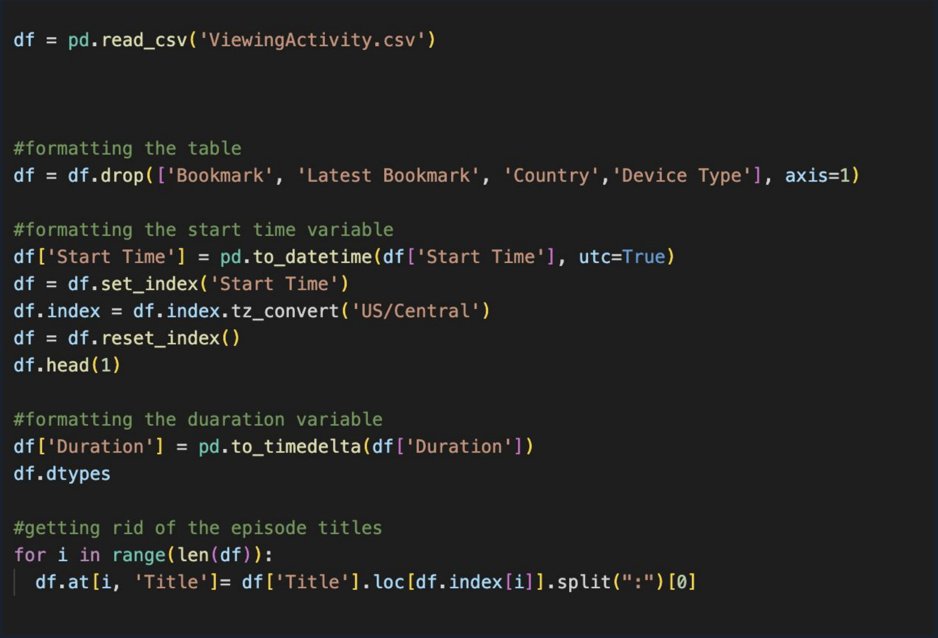
Gabe Meier and Spencer French

Background information

- Data requested from Netflix
- Packages
- Formatting the data



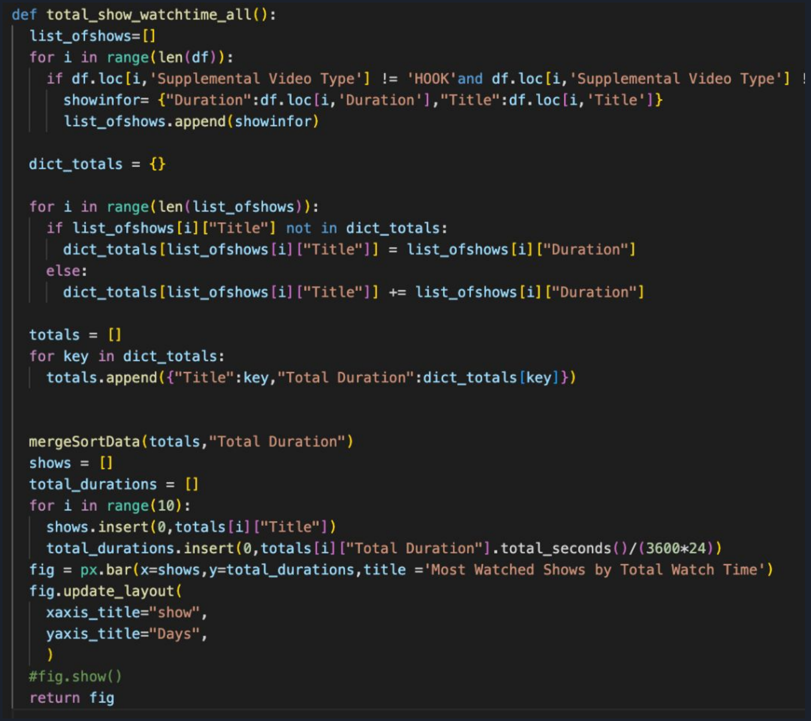
```python
import pandas as pd
from datetime import timedelta
import plotly.express as px
import plotly.graph_objects as go
from dash import Dash, html, dcc
```



```python
df = pd.read_csv('ViewingActivity.csv')



#formatting the table
df = df.drop(['Bookmark', 'Latest Bookmark', 'Country','Device Type'], axis=1)

#formatting the start time variable
df['Start Time'] = pd.to_datetime(df['Start Time'], utc=True)
df = df.set_index('Start Time')
df.index = df.index.tz_convert('US/Central')
df = df.reset_index()
df.head(1)

#formatting the duaration variable
df['Duration'] = pd.to_timedelta(df['Duration'])
df.dtypes

#getting rid of the episode titles
for i in range(len(df)):
    df.at[i, 'Title']= df['Title'].loc[df.index[i]].split(":")[0]
```

