Example of gathering voice data using microphone

Note: Run the snippet of codes using local jupyter notebook

In [17]:

```
!pip3 install sounddevice
```

```
Requirement already satisfied: sounddevice in c:\users\personal computer\anaconda3\envs\temporary\lib\site-packages
(0.5.1)
Requirement already satisfied: CFFI>=1.0 in c:\users\personal computer\anaconda3\envs\temporary\lib\site-packages (fr
om sounddevice) (1.17.1)
Requirement already satisfied: pycparser in c:\users\personal computer\anaconda3\envs\temporary\lib\site-packages (fr
om CFFI>=1.0->sounddevice) (2.21)
```

In [18]:

```
!pip3 install wavio
```

```
Requirement already satisfied: wavio in c:\users\personal computer\anaconda3\envs\temporary\lib\site-packages (0.0.9)
Requirement already satisfied: numpy>=1.19.0 in c:\users\personal computer\anaconda3\envs\temporary\lib\site-packages
(from wavio) (2.0.1)
```

In [19]:

```
!pip3 install scipy
```

```
Requirement already satisfied: scipy in c:\users\personal computer\anaconda3\envs\temporary\lib\site-packages (1.15.
2)
Requirement already satisfied: numpy<2.5,>=1.23.5 in c:\users\personal computer\anaconda3\envs\temporary\lib\site-pac
kages (from scipy) (2.0.1)
```

In [20]:

```
!apt-get install libportaudio2
```

```
'apt-get' is not recognized as an internal or external command,
operable program or batch file.
```

In [21]:

```
# import required libraries
import sounddevice as sd
from scipy.io.wavfile import write
import wavio as wv

# Sampling frequency
freq = 44100

# Recording duration
```

```
duration = 5

# Start recorder with the given values
# of duration and sample frequency
recording = sd.rec(int(duration * freq),samplerate=freq, channels=2)

# Record audio for the given number of seconds
sd.wait()

# This will convert the NumPy array to an audio
# file with the given sampling frequency
write("recording0.wav", freq, recording)

# Convert the NumPy array to audio file
wv.write("recording1.wav", recording, freq, sampwidth=2)
```

Web Scraping

Image Scraping using BeautifulSoup and Request

In [22]:

```
!pip install bs4
```

```
Requirement already satisfied: bs4 in c:\users\personal computer\anaconda3\envs\temporary\lib\site-packages (0.0.2)
Requirement already satisfied: beautifulsoup4 in c:\users\personal computer\anaconda3\envs\temporary\lib\site-package
s (from bs4) (4.12.3)
Requirement already satisfied: soupsieve>1.2 in c:\users\personal computer\anaconda3\envs\temporary\lib\site-packages
(from beautifulsoup4->bs4) (2.5)
```

In [23]:

```
pip install requests
```

```
Requirement already satisfied: requests in c:\users\personal computer\anaconda3\envs\temporary\lib\site-packages (2.3
2.3)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\personal computer\anaconda3\envs\temporary\lib\si
te-packages (from requests) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in c:\users\personal computer\anaconda3\envs\temporary\lib\site-packages
(from requests) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\personal computer\anaconda3\envs\temporary\lib\site-pac
kages (from requests) (2.3.0)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\personal computer\anaconda3\envs\temporary\lib\site-pac
kages (from requests) (2025.1.31)
Note: you may need to restart the kernel to use updated packages.
```

In [24]:

```python
import requests
from bs4 import BeautifulSoup

def getdata(url):
    r = requests.get(url)
    return r.text

htmldata = getdata("https://www.google.com/")
soup = BeautifulSoup(htmldata, 'html.parser')
for item in soup.find_all('img'):
    print(item['src'])
```

```
/images/branding/googlelogo/1x/googlelogo_white_background_color_272x92dp.png
```

