Names: Ava Anderson, Kamila Podsiadlo, Emma Marion

1. Project Goals

The original goal of our project was to cross reference Star Wars movie, lego, and in-universe data. While we originally planned on using SWAPI, OMDB API, and Rebrickable LEGO API. We originally planned on gathering in-universe character (height, homeworld) & planet data (climate, size, population) from SWAPI, critic and audience ratings from OMDB, and number of lego pieces in star wars sets from Rebrickable.

We planned on showing the rating comparisons of each movie, data for the characters and planets in the movie (average height per climate), and the average lego set complexity for sets based on each movie.

2. Goals Achieved

While we ended up using OMDB API and Rebrickable, we had to drop SWAPI due to a lack of data to fulfill the project's requirement of "100 rows per table." We replaced it with webscraping Wookiepedia, the Star Wars wikipedia. We decided on webscraping their timeline of canon media page for star wars comic data.

- For OMDB we gathered the box office, IMDB rating, and Rotten Tomatoes rating for each movie.
- For Rebrickable, we gathered the theme, number of pieces, year released, and name for each lego set.
- For Wookiepedia, we gathered the name and year published for each comic.

In the end, we ended up achieving a fun and insightful analysis of the Star Wars franchise across multiple media categories.

3. Problems Encountered

Throughout the process, we ran into a number of issues that required debugging and iteration.

Firstly, the SWAPI didn't have enough people or planets to fill 100 rows for either table. Then, Emma found an updated SWAPI called SWAPI Reborn that seemed to have enough vehicle data to meet the project requirements. The plan was to calculate the average cost of vehicles for each manufacturer. Through this process she even found a typo in the manufacturer names in the API and made her first "issue" request on GitHub! Unfortunately, the documentation for the API was slightly misleading: the vehicle data had large gaps. This meant scrapping everything (again) and finding a new API/webscrape. This led her to using wookiepedia!

Secondly, originally we had a separate main.py file but it became confusing on what parts were missing from each collect_.py. We then decided to make the code more modular, and make the user run each python file individually to run the entire program logic.

Thirdly, lego sets have duplicate names, which means duplicate string data in our database. We fixed this by adding a separate table for lego set names that shares an integer key with the main lego sets table. This did not satisfy the project requirements for having two tables that share a key, as this was effectively splitting one table into two, which leads us to our fourth problem…

Fourthly, collecting and comparing lego themes proved to be a challenge. This meant we had to divide the 25 item insertion limit between multiple lego themes. Since the Rebrickable API uses "pages" to serve each request, we had to implement a way for the program to calculate what page of the API to call per theme, based on the data already collected.

4. Calculation File

calculation_results.txt

```
Star Wars Comics Released Per Year
==============================
Year: 2021 | Comics Released: 40
Year: 2022 | Comics Released: 19
Year: 2023 | Comics Released: 24
Year: 2024 | Comics Released: 8
Year: 2025 | Comics Released: 2
Year: 2026 | Comics Released: 1


======================================================================
STAR WARS vs ALL OTHER MOVIES - RATING ANALYSIS
======================================================================

AVERAGE RATINGS COMPARISON
----------------------------------------------------------------------
Star Wars Movies (11 total):
  Average IMDb Rating:           74.5/100
  Average Rotten Tomatoes Score: 77.5/100

Other Movies (93 total):
  Average IMDb Rating:           82.4/100
  Average Rotten Tomatoes Score: 88.3/100

Comparison:
  Star Wars IMDb is 7.9 points LOWER
  Star Wars RT is 10.8 points LOWER

```

calculation_results.txt

```
STAR WARS MOVIES - CRITIC vs AUDIENCE AGREEMENT
------------------------------------------------------------------------
Movie Title                                IMDb     RT       Diff
------------------------------------------------------------------------
Rogue One: A Star Wars Story                78.0    84.0    -6.0
Solo: A Star Wars Story                     69.0    69.0    +0.0
Star Wars: Episode I - The Phantom Menace   65.0    54.0   +11.0
Star Wars: Episode II - Attack of the Clon...  66.0    62.0    +4.0
Star Wars: Episode III - Revenge of the Si...  76.0    79.0    -3.0
Star Wars: Episode IV - A New Hope          86.0    94.0    -8.0
Star Wars: Episode IX - The Rise of Skywal...  64.0    51.0   +13.0
Star Wars: Episode V - The Empire Strikes ...  87.0    93.0    -6.0
Star Wars: Episode VI - Return of the Jedi  83.0    83.0    +0.0
Star Wars: Episode VII - The Force Awakens  77.0    93.0   -16.0
Star Wars: Episode VIII - The Last Jedi     68.0    91.0   -23.0