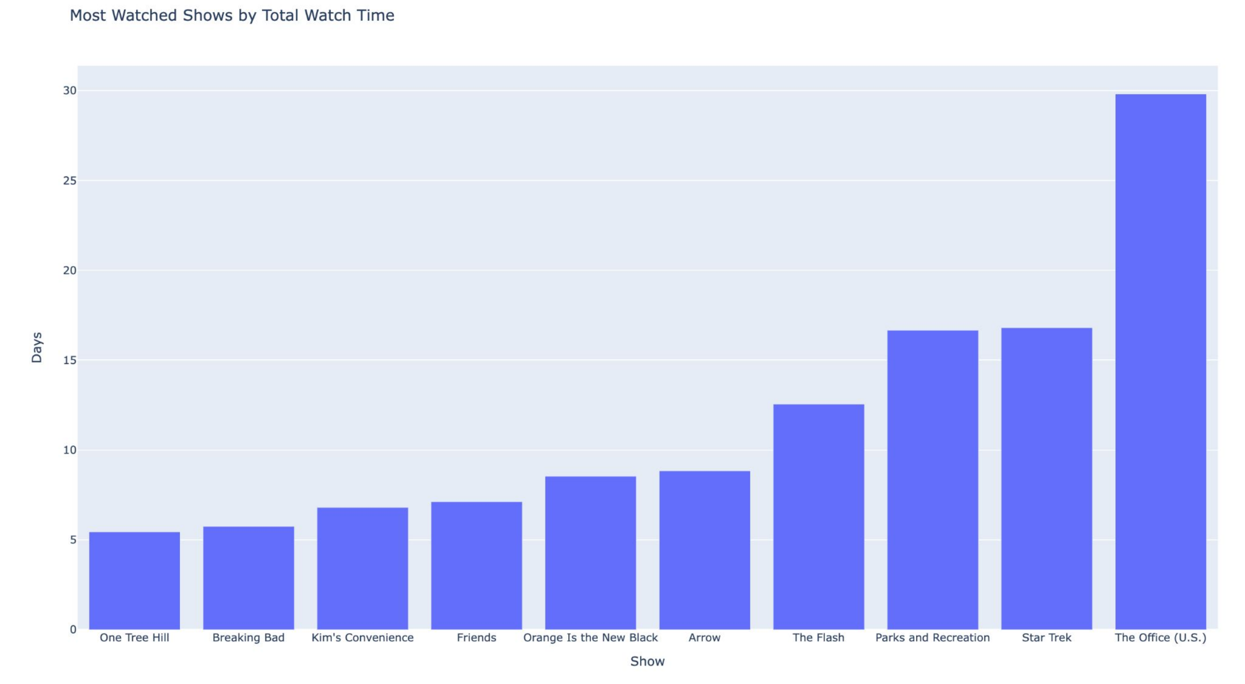
Top 10 Shows Function

- Filter out hooks, trailers, and recaps
- Dictionary for total watchtimes
  - Add a show if it’s not in the dictionary
  - Add watchtime to previous entry if it is
- Make dictionary into a list of dictionaries
  - “Title”, “Total Watchtime” keys
- Sort using custom merge sort algorithm
- New list containing only the top 10 shows
- Create graph