Image Scraping using Selenium

In [25]:

```
pip install selenium
```

```
Requirement already satisfied: selenium in c:\users\personal computer\anaconda3\envs\temporary\lib\site-packages (4.31.0)
Requirement already satisfied: urllib3<3,>=1.26 in c:\users\personal computer\anaconda3\envs\temporary\lib\site-packages (from urllib3[socks]<3,>=1.26->selenium) (2.3.0)
Requirement already satisfied: trio~=0.17 in c:\users\personal computer\anaconda3\envs\temporary\lib\site-packages (from selenium) (0.30.0)
Requirement already satisfied: trio-websocket~=0.9 in c:\users\personal computer\anaconda3\envs\temporary\lib\site-packages (from selenium) (0.12.2)
Requirement already satisfied: certifi>=2021.10.8 in c:\users\personal computer\anaconda3\envs\temporary\lib\site-packages (from selenium) (2025.1.31)
Requirement already satisfied: typing_extensions~=4.9 in c:\users\personal computer\anaconda3\envs\temporary\lib\site-packages (from selenium) (4.12.2)
Requirement already satisfied: websocket-client~=1.8 in c:\users\personal computer\anaconda3\envs\temporary\lib\site-packages (from selenium) (1.8.0)
Requirement already satisfied: attrs>=23.2.0 in c:\users\personal computer\anaconda3\envs\temporary\lib\site-packages (from trio~=0.17->selenium) (24.3.0)
Requirement already satisfied: sortedcontainers in c:\users\personal computer\anaconda3\envs\temporary\lib\site-packages (from trio~=0.17->selenium) (2.4.0)
Requirement already satisfied: idna in c:\users\personal computer\anaconda3\envs\temporary\lib\site-packages (from trio~=0.17->selenium) (3.7)
Requirement already satisfied: outcome in c:\users\personal computer\anaconda3\envs\temporary\lib\site-packages (from trio~=0.17->selenium) (1.3.0.post0)
Requirement already satisfied: sniffio>=1.3.0 in c:\users\personal computer\anaconda3\envs\temporary\lib\site-packages (from trio~=0.17->selenium) (1.3.0)
Requirement already satisfied: cffi>=1.14 in c:\users\personal computer\anaconda3\envs\temporary\lib\site-packages (from trio~=0.17->selenium) (1.17.1)
Requirement already satisfied: wsproto>=0.14 in c:\users\personal computer\anaconda3\envs\temporary\lib\site-packages (from trio-websocket~=0.9->selenium) (1.2.0)
Requirement already satisfied: pysocks!=1.5.7,<2.0,>=1.5.6 in c:\users\personal computer\anaconda3\envs\temporary\lib\site-packages (from urllib3[socks]<3,>=1.26->selenium) (1.7.1)
Requirement already satisfied: pycparser in c:\users\personal computer\anaconda3\envs\temporary\lib\site-packages (from cffi>=1.14->trio~=0.17->selenium) (2.21)
Requirement already satisfied: h11<1,>=0.9.0 in c:\users\personal computer\anaconda3\envs\temporary\lib\site-packages (from wsproto>=0.14->trio-websocket~=0.9->selenium) (0.14.0)
Note: you may need to restart the kernel to use updated packages.
```