```

calculation_results.txt

```
TOP 10 HIGHEST RATED MOVIES (ALL MOVIES)
------------------------------------------------------------------------
By IMDb Rating:
 1. [Other]     The Shawshank Redemption                  93.0
 2. [Other]     The Dark Knight                           91.0
 3. [Other]     Schindler's List                          90.0
 4. [Other]     The Lord of the Rings: The Return of the... 90.0
 5. [Other]     Forrest Gump                              88.0
 6. [Other]     Pulp Fiction                              88.0
 7. [Other]     Fight Club                                88.0
 8. [Other]     Inception                                 88.0
 9. [STAR WARS] Star Wars: Episode V - The Empire Strike... 87.0
10. [Other]     The Matrix                                87.0

By Rotten Tomatoes Score:
 1. [Other]     Toy Story                                 100
 2. [Other]     3 Idiots                                  100
 3. [Other]     Parasite                                  99
 4. [Other]     Finding Nemo                              99
 5. [Other]     Schindler's List                          98
 6. [Other]     Toy Story 3                               98
 7. [Other]     Up                                        98
 8. [Other]     Inside Out                                98
 9. [Other]     Mad Max: Fury Road                        97
10. [Other]     Spotlight                                 97

```

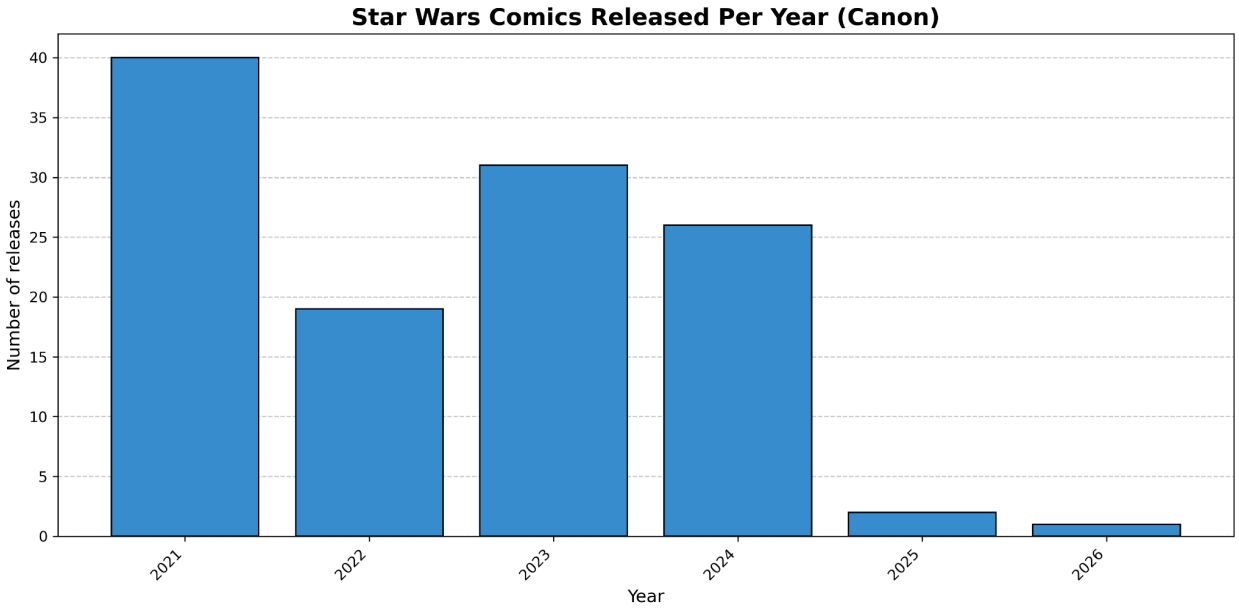
calculation_results.txt

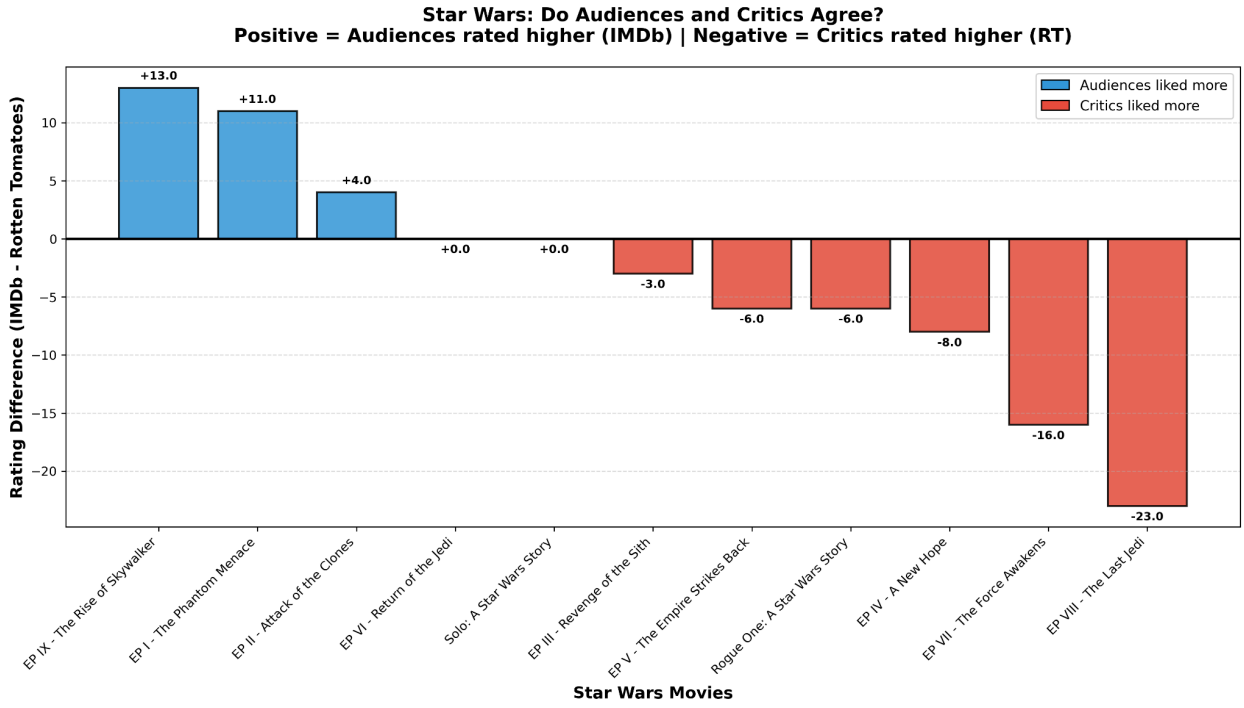
```
76  ======================================================================
77  LEGO SET COMPLEXITY ANALYSIS