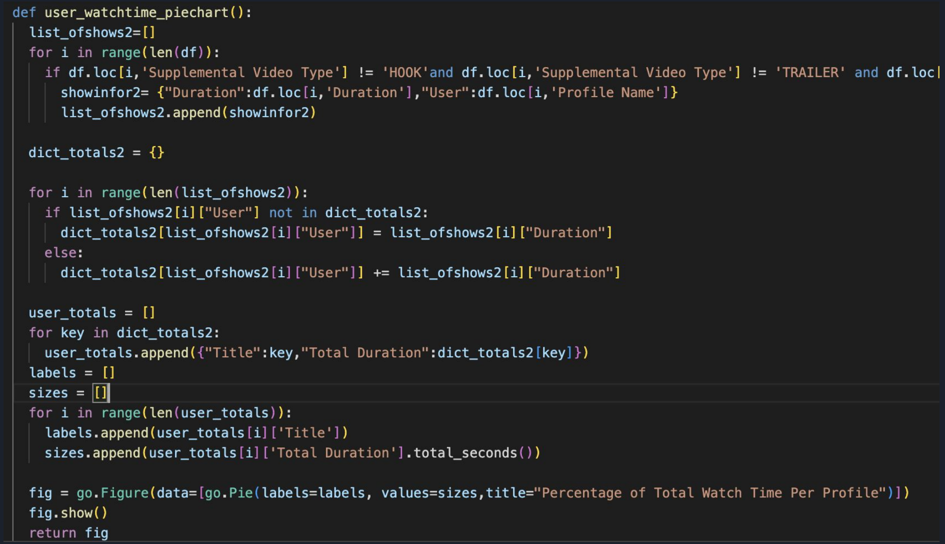
```python
def total_show_watchtime_all():
  list_ofshows=[]
  for i in range(len(df)):
    if df.loc[i,'Supplemental Video Type'] != 'HOOK'and df.loc[i,'Supplemental Video Type'] !
      showinfor= {"Duration":df.loc[i,'Duration'],"Title":df.loc[i,'Title']}
      list_ofshows.append(showinfor)

  dict_totals = {}

  for i in range(len(list_ofshows)):
    if list_ofshows[i]["Title"] not in dict_totals:
      dict_totals[list_ofshows[i]["Title"]] = list_ofshows[i]["Duration"]
    else:
      dict_totals[list_ofshows[i]["Title"]] += list_ofshows[i]["Duration"]

  totals = []
  for key in dict_totals:
    totals.append({"Title":key,"Total Duration":dict_totals[key]})


  mergeSortData(totals,"Total Duration")
  shows = []
  total_durations = []
  for i in range(10):
    shows.insert(0,totals[i]["Title"])
    total_durations.insert(0,totals[i]["Total Duration"].total_seconds()/(3600*24))
  fig = px.bar(x=shows,y=total_durations,title ='Most Watched Shows by Total Watch Time')
  fig.update_layout(
    xaxis_title="show",
    yaxis_title="Days",
    )
  #fig.show()
  return fig
```