In [26]:

```
!pip install selenium
!apt-get update # to update ubuntu to correctly run apt install
!apt install chromium-chromedriver
!cp /usr/lib/chromium-browser/chromedriver /usr/bin
import sys
sys.path.insert(0,'/usr/lib/chromium-browser/chromedriver')
```

```
from selenium import webdriver
import time
import requests
import shutil
import os
import getpass
import urllib.request
import io
import time
from PIL import Image

user = getpass.getuser()
chrome_options = webdriver.ChromeOptions()
chrome_options.add_argument('--headless')
chrome_options.add_argument('--no-sandbox')
chrome_options.add_argument('--disable-dev-shm-usage')
driver = webdriver.Chrome('chromedriver',chrome_options=chrome_options)

search_url = "https://www.google.com/search?q={q}&tbm=isch&tbs=sur%3Afc&hl=en&ved=0CAIQpwVqFwoTCKCa1c6s4-oCFQAAAAAdAA
driver.get(search_url.format(q='Car'))

def scroll_to_end(driver):
    driver.execute_script("window.scrollTo(0, document.body.scrollHeight);")
    time.sleep(5)#sleep_between_interactions

def getImageUrls(name,totalImgs,driver):
    search_url = "https://www.google.com/search?q={q}&tbm=isch&tbs=sur%3Afc&hl=en&ved=0CAIQpwVqFwoTCKCa1c6s4-oCFQAAAA
    driver.get(search_url.format(q=name))
    img_urls = set()
    img_count = 0
    results_start = 0

    while(img_count<totalImgs): #Extract actual images now

        scroll_to_end(driver)

        thumbnail_results = driver.find_elements_by_xpath("//img[contains(@class,'Q4LuWd')]")
        totalResults=len(thumbnail_results)
        print(f"Found: {totalResults} search results. Extracting links from{results_start}:{totalResults}")

        for img in thumbnail_results[results_start:totalResults]:
            img.click()
            time.sleep(2)
            actual_images = driver.find_elements_by_css_selector('img.n3VNCb')
            for actual_image in actual_images:
                if actual_image.get_attribute('src') and 'https' in actual_image.get_attribute('src'):
                    img_urls.add(actual_image.get_attribute('src'))

            img_count=len(img_urls)

            if img_count >= totalImgs:
                print(f"Found: {img_count} image links")
                break
            else:
                print("Found:", img_count, "looking for more image links ...")
                load_more_button = driver.find_element_by_css_selector(".mye4qd")
                driver.execute_script("document.querySelector('.mye4qd').click();")
                results_start = len(thumbnail_results)
    return img_urls
def downloadImages(folder_path,file_name,url):
    try:

        image_content = requests.get(url).content
    except Exception as e:
        print(f"ERROR - COULD NOT DOWNLOAD {url} - {e}")
    try:
        image_file = io.BytesIO(image_content)
        image = Image.open(image_file).convert('RGB')

        file_path = os.path.join(folder_path, file_name)

        with open(file_path, 'wb') as f:
            image.save(f, "JPEG", quality=85)
        print(f"SAVED - {url} - AT: {file_path}")
    except Exception as e:
        print(f"ERROR - COULD NOT SAVE {url} - {e}")

def saveInDestFolder(searchNames,destDir,totalImgs,driver):
    for name in list(searchNames):
        path=os.path.join(destDir,name)
        if not os.path.isdir(path):
            os.mkdir(path)
        print('Current Path',path)
        totalLinks=getImageUrls(name,totalImgs,driver)
        print('totalLinks',totalLinks)

    if totalLinks is None:
            print('images not found for :',name)

    else:
        for i, link in enumerate(totalLinks):
            file_name = f"{i:150}.jpg"
            downloadImages(path,file_name,link)

searchNames=['cat']
destDir=f'/content/drive/My Drive/Colab Notebooks/Dataset/'
totalImgs=5

saveInDestFolder(searchNames,destDir,totalImgs,driver)
```

```
Requirement already satisfied: selenium in c:\users\personal computer\anaconda3\envs\temporary\lib\site-packages (4.3
1.0)
Requirement already satisfied: urllib3<3,>=1.26 in c:\users\personal computer\anaconda3\envs\temporary\lib\site-packa
ges (from urllib3[socks]<3,>=1.26->selenium) (2.3.0)
Requirement already satisfied: trio~=0.17 in c:\users\personal computer\anaconda3\envs\temporary\lib\site-packages (f
rom selenium) (0.30.0)
Requirement already satisfied: trio-websocket~=0.9 in c:\users\personal computer\anaconda3\envs\temporary\lib\site-pa
ckages (from selenium) (0.12.2)
Requirement already satisfied: certifi>=2021.10.8 in c:\users\personal computer\anaconda3\envs\temporary\lib\site-pac
kages (from selenium) (2025.1.31)
Requirement already satisfied: typing_extensions~=4.9 in c:\users\personal computer\anaconda3\envs\temporary\lib\site
-packages (from selenium) (4.12.2)
Requirement already satisfied: websocket-client~=1.8 in c:\users\personal computer\anaconda3\envs\temporary\lib\site-
packages (from selenium) (1.8.0)
Requirement already satisfied: attrs>=23.2.0 in c:\users\personal computer\anaconda3\envs\temporary\lib\site-packages
(from trio~=0.17->selenium) (24.3.0)
Requirement already satisfied: sortedcontainers in c:\users\personal computer\anaconda3\envs\temporary\lib\site-packa
ges (from trio~=0.17->selenium) (2.4.0)
Requirement already satisfied: idna in c:\users\personal computer\anaconda3\envs\temporary\lib\site-packages (from tr
io~=0.17->selenium) (3.7)
Requirement already satisfied: outcome in c:\users\personal computer\anaconda3\envs\temporary\lib\site-packages (from
trio~=0.17->selenium) (1.3.0.post0)
Requirement already satisfied: sniffio>=1.3.0 in c:\users\personal computer\anaconda3\envs\temporary\lib\site-package
s (from trio~=0.17->selenium) (1.3.0)
Requirement already satisfied: cffi>=1.14 in c:\users\personal computer\anaconda3\envs\temporary\lib\site-packages (f
rom trio~=0.17->selenium) (1.17.1)
Requirement already satisfied: wsproto>=0.14 in c:\users\personal computer\anaconda3\envs\temporary\lib\site-packages
(from trio-websocket~=0.9->selenium) (1.2.0)
Requirement already satisfied: pysocks!=1.5.7,<2.0,>=1.5.6 in c:\users\personal computer\anaconda3\envs\temporary\lib
\site-packages (from urllib3[socks]<3,>=1.26->selenium) (1.7.1)
Requirement already satisfied: pycparser in c:\users\personal computer\anaconda3\envs\temporary\lib\site-packages (fr
om cffi>=1.14->trio~=0.17->selenium) (2.21)
Requirement already satisfied: h11<1,>=0.9.0 in c:\users\personal computer\anaconda3\envs\temporary\lib\site-packages
(from wsproto>=0.14->trio-websocket~=0.9->selenium) (0.14.0)
```

```
'apt-get' is not recognized as an internal or external command,
operable program or batch file.
'apt' is not recognized as an internal or external command,
operable program or batch file.
'cp' is not recognized as an internal or external command,
operable program or batch file.
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[26], line 24
     22 chrome_options.add_argument('--no-sandbox')
     23 chrome_options.add_argument('--disable-dev-shm-usage')
---> 24 driver = webdriver.Chrome('chromedriver',chrome_options=chrome_options)
     26 search_url = "https://www.google.com/search?q={q}&tbm=isch&tbs=sur%3Afc&hl=en&ved=0CAIQpwVqFwoTCKCa1c6s4-oCFQAAAAAdAAAAABAC&biw=1251&bih=568"
     27 driver.get(search_url.format(q='Car'))

TypeError: WebDriver.__init__() got an unexpected keyword argument 'chrome_options'
```

