MoViz: An Analysis and Visualization of Movie Data

Sumedh Mohile
Rutgers University
Piscataway, NJ, USA
sumedh.mohile@rutgers.edu

Vipul Gharde
Rutgers University
Piscataway, NJ, USA
vipul.gharde@rutgers.edu

Bryan Law
Rutgers University
Piscataway, NJ, USA
bryan.law@rutgers.edu

***Abstract*—We have built a web application to visualize multiple plots describing the movie trends and various relationships between movie cast and production. By utilizing and analyzing these plots, we want to answer the objective question of what properties of a movie make it successful, with success being defined as high revenue or popularity. Users can set the properties and parameters of the plots to gain insights into the data. The data we are using is provided by The Movie Database API.**

***Index Terms*—Movies, actors, crew, analysis, visualization**

I. Introduction

With the rate at which new movies are produced and released, the race to capture the attention, viewership, and wallet of today's audience has never been more competitive. Add to this the fact that with streaming media, consumers have never been this spoiled for choice. Consequently, for writers and production companies, the ability to make data-driven decisions about writing, casting, and production has quickly become the need of the hour. However, without the presence of an integrated system to allow for the visualization and analysis of data about actors, movies, and production crew, producers are often forced to rely on scripts that may not connect with the audience, casting decisions that don't account for working chemistry between actors, and production crews that don't always get along.

This project aims to provide a solution to exactly this problem by aggregating, visualizing, and analyzing data about movies, actors, and crew to provide insights into how different factors play into the success, popularity, and commercial viability of a movie.

The target users of this system are:

1) **Producers**, who want to know how successful their movies could be given the genres, cast, crew, etc.
2) **Writers**, who want to write data-driven scripts based on the success of different actors in certain genres.
3) **Analysts**, who want to understand and visualize the performance and characteristics of movies over time.

II. Design Methodology

A. Data

For the utility that our project aims to provide, there is no readily available pre-constructed dataset with all the data that we need. The Movie Database (TMDB) is an online platform that provides data for over 700,000 movies with close to 30 fields and over 2.4 million people with around 15 fields [1]. This data is made available through a REST API. For our system, we use this API to build out our dataset. However, due to the size of the data and the restriction of fetching only one entity per API call, it is challenging to pull all the data from the API.

To do this, we have implemented a multithreaded data builder module that incrementally queries the API to build the dataset. Additionally, since the movie data is dynamic and changes daily, we have built the module in such a way that there is a nightly run to refresh any outdated data. This ensures that on any given day, the application always has the latest and most accurate data.


```
{
  "adult": false,
  "backdrop_path": "/fCayJrkfRaCRCTh8GqN30f8oyQF.jpg",
  "belongs_to_collection": null,
  "budget": 63000000,
  "genres": [
    {
      "id": 18,
      "name": "Drama"
    }
  ],
  "homepage": "",
  "id": 550,
  "imdb_id": "tt0137523",
  "original_language": "en",
  "original_title": "Fight Club",
  "overview": "...",
  "popularity": 0.5,
  "poster_path": null,
  "production_countries": [
    {
      "iso_3166_1": "US",
      "name": "United States of America"
    }
  ],
  "release_date": "1999-10-12",
  "revenue": 100853753,
  "runtime": 139,
  "spoken_languages": [
    {
      "iso_639_1": "en",
      "name": "English"
    }
  ],
  "status": "Released",
  "tagline": "How much can you know about yourself if you've never been in a fight?",
  "title": "Fight Club",
  "video": false,
  "vote_average": 7.8,
  "vote_count": 3439
}
```


Fig. 1. Movie Object

The normalized system schema is shown in Fig. 3. Apart from the movie and people objects, we also have genres, credits, languages, production companies, and countries. Since each movie can have multiple genres, we create a genre mapping entity that maps each movie to all its genres. This is similarly done for the movie languages, production companies,

```
{
  "birthday": "1963-12-18",
  "known_for_department": "Acting",
  "deathday": null,
  "id": 287,
  "name": "Brad Pitt",
  "also_known_as": [
    "براد بيت",
    "Бред Питт",
    "Бред Пітт",
    "Buratto Pitto",
    "Брэд Питт",
    "畢·彼特",
    "ブラッド・ピット",
    "브래드 피트",
    "براد بیت",
    "แบรด พิตต์"
  ],
  "gender": 2,
  "biography": "William Bradley \"Brad\" Pitt (born December 18, 1963) is an American actor
  "popularity": 10.647,
  "place_of_birth": "Shawnee, Oklahoma, USA",
  "profile_path": "/kU3B75TyRiCgE270EyZnHjfivoq.jpg",
  "adult": false,
  "imdb_id": "nm0000093",
  "homepage": null
}
```