78  ======================================================================
79
80  AVERAGE LEGO COMPLEXITY BY YEAR
81  ----------------------------------------------------------------------
82  Year: 1983   | Average Pieces per Set:  103.0
83  Year: 1985   | Average Pieces per Set:  142.7
84  Year: 1986   | Average Pieces per Set:   39.0
85  Year: 1997   | Average Pieces per Set:   64.0
86  Year: 1998   | Average Pieces per Set:   29.0
87  Year: 1999   | Average Pieces per Set:   28.2
88  Year: 2000   | Average Pieces per Set:  106.0
89  Year: 2001   | Average Pieces per Set:  239.0
90  Year: 2004   | Average Pieces per Set:  385.5
91  Year: 2005   | Average Pieces per Set:    0.0
92  Year: 2007   | Average Pieces per Set:    0.0
93  Year: 2008   | Average Pieces per Set:   64.0
94  Year: 2009   | Average Pieces per Set:   52.7
95  Year: 2010   | Average Pieces per Set:   54.6
96  Year: 2011   | Average Pieces per Set:  215.8
97  Year: 2012   | Average Pieces per Set:   37.8
98  Year: 2013   | Average Pieces per Set:   47.8
99  Year: 2014   | Average Pieces per Set:   51.0
100 Year: 2015   | Average Pieces per Set:   21.0
101 Year: 2016   | Average Pieces per Set:   53.0
102 Year: 2017   | Average Pieces per Set:  102.0
103 Year: 2018   | Average Pieces per Set:  115.4
104 Year: 2019   | Average Pieces per Set:   16.8
105 Year: 2020   | Average Pieces per Set:   60.6
106 Year: 2021   | Average Pieces per Set:   36.2
107 Year: 2022   | Average Pieces per Set:   49.9
108 Year: 2023   | Average Pieces per Set:   44.3
109 Year: 2024   | Average Pieces per Set:   63.5
110 Year: 2025   | Average Pieces per Set:   82.2
111 Year: 2026   | Average Pieces per Set:    0.0
```