Web Scraping of Movies Information using BeautifulSoup

In [43]:

```python
from requests import get
url = 'https://www.imdb.com/search/title?release_date=2017&sort=num_votes,desc&page=1'
response = get(url)
print(response.text[:500])

# since the output is forbidden. I need to find another web
```

```
<html>
<head><title>403 Forbidden</title></head>
<body>
<center><h1>403 Forbidden</h1></center>
</body>
</html>
```

In [29]:

```python
# the web ive search that is similar to imdb for list of movies. Rotten Tomatoes

from requests import get
url = 'https://editorial.rottentomatoes.com/guide/popular-movies/'
response = get(url)
print(response.text[:500])
```

```
<!DOCTYPE html>
<html lang="en-US" class="hitim">
<head prefix="og: http://ogp.me/ns# flixstertomatoes: http://ogp.me/ns/apps/flixstertomatoes#">
        <meta http-equiv="content-type" content="text/html; charset=UTF-8" />

        <!-- OneTrust Cookies Consent Notice start for rottentomatoes.com -->
        <script src="https://cdn.cookielaw.org/consent/7e979733-6841-4fce-9182-515fac69187f/otSDKStub.js"
                type="text/javascript"
                charset="UTF-8"
                data-domain-script="7e979733-6841-4fce-9182-515fac69187f"
                integr
```

In [30]:

```
from bs4 import BeautifulSoup
html_soup = BeautifulSoup(response.text, 'html.parser')
headers = {'Accept-Language': 'en-US,en;q=0.8'}
type(html_soup)
```

Out[30]:

```
bs4.BeautifulSoup
```

In [31]:

```
movie_containers = html_soup.find_all('div', class_ = 'col-sm-18 col-full-xs countdown-item-content')
print(type(movie_containers))
print(len(movie_containers))
```

```
<class 'bs4.element.ResultSet'>
30
```

In [32]:

```
first_movie = movie_containers[0]
first_movie
```

```
Out[32]: <div class="col-sm-18 col-full-xs countdown-item-content">
<div class="row countdown-item-title-bar">
<div class="col-sm-20 col-full-xs" style="height: 100%;">
<div class="article_movie_title" style="float: left;">
<h2>
<a href="https://www.rottentomatoes.com/m/sinners_2025">Sinners</a>
<span class="subtle start-year">(2025)</span>
<br/>
<img alt="Tomatometer icon" class="icon tiny" decoding="async" src="https://images.fandango.com/cms/assets/c6672520-d359-11ea-a15f-bdf29fa24277--certified-fresh.png"> <span class="tMeterScore" style="margin-right: 10px;">98%</span>
</img></h2>
</div>
</div>
<div class="col-sm-4 col-full-xs" style="height: 100%;">
<div class="countdown-index">#1</div>
</div>
</div>
<div class="row countdown-item-details">
<div class="col-sm-24">
<div class="info critics-consensus">
<span class="descriptor">Critics Consensus:</span> A rip-roaring fusion of masterful visual storytelling and toe-tapping music, writer-director Ryan Coogler's first original blockbuster reveals the full scope of his singular imagination.                                   </div>
<div class="info synopsis">
<span class="descriptor">Synopsis:</span> Trying to leave their troubled lives behind, twin brothers (Michael B. Jordan) return to their hometown to start again, only                                              <a class="" data-pageheader="" href="https://www.rottentomatoes.com/m/sinners_2025" target="_top"> [More]</a>
</div>
<div class="info cast">
<span class="descriptor">
                Starring:       </span>
<a class="" href="//www.rottentomatoes.com/celebrity/michael_b_jordan">Michael B. Jordan</a>
                ,               <a class="" href="//www.rottentomatoes.com/celebrity/hailee_steinfeld">Hailee Steinfeld</a>
                ,               <a class="" href="//www.rottentomatoes.com/celebrity/miles_caton">Miles Caton</a>
                ,               <a class="" href="//www.rottentomatoes.com/celebrity/jack-oconnell">Jack O'Connell</a>
</div>
<div class="info director">
<span class="descriptor">
                Directed By:    </span>
<a class="" href="//www.rottentomatoes.com/celebrity/ryan_coogler">Ryan Coogler</a>
```

```
</div> </div>
</div>
</div>
```

```
In [33]: first_movie.div
```

```
Out[33]: <div class="row countdown-item-title-bar">
<div class="col-sm-20 col-full-xs" style="height: 100%;">
<div class="article_movie_title" style="float: left;">
<h2>
<a href="https://www.rottentomatoes.com/m/sinners_2025">Sinners</a>
<span class="subtle start-year">(2025)</span>
<br/>
<img alt="Tomatometer icon" class="icon tiny" decoding="async" src="https://images.fandango.com/cms/assets/c6672520-d359-11ea-a15f-bdf29fa24277--certified-fresh.png"> <span class="tMeterScore" style="margin-right: 10px;">98%</span>
</img></h2>
</div>
</div>
<div class="col-sm-4 col-full-xs" style="height: 100%;">
<div class="countdown-index">#1</div>
</div>
</div>
```

```
In [34]: first_movie.a
```

```
Out[34]: <a href="https://www.rottentomatoes.com/m/sinners_2025">Sinners</a>
```