Top 10 Shows Graph



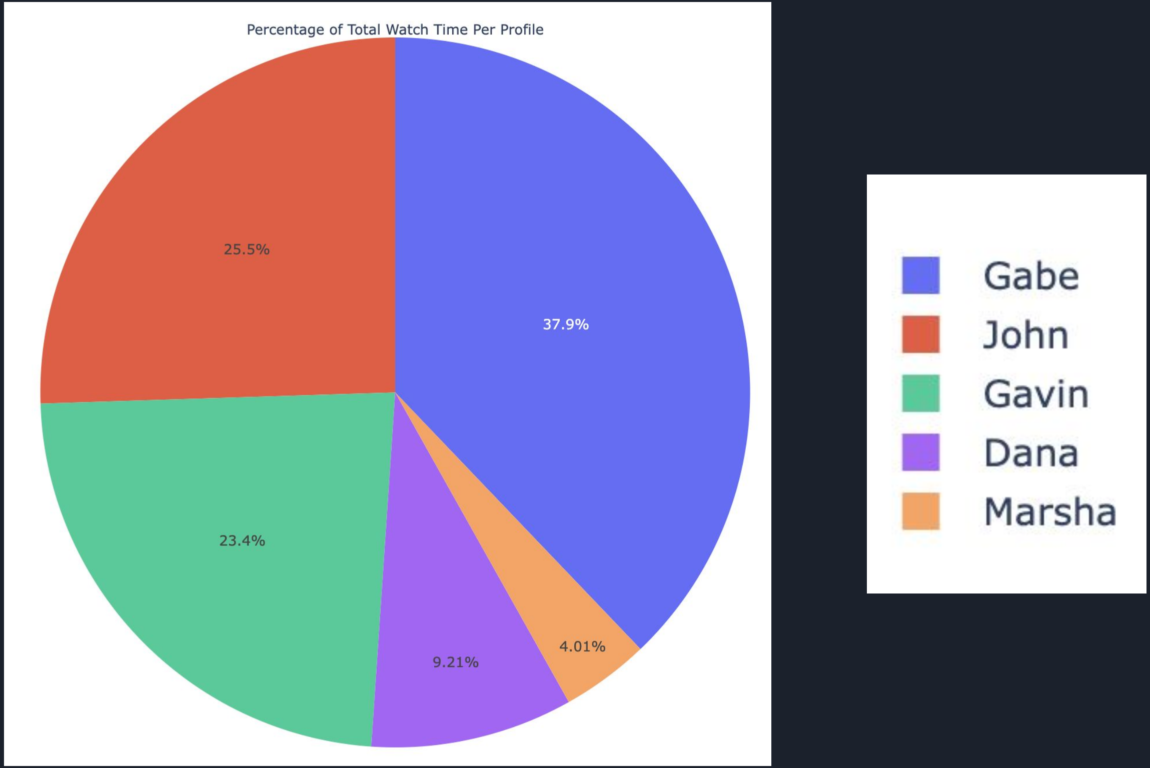
Profile Pie Chart Function

- Similar process to last time
- Instead of Dictionary keys being shows, they are now profiles
- Make two lists: Profile names and total duration
- Create graph



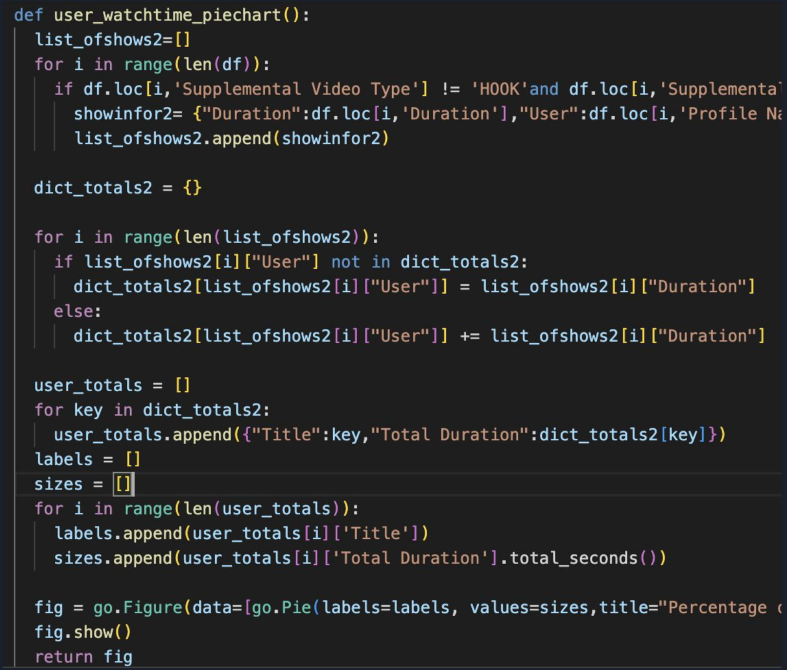
```python
def user_watchtime_piechart():
  list_ofshows2=[]
  for i in range(len(df)):
    if df.loc[i,'Supplemental Video Type'] != 'HOOK'and df.loc[i,'Supplemental Video Type'] != 'TRAILER' and df.loc
      showinfor2= {"Duration":df.loc[i,'Duration'],"User":df.loc[i,'Profile Name']}
      list_ofshows2.append(showinfor2)

  dict_totals2 = {}

  for i in range(len(list_ofshows2)):
    if list_ofshows2[i]["User"] not in dict_totals2:
      dict_totals2[list_ofshows2[i]["User"]] = list_ofshows2[i]["Duration"]
    else:
      dict_totals2[list_ofshows2[i]["User"]] += list_ofshows2[i]["Duration"]

  user_totals = []
  for key in dict_totals2:
    user_totals.append({"Title":key,"Total Duration":dict_totals2[key]})
  labels = []
  sizes = []
  for i in range(len(user_totals)):
    labels.append(user_totals[i]['Title'])
    sizes.append(user_totals[i]['Total Duration'].total_seconds())

  fig = go.Figure(data=[go.Pie(labels=labels, values=sizes,title="Percentage of Total Watch Time Per Profile")])
  fig.show()
  return fig
```