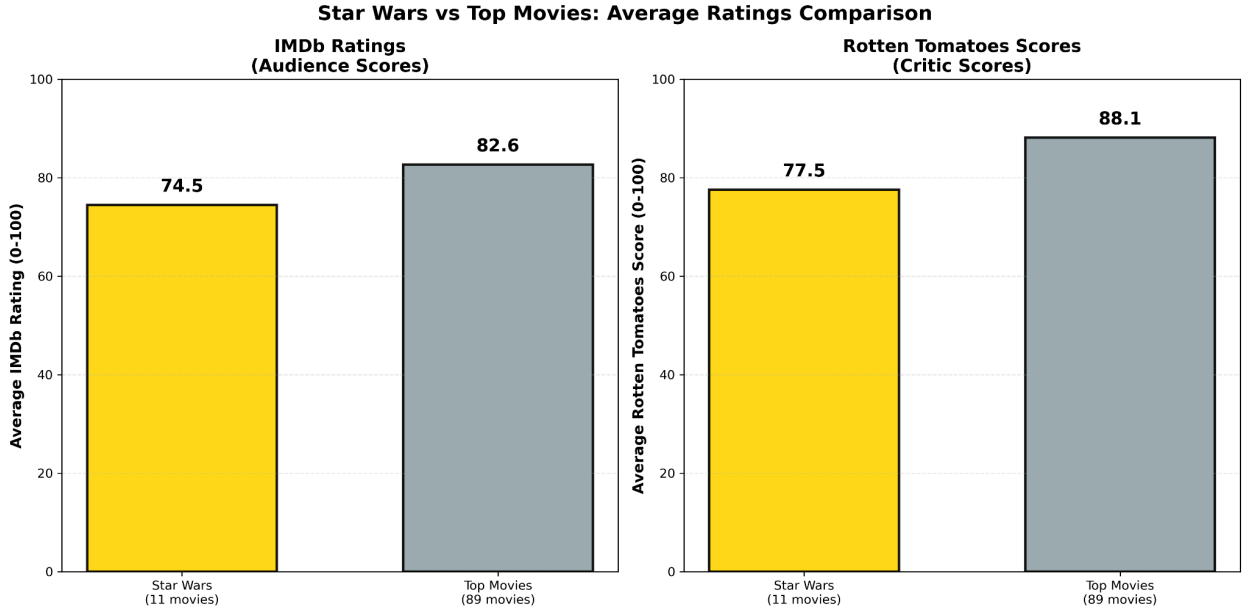
calculation_results.txt

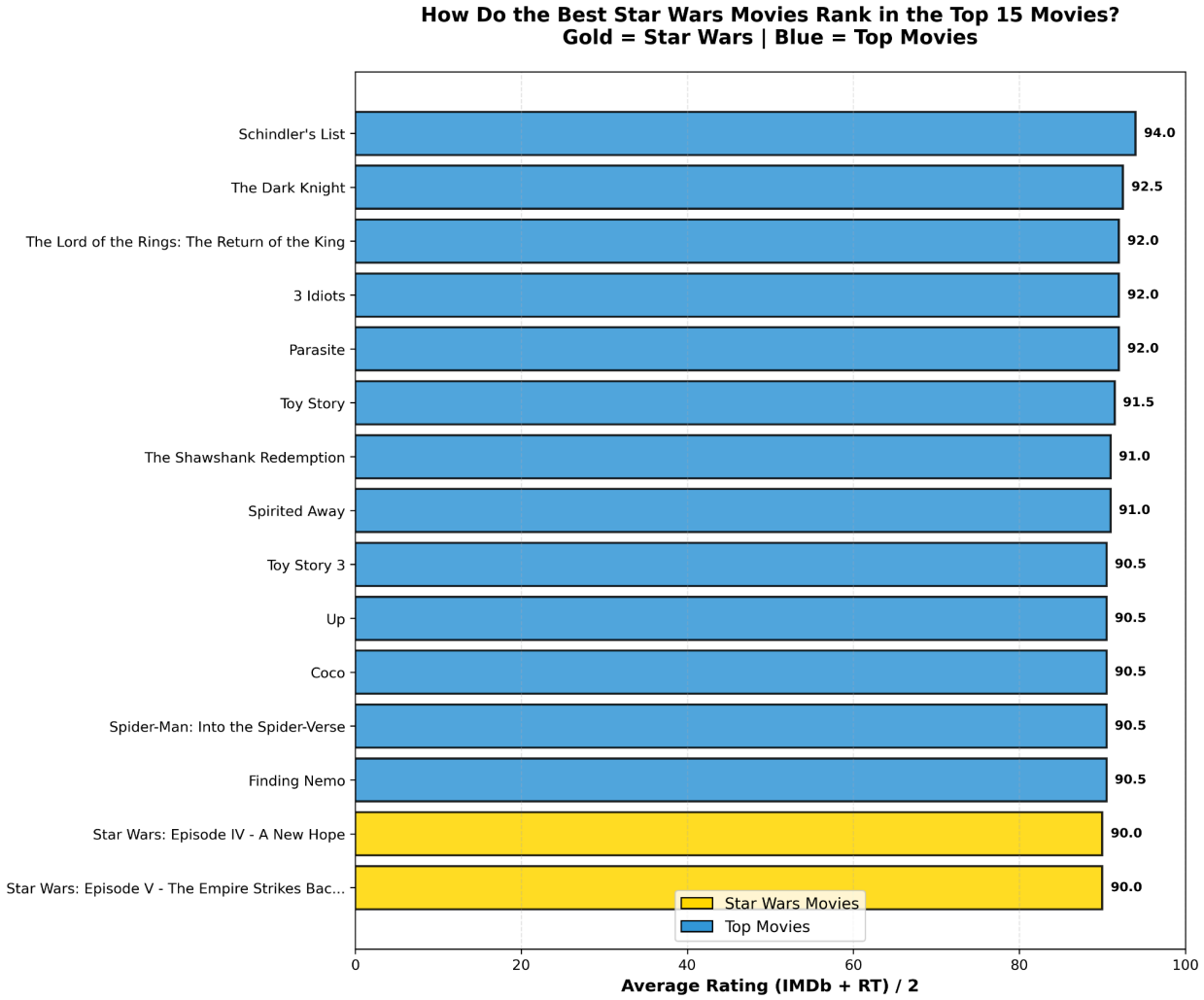
```
TOP MOST COMPLEX LEGO SETS (BY PART COUNT)
--------------------------------------------------------------------------------
 1. Diagon Alley (Set 10217-1, 2011) - 2029 pieces
 2. Motorized Hogwarts Express (Set 10132-1, 2004) - 713 pieces
 3. Hogwarts Castle (Set 4709-1, 2001) - 696 pieces
 4. Rocket Launch Center (Set 3368-1, 2011) - 499 pieces
 5. Hogwarts Express (Set 4708-1, 2001) - 414 pieces
 6. Diagon Alley (Set 40289-1, 2018) - 374 pieces
 7. The Monster Book of Monsters (Set 30628-1, 2020) - 320 pieces
 8. Hagrid's Hut (Set 4707-1, 2001) - 298 pieces
 9. Space Shuttle (Set 3367-1, 2011) - 231 pieces
10. Forbidden Corridor (Set 4706-1, 2001) - 228 pieces

================================================================================

TOP 10 LEGO THEMES BY COMPLEXITY (AVG PARTS)
--------------------------------------------------------------------------------
Theme Name                                Avg Parts  Set Count
--------------------------------------------------------------------------------
Harry Potter                              235.2      30
City                                      65.6       30
Technic                                   61.4       30
Star Wars                                 49.8       30
Ninjago                                   12.5       30
```