Movie Name

```
In [37]: first_name = first_movie.a.text
         first_name
```

```
Out[37]: 'Sinners'
```

Movie Year

```
In [39]: first_year = first_movie.span.text
         first_year
```

```
Out[39]: '(2025)'
```

Movie Score (Tomato Meter)

In [93]:

```
# i copied the previous code to find a specific output. turned out right
meter_score = first_movie.find('span', class_ = 'tMeterScore')
print(len(meter_score))
meter_score
```

```
1
```

Out[93]:

```
<span class="tMeterScore" style="margin-right: 10px;">98%</span>
```

In [94]:

```
first_mscore = meter_score.text
first_mscore
```

Out[94]:

```
'98%'
```

In [ ]:

```
# Since there is no metascore and number of votes. Ill have to substitute it with the first cast and first director.
# why did I only picked the first cast and director? because its kind of difficult for me to extract all since the ou
```

Cast (First cast only)

In [101…

```
first_cast = first_movie.find('div', class_ = 'info cast').a.text
first_cast
```

Out[101…

```
'Michael B. Jordan'
```

Director (First director only)

In [104…

```
first_director = first_movie.find('div', class_ = 'info director').a.text
first_director
```

Out[104…

```
'Ryan Coogler'
```

In [121…

```
# Lists to store the scraped data in
names = []
years = []
tomatometer = []
cast = []
director = []
# Extract data from individual movie container
```

```
for container in movie_containers:

# If the movie has Metascore, then extract:
    if container.find('span', class_ = 'tMeterScore') is not None:
# The name
        name = container.a.text
        names.append(name)
# The year
        year = container.span.text
        years.append(year)
# The tomatometer
        tmeter = container.find('span', class_ = 'tMeterScore').text
        tomatometer.append(tmeter)
# The cast
        f_cast = container.find('div', class_ = 'info cast').a.text
        cast.append(f_cast)
# The director
        f_director = container.find('div', class_ = 'info director').a.text
        director.append(f_director)
```

In [122…

```
import pandas as pd
movie_ratings = pd.DataFrame({'movie': names,
'year': years,
'tomato meter': tomatometer,
'cast': cast,
'director': director
})
print(test_df.info())
movie_ratings
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 30 entries, 0 to 29
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   movie         30 non-null     object
 1   year          30 non-null     object
 2   tomato meter  30 non-null     object
 3   cast          30 non-null     object
 4   director      30 non-null     object
dtypes: object(5)
memory usage: 1.3+ KB
None
```

Out[122…

	movie	year	tomato meter	cast	director
0	Sinners	(2025)	98%	Michael B. Jordan	Ryan Coogler
1	A Minecraft Movie	(2025)	48%	Jason Momoa	Jared Hess
2	Warfare	(2025)	93%	D'Pharaoh Woon-A-Tai	Ray Mendoza
3	The Amateur	(2025)	61%	Rami Malek	James Hawes
4	Black Bag	(2025)	96%	Cate Blanchett	Steven Soderbergh
5	Drop	(2025)	84%	Meghann Fahy	Christopher Landon
6	Companion	(2025)	93%	Sophie Thatcher	Drew Hancock
7	Disney's Snow White	(2025)	40%	Rachel Zegler	Marc Webb
8	Mickey 17	(2025)	77%	Robert Pattinson	Bong Joon Ho
9	Novocaine	(2025)	81%	Jack Quaid	Dan Berk
10	The King of Kings	(2025)	64%	Kenneth Branagh	Jang Seongho
11	Captain America: Brave New World	(2025)	48%	Anthony Mackie	Julius Onah
12	A Working Man	(2025)	49%	Jason Statham	David Ayer
13	G20	(2025)	58%	Clark Gregg	Patricia Riggen
14	The Ballad of Wallis Island	(2025)	97%	Tom Basden	James Griffiths
15	I'm Still Here	(2024)	97%	Fernanda Torres	Walter Salles
16	The Wedding Banquet	(2025)	88%	Bowen Yang	Andrew Ahn
17	The Order	(2024)	92%	Jude Law	Justin Kurzel
18	One of Them Days	(2025)	94%	Keke Palmer	Lawrence Lamont
19	The Accountant 2	(2025)	77%	Ben Affleck	Gavin O'Connor
20	The Ugly Stepsister	(2025)	96%	Lea Myren	Emilie Blichfeldt
21	The Woman in the Yard	(2025)	44%	Danielle Deadwyler	Jaume Collet-Serra

	movie	year	tomato meter	cast	director
22	Death of a Unicorn	(2025)	54%	Paul Rudd	Alex Scharfman
23	iHostage	(2025)	- -	Soufiane Moussouli	Bobby Boermans
24	2073	(2024)	48%	Naomi Ackie	Asif Kapadia
25	Thunderbolts	(2025)	- -	Florence Pugh	Jake Schreier
26	The Electric State	(2025)	15%	Millie Bobby Brown	Anthony Russo
27	The Passion of the Christ	(2004)	49%	Jim Caviezel	Mel Gibson
28	The Chosen: Last Supper - Part 1	(2025)	- -	Jonathan Roumie	Dallas Jenkins
29	The Life List	(2025)	45%	Sofia Carson	Adam Brooks