Profile Pie Chart



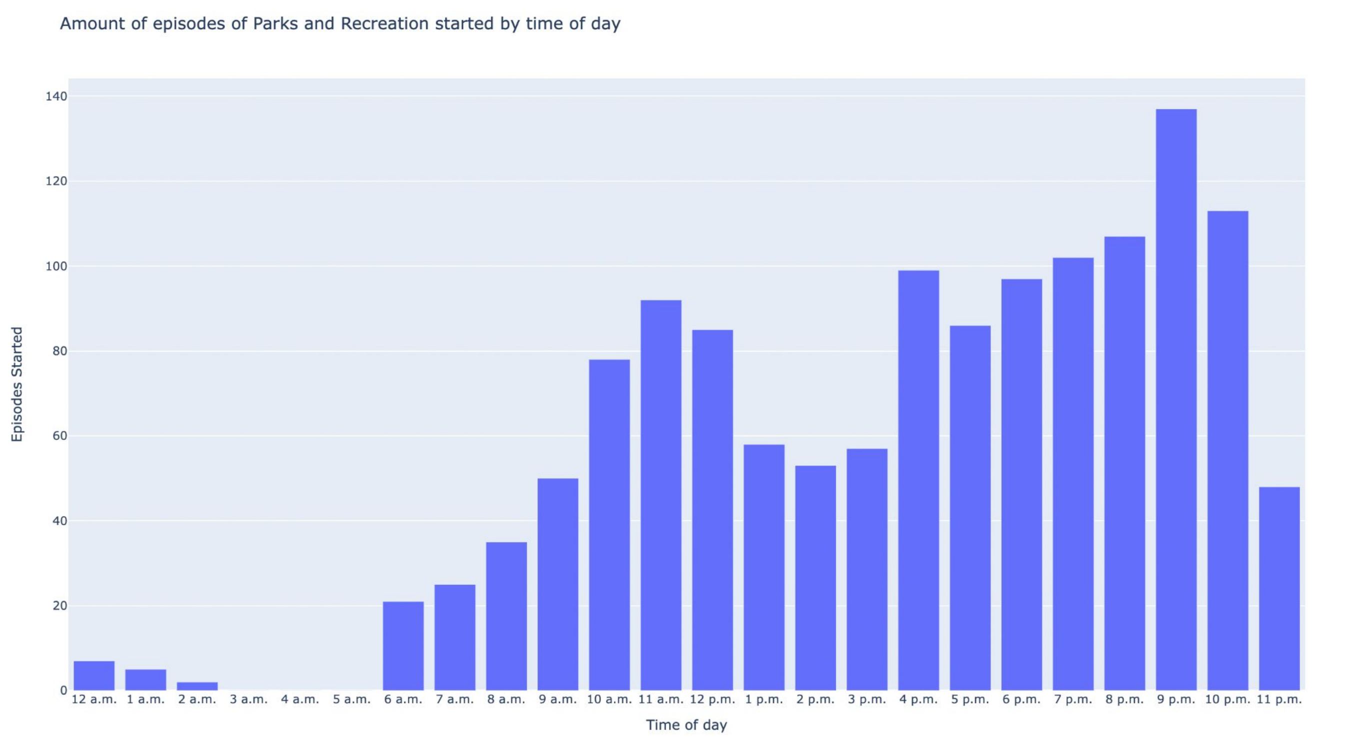
Time of Day Function

- Graph of the time of day episodes are started
- Option to make graph for one or all shows and for one or all profiles
- Filter out trailers etc
- Filter to get specified show/profile
- Append the hour parameter of the time episode was started to a list
- Iterate through the list and count how many times each hour comes up
- Make a list with all times for the x axis
- Create the graph