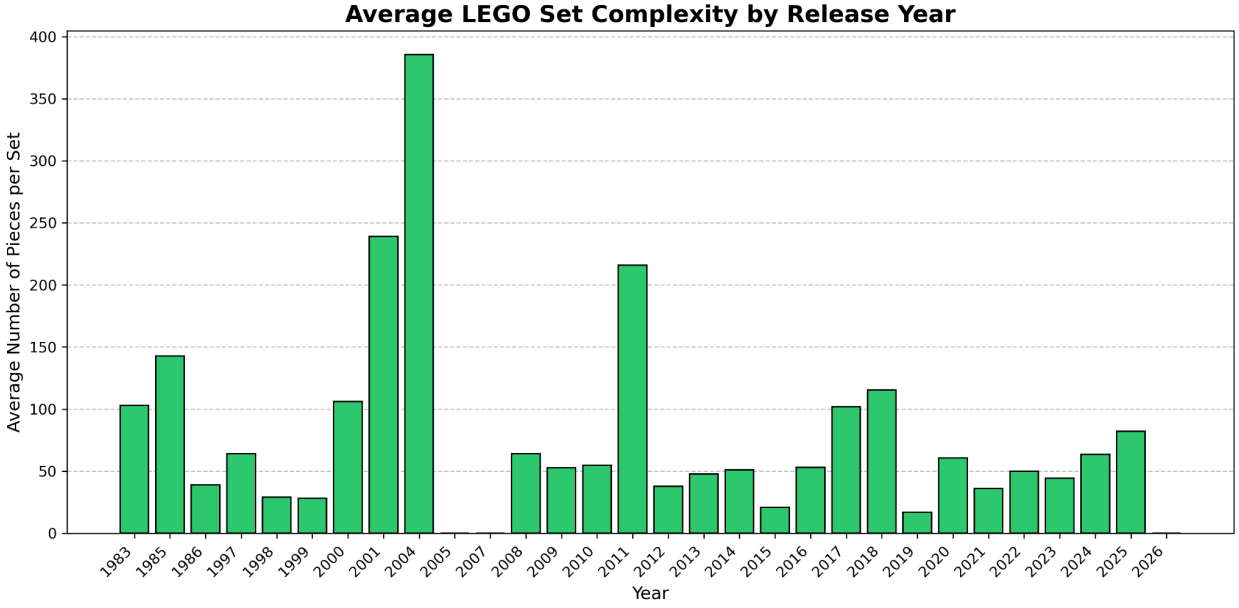
5.. Visualizations Created




Star Wars: Do Audiences and Critics Agree?
Positive = Audiences rated higher (IMDb) | Negative = Critics rated higher (RT)
Audiences liked more
Critics liked more
Rating Difference (IMDb - Rotten Tomatoes)
+13.0
+11.0
+4.0
+0.0
+0.0
-3.0
-6.0
-6.0
-8.0
-16.0
-23.0
EP IX - The Rise of Skywalker
EP I - The Phantom Menace
EP II - Attack of the Clones
EP VI - Return of the Jedi
Solo: A Star Wars Story
EP III - Revenge of the Sith
EP V - The Empire Strikes Back
Rogue One: A Star Wars Story
EP IV - A New Hope
EP VII - The Force Awakens
EP VIII - The Last Jedi
Star Wars Movies


Star Wars vs Top Movies: Average Ratings Comparison
IMDb Ratings
(Audience Scores)
Average IMDb Rating (0-100)
74.5
82.6
Star Wars
(11 movies)
Top Movies
(89 movies)
Rotten Tomatoes Scores
(Critic Scores)
Average Rotten Tomatoes Score (0-100)
77.5
88.1
Star Wars
(11 movies)
Top Movies
(89 movies)