In [ ]:

```
# Can't do script for multiple page because the web rotten tomato only listed 30 movies which is enough in 1 page. th
```

In [126…

```
movie_ratings['year'].unique()
```

Out[126…

```
array(['(2025)', '(2024)', '(2004)'], dtype=object)
```

In [127…

```
movie_ratings['year'] = (movie_ratings.year.apply(lambda x:x.replace('(','')))
```

In [128…

```
movie_ratings['year'].unique()
```

Out[128…

```
array(['2025)', '2024)', '2004)'], dtype=object)
```

In [129…

```
movie_ratings['year'] = (movie_ratings.year.apply(lambda x:x.replace(')','')))
```

In [130…

```
movie_ratings['year'].unique()
```

Out[130…

```
array(['2025', '2024', '2004'], dtype=object)
```

In [131…

```
movie_ratings['year'] = movie_ratings['year'].astype(int)
```

In [132…

```
movie_ratings['year'].unique()
```

Out[132…

```
array([2025, 2024, 2004])
```

In [133…

```
movie_ratings.dtypes
```

Out[133…

```
movie           object
year             int64
tomato meter    object
cast            object
director        object
dtype: object
```

In [134…

```
movie_ratings.head(10)
```

Out[134…

	movie	year	tomato meter	cast	director
0	Sinners	2025	98%	Michael B. Jordan	Ryan Coogler
1	A Minecraft Movie	2025	48%	Jason Momoa	Jared Hess
2	Warfare	2025	93%	D'Pharaoh Woon-A-Tai	Ray Mendoza
3	The Amateur	2025	61%	Rami Malek	James Hawes
4	Black Bag	2025	96%	Cate Blanchett	Steven Soderbergh
5	Drop	2025	84%	Meghann Fahy	Christopher Landon
6	Companion	2025	93%	Sophie Thatcher	Drew Hancock
7	Disney's Snow White	2025	40%	Rachel Zegler	Marc Webb
8	Mickey 17	2025	77%	Robert Pattinson	Bong Joon Ho
9	Novocaine	2025	81%	Jack Quaid	Dan Berk

In [135…

```
movie_ratings.tail(10)
```

Out[135…

	movie	year	tomato meter	cast	director
20	The Ugly Stepsister	2025	96%	Lea Myren	Emilie Blichfeldt
21	The Woman in the Yard	2025	44%	Danielle Deadwyler	Jaume Collet-Serra
22	Death of a Unicorn	2025	54%	Paul Rudd	Alex Scharfman
23	iHostage	2025	- -	Soufiane Moussouli	Bobby Boermans
24	2073	2024	48%	Naomi Ackie	Asif Kapadia
25	Thunderbolts	2025	- -	Florence Pugh	Jake Schreier
26	The Electric State	2025	15%	Millie Bobby Brown	Anthony Russo
27	The Passion of the Christ	2004	49%	Jim Caviezel	Mel Gibson
28	The Chosen: Last Supper - Part 1	2025	- -	Jonathan Roumie	Dallas Jenkins
29	The Life List	2025	45%	Sofia Carson	Adam Brooks

In [136…

```
# I'll try to clean the tomator meter also
```

In [137…

```
movie_ratings['tomato meter'].unique()
```

Out[137…

```
array(['98%', '48%', '93%', '61%', '96%', '84%', '40%', '77%', '81%',
       '64%', '49%', '58%', '97%', '88%', '92%', '94%', '44%', '54%',
       '- -', '15%', '45%'], dtype=object)
```

In [139…

```
movie_ratings['tomato meter'] = (movie_ratings['tomato meter'].apply(lambda x:x.replace('- -','0')))
```

In [140…

```
movie_ratings['tomato meter'].unique()
```

Out[140…

```
array(['98%', '48%', '93%', '61%', '96%', '84%', '40%', '77%', '81%',
       '64%', '49%', '58%', '97%', '88%', '92%', '94%', '44%', '54%', '0',
       '15%', '45%'], dtype=object)
```

In [141…

```
movie_ratings['tomato meter'] = (movie_ratings['tomato meter'].apply(lambda x:x.replace('%','')))
```

In [142…

```
movie_ratings['tomato meter'].unique()
```

Out[142…

```
array(['98', '48', '93', '61', '96', '84', '40', '77', '81', '64', '49',
       '58', '97', '88', '92', '94', '44', '54', '0', '15', '45'],
      dtype=object)
```

In [145…

```
# since that the tomato meter is still an object even if its numerial. Ill have to turn it into integer data type jus
movie_ratings['tomato meter'] = movie_ratings['tomato meter'].astype(int)
```

In [144…

```
movie_ratings.dtypes
```

Out[144…

```
movie            object
year              int64
tomato meter      int64
cast             object
director         object
dtype: object
```