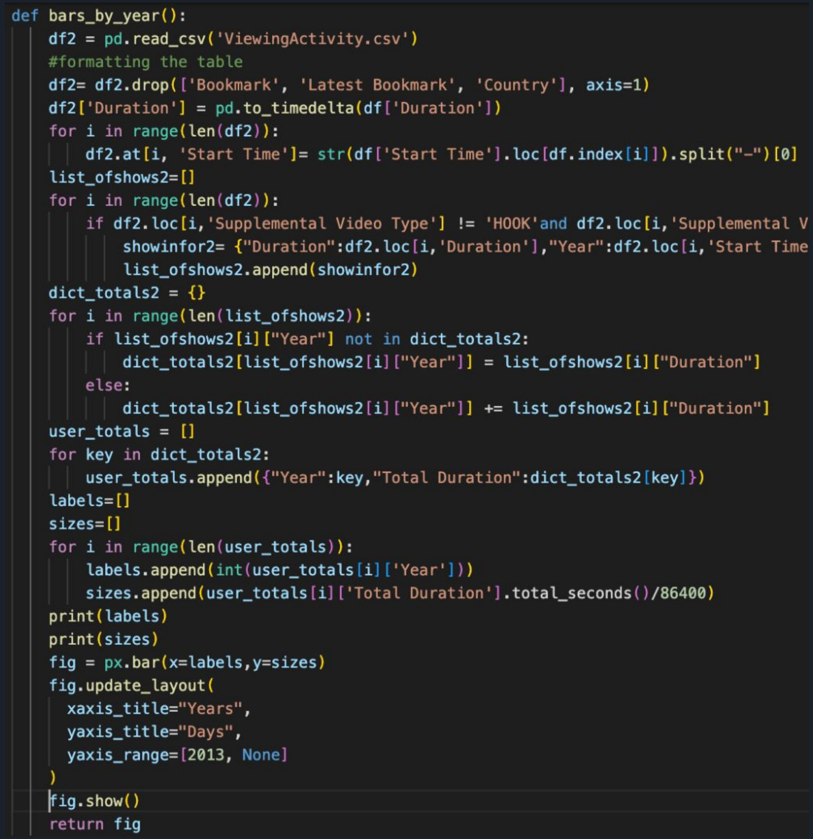
```python
def user_watchtime_piechart():
  list_ofshows2=[]
  for i in range(len(df)):
    if df.loc[i,'Supplemental Video Type'] != 'HOOK'and df.loc[i,'Supplementa
      showinfor2= {"Duration":df.loc[i,'Duration'],"User":df.loc[i,'Profile Na
      list_ofshows2.append(showinfor2)

  dict_totals2 = {}

  for i in range(len(list_ofshows2)):
    if list_ofshows2[i]["User"] not in dict_totals2:
      dict_totals2[list_ofshows2[i]["User"]] = list_ofshows2[i]["Duration"]
    else:
      dict_totals2[list_ofshows2[i]["User"]] += list_ofshows2[i]["Duration"]

  user_totals = []
  for key in dict_totals2:
    user_totals.append({"Title":key,"Total Duration":dict_totals2[key]})
  labels = []
  sizes = []
  for i in range(len(user_totals)):
    labels.append(user_totals[i]['Title'])
    sizes.append(user_totals[i]['Total Duration'].total_seconds())

  fig = go.Figure(data=[go.Pie(labels=labels, values=sizes,title="Percentage
  fig.show()
  return fig
```