How Do the Best Star Wars Movies Rank in the Top 15 Movies?
Gold = Star Wars | Blue = Top Movies
Schindler's List
The Dark Knight
The Lord of the Rings: The Return of the King
3 Idiots
Parasite
Toy Story
The Shawshank Redemption
Spirited Away
Toy Story 3
Up
Coco
Spider-Man: Into the Spider-Verse
Finding Nemo
Star Wars: Episode IV - A New Hope
Star Wars: Episode V - The Empire Strikes Bac...
94.0
92.5
92.0
92.0
92.0
91.5
91.0
91.0
90.5
90.5
90.5
90.5
90.5
90.0
90.0
Star Wars Movies
Top Movies
0
20
40
60
80
100
Average Rating (IMDb + RT) / 2


Average LEGO Set Complexity by Release Year
Average Number of Pieces per Set
0
50
100
150
200
250
300
350
400
1983
1985
1986
1997
1998
1999
2000
2001
2004
2005
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
Year

6. Instructions for Running Our Code

1. Sign up for OMDB API key here: https://www.omdbapi.com/apikey.aspx
2. Create a Rebrickable account for an api key here: https://rebrickable.com/api/
    a. Once an account is created, generate an API Key from your profile's settings page.
3. Add API keys to the *api_keys.txt* file. **Make sure to leave a space between the colon and the API key.** Our API keys are included below.
    a. rebrickable: ab31658305e83ef2ca0056f466bf7f74
    b. omdb: 111e1393
4. Run database_setup.py
5. Run **each** collect_().py file *at least 5 times*, located in the collection_files folder.
    a. collect_lego.py
    b. collect_omdb.py
    c. collect_wookiepedia.py
6. Run calculations.py
7. Run visualizations.py

The database and calculation results files are saved directly into the root folder, while the PNG visualization files are saved to the visualizations folder, and PNG visualizations are saved directly into the project folder.

IMPORTANT NOTES:

- **collect_wookiepedia.py:**
    - While string data for comic names look very similar to one another, they are unique. The reason for them looking so similar is that wookiepedia counts each edition of a comic as a separate entry. When the table is created, the "unique" keyword is used for the name column.
- **collect_OMDB.py:**
    - Movie names are hardcoded, which is a limitation of OMDB.
    - There might be duplicate movies in the "top movies" list from the *get_top_movies* function. However, duplicate movies are **NOT** added to the database.
    - Although the *insert_into_database* does 100 API calls every time the file is run, it only inserts 25 new rows into the database. This satisfies the project requirements.

7. Function Diagram

- Function diagram (image too large to include in report).

8. Resource Documentation

Date	Issue Description	Location of Resource	Result (did it solve the issue?)
12/11/2025	Ava - My visualization for Lego complexity wasn't displaying properly because my dictionary data structure wasn't sorted and values were incorrect.	ChatGPT	ChatGPT walked me through debugging, helped me calculate per-year averages, and helped my matplotlib visualization run correctly.
Most of the days Working on the project	Ava - For debugging and simple questions	ChatGPT	Helped me solve easy problems where sometimes I could not identify the error
12/14/25	Ava - Had problems with lego duplicate string data	ChatGPT	Helped me figure out how to get rid of the duplicates!
12/8/2025	Emma - Duplicate string data in the lego table, need a way to get themes.	https://rebrickable.com/api/v3/docs/?key=710ebabd74df135f607ce1dcfc30851f	Helped me understand what the API request needs, and that we could hardcode a few themes for comparison.
11/23/2025	Emma - not enough data for vehicles & starships.	https://swapi.info/documentation	I realized I cannot use this API.
11/21/2025	Emma - not enough data for people & planets.	https://swapi.dev/documentation	Realized I cannot use this API

Date	Issue Description	Location of Resource	Result (did it solve the issue?)
Most days working on the project	Emma - It would be unfair of me to not disclose the use of Google Gemini for debugging.	Google Gemini	Helped me solve many problems, but also created a few!
12/8/25	Kamila - OMDb required hardcoding IDs because it lacks a "sort by rating" endpoint.	https://www.omdbapi.com/	Didn't solve the issue, but justified our approach to hardcode the movie list like in HW 6
Most days	Kamila - debugging and structuring code, must disclose this bad boy :)	Claude	Yes! it was very helpful especially in debugging windows specific issues
12/06/25	Kamila - Extra movie in the database, had to run extra code to identify the movie	Deep AI	Yes! Was able to find the duplicate movie and recollect them into the database