In [146…

```
movie_ratings
```

Out[146…

	movie	year	tomato meter	cast	director
0	Sinners	2025	98	Michael B. Jordan	Ryan Coogler
1	A Minecraft Movie	2025	48	Jason Momoa	Jared Hess
2	Warfare	2025	93	D'Pharaoh Woon-A-Tai	Ray Mendoza
3	The Amateur	2025	61	Rami Malek	James Hawes
4	Black Bag	2025	96	Cate Blanchett	Steven Soderbergh
5	Drop	2025	84	Meghann Fahy	Christopher Landon
6	Companion	2025	93	Sophie Thatcher	Drew Hancock
7	Disney's Snow White	2025	40	Rachel Zegler	Marc Webb
8	Mickey 17	2025	77	Robert Pattinson	Bong Joon Ho
9	Novocaine	2025	81	Jack Quaid	Dan Berk
10	The King of Kings	2025	64	Kenneth Branagh	Jang Seongho
11	Captain America: Brave New World	2025	48	Anthony Mackie	Julius Onah
12	A Working Man	2025	49	Jason Statham	David Ayer
13	G20	2025	58	Clark Gregg	Patricia Riggen
14	The Ballad of Wallis Island	2025	97	Tom Basden	James Griffiths
15	I'm Still Here	2024	97	Fernanda Torres	Walter Salles
16	The Wedding Banquet	2025	88	Bowen Yang	Andrew Ahn
17	The Order	2024	92	Jude Law	Justin Kurzel
18	One of Them Days	2025	94	Keke Palmer	Lawrence Lamont
19	The Accountant 2	2025	77	Ben Affleck	Gavin O'Connor
20	The Ugly Stepsister	2025	96	Lea Myren	Emilie Blichfeldt
21	The Woman in the Yard	2025	44	Danielle Deadwyler	Jaume Collet-Serra

	movie	year	tomato meter	cast	director
22	Death of a Unicorn	2025	54	Paul Rudd	Alex Scharfman
23	iHostage	2025	0	Soufiane Moussouli	Bobby Boermans
24	2073	2024	48	Naomi Ackie	Asif Kapadia
25	Thunderbolts	2025	0	Florence Pugh	Jake Schreier
26	The Electric State	2025	15	Millie Bobby Brown	Anthony Russo
27	The Passion of the Christ	2004	49	Jim Caviezel	Mel Gibson
28	The Chosen: Last Supper - Part 1	2025	0	Jonathan Roumie	Dallas Jenkins
29	The Life List	2025	45	Sofia Carson	Adam Brooks

In [203…

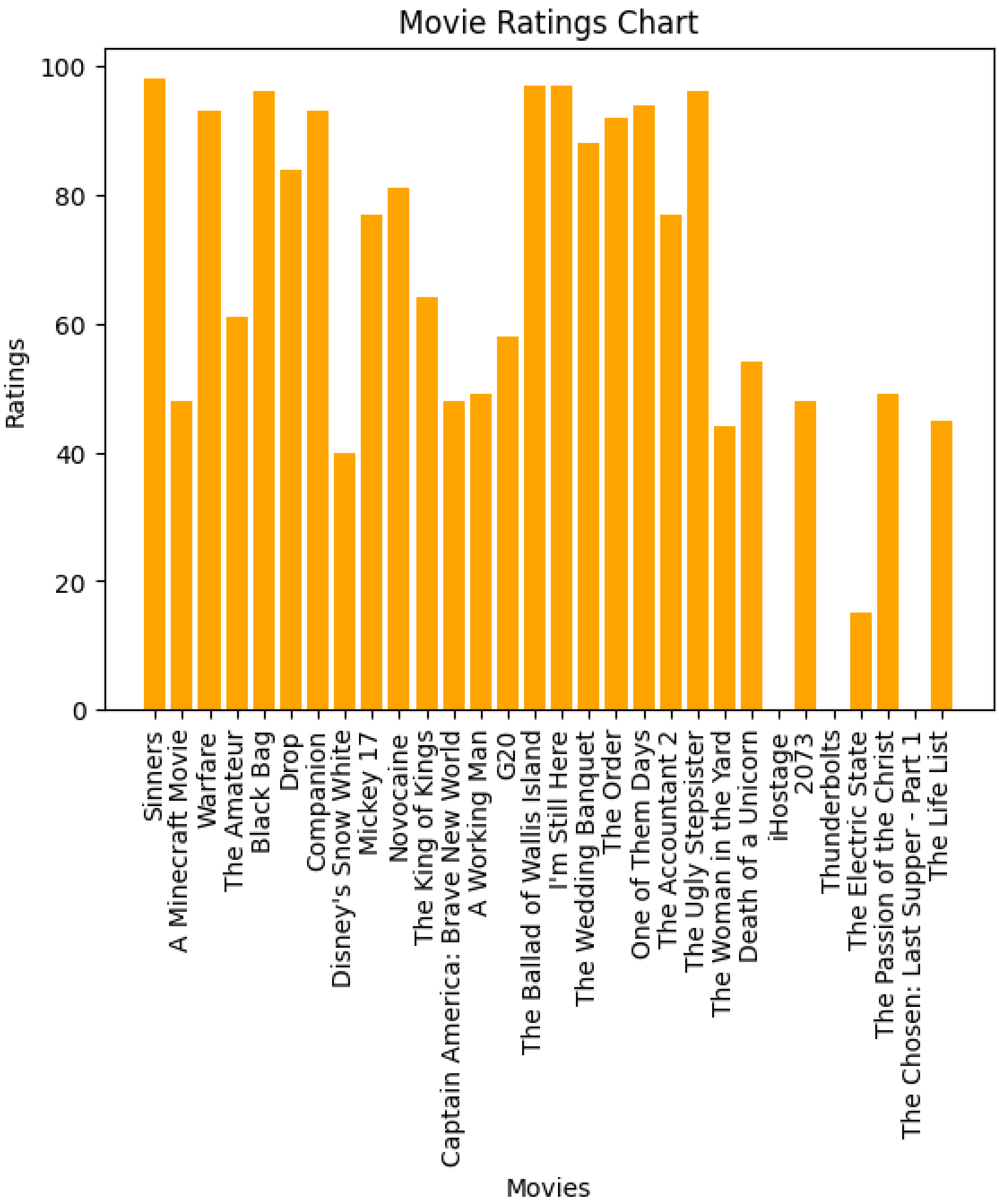
```python
# ill try to visualize
import matplotlib.pyplot as plt

x = movie_ratings['movie']
y = movie_ratings['tomato meter']

plt.bar(x, y, color = 'orange')

plt.xlabel('Movies')
plt.ylabel('Ratings')
plt.title('Movie Ratings Chart')
plt.tick_params(axis = 'x', rotation = 90)

plt.show()
```