Time of Day Graph



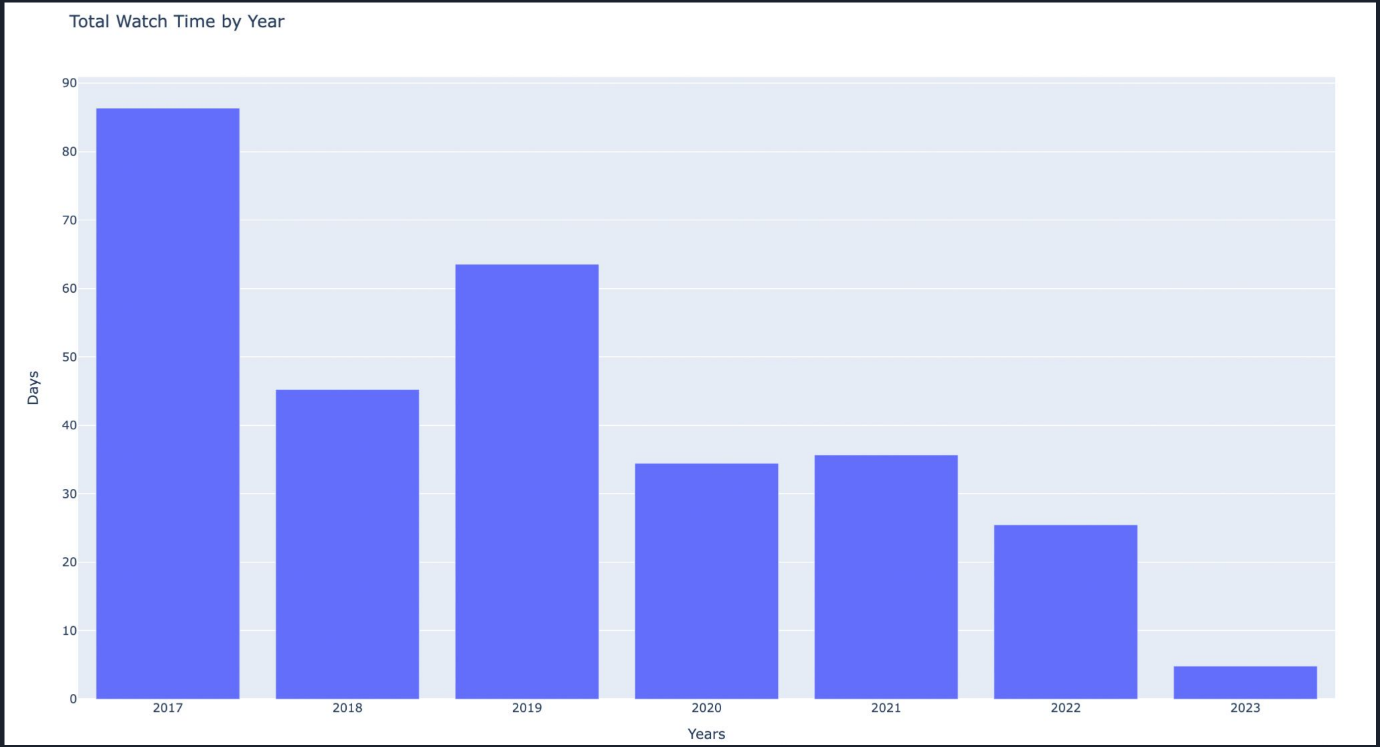
Watch Time by Year Function

- Filter out trailers etc
- Dictionary for total watchtimes
  - Add a year if it’s not in the dictionary
  - Add watchtime to year if it is
- Make list of dictionaries
  - “Year”, “Total Duration” keys
- Make units of Total Duration days
- Create Graph



```python
def bars_by_year():
    df2 = pd.read_csv('ViewingActivity.csv')
    #formatting the table
    df2= df2.drop(['Bookmark', 'Latest Bookmark', 'Country'], axis=1)
    df2['Duration'] = pd.to_timedelta(df['Duration'])
    for i in range(len(df2)):
        df2.at[i, 'Start Time']= str(df['Start Time'].loc[df.index[i]]).split("-")[0]
    list_ofshows2=[]
    for i in range(len(df2)):
        if df2.loc[i,'Supplemental Video Type'] != 'HOOK'and df2.loc[i,'Supplemental V
            showinfor2= {"Duration":df2.loc[i,'Duration'],"Year":df2.loc[i,'Start Time
            list_ofshows2.append(showinfor2)
    dict_totals2 = {}
    for i in range(len(list_ofshows2)):
        if list_ofshows2[i]["Year"] not in dict_totals2:
            dict_totals2[list_ofshows2[i]["Year"]] = list_ofshows2[i]["Duration"]
        else:
            dict_totals2[list_ofshows2[i]["Year"]] += list_ofshows2[i]["Duration"]
    user_totals = []
    for key in dict_totals2:
        user_totals.append({"Year":key,"Total Duration":dict_totals2[key]})
    labels=[]
    sizes=[]
    for i in range(len(user_totals)):
        labels.append(int(user_totals[i]['Year']))
        sizes.append(user_totals[i]['Total Duration'].total_seconds()/86400)
    print(labels)
    print(sizes)
    fig = px.bar(x=labels,y=sizes)
    fig.update_layout(
      xaxis_title="Years",
      yaxis_title="Days",
      yaxis_range=[2013, None]
    )
    fig.show()
    return fig
```

Watch Time by Year Graph



Challenges

- Initial formatting and filtering of the data
  - Removing episode names from title
- Understanding how to work with new packages
- Editing merge sort algorithm
- Allowing functions to work with a specified user and for all users
- We attempted to make it all into one site with a drop down menu but failed
  - Instead we took input from the user about which graph they want to see

Code Demonstration