Movie Ratings Chart
Ratings
100
80
60
40
20
0
Sinners
A Minecraft Movie
Warfare
The Amateur
Black Bag
Drop
Companion
Disney's Snow White
Mickey 17
Novocaine
The King of Kings
Captain America: Brave New World
A Working Man
G20
The Ballad of Wallis Island
I'm Still Here
The Wedding Banquet
The Order
One of Them Days
The Accountant 2
The Ugly Stepsister
The Woman in the Yard
Death of a Unicorn
iHostage
2073
Thunderbolts
The Electric State
The Passion of the Christ
The Chosen: Last Supper - Part 1
The Life List
Movies

In [208…

```
'''The bar chart above shows that taller bars are more favorable
   among consumers, with the movie "Sinners" ranked as the most favored.'''
```

Out[208…

```
'The bar chart shows that taller bars are more favorable \n   among consumers, with the movie "Sinners" ranked as th
e most favored.'
```

In [204…

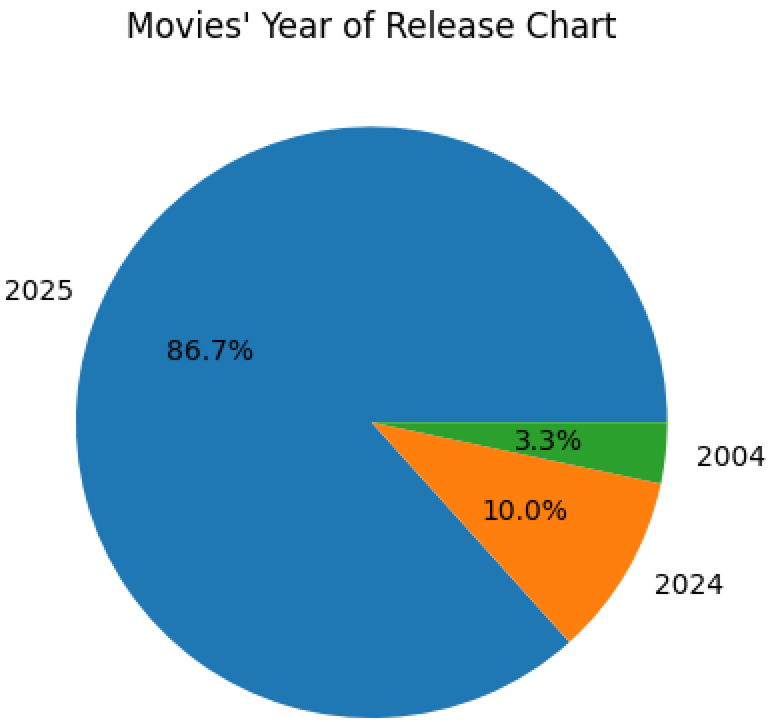
```
year_counts = movie_ratings.year.value_counts()

year_label = movie_ratings.year.unique()

plt.pie(year_counts, labels = year_label, autopct='%1.1f%%')
plt.title("Movies' Year of Release Chart")
```

Out[204…

```
Text(0.5, 1.0, "Movies' Year of Release Chart")
```



In [ ]:

```
# The pie chart above suggests that movies released in 2025 is popular today.
```