Fig. 2. People Object

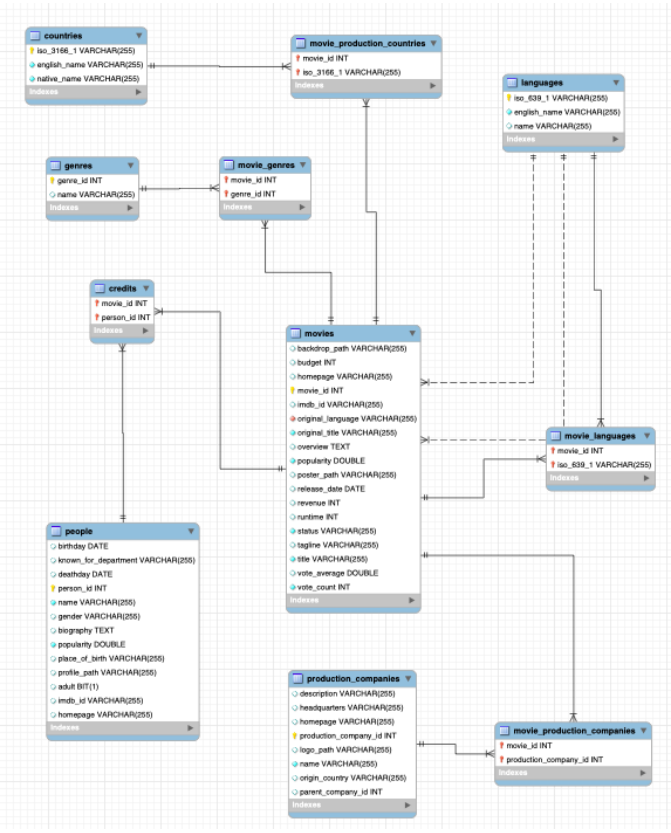

Fig. 3. Schema

and production countries. The movies and people entities are linked to each other using the credit entity.

This normalized schema provides a good balance between query performance and data duplication and is optimized for the utilities of the system.

B. Questions

The fundamental question that we plan to visualize and analyze is what properties of a movie and its production make it successful in terms of its revenue and popularity. To answer this question, we plan to analyze a number of the relationships between these different properties through visualization and interaction of complex plots. Some of the properties that we plan to generate plots and gain insights on are:

- **Genre vs. Revenue:**
  - What genres of a movie generate the most revenue throughout the evolution of movies in the past decades?
  - What trends can we see in successful genres?
- **Genre vs. Ratings:**
  - What trends can we see in the amount and value of ratings across different genres?
  - Can a movie with low ratings still be successful?
  - How have ratings affected the popularity of genres?
- **Actors vs. Movies vs. Revenue:**
  - What actors are often in the same cast together?
  - What is the effect of actors working in the same cast on movie revenue?
  - What combinations of actors result in the most revenue?
  - Does the popularity of actors affect the revenue and popularity of their movies?
- **Actors vs. Genre:**
  - What genres are actors mostly associated with?
  - What genres are most profitable for certain actors?
- **Budget vs. Revenue:**
  - Does a higher-budget movie necessarily mean higher revenue?
  - Can low-budget movies achieve high revenue and popularity?
- **Crew vs. Revenue:**
  - How does the cast of a movie production affect its revenue and ratings?
  - Which directors produce the most revenue?
  - Are there certain recurring production members associated with more revenues?
- **Duration vs. Revenue:**
  - Does the duration of a movie affect its revenue, ratings, and popularity?
  - What trends can we see in the length of movies over time?
- **Language vs. Revenue:**
  - How popular are foreign-language films in the U.S. and vice versa?

C. Modes of Processing

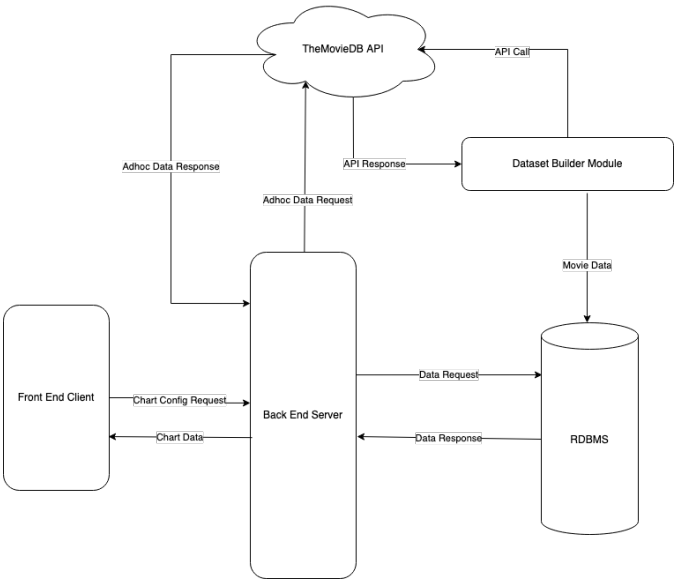

Fig. 4. System Architecture

The overall system architecture is shown in Fig. 4. The system is designed to be modular to ensure scalability in production and velocity in development.

The system requests the movie data via an API call to TheMovieDB API using a custom dataset builder module and then stores this data in the database. Since the movie data is dynamic in nature, the dataset builder module makes these calls once a day to ensure that the data present in the system is always accurate and up to date. It should be noted that TheMovieDB also publishes its updated data once a day, so the daily data refresh frequency is accurate. Additionally, since network calls are expensive with high latency, building this database is a time-consuming task. Hence, the initial phase of the project focuses on building this dataset by running the dataset builder module over a period of several days. Once the initial dataset is built, we move to the daily incremental updates and refreshes of this data. The dataset builder module has been created with performance in mind. We have built a robust, high-performance multithreaded implementation that has successfully pulled all the available data from TheMovieDB including the daily dynamic refreshes and increments.

The database we have used is MySQL, which is a Relational Database Management System (RDBMS). We have chosen to use an RDBMS since the movie dataset has a well-defined schema with an inherent relationship between the two, and the availability of SQL allows the system to run complex queries on the server.

The server fetches the data from the database, processes the results, and sends them to the client. The connection between the server and the database is strictly read-only - the dataset builder module is the only process that writes to the database. This enables a proper modular design for the application and ensures that we never have any inconsistencies in our data.

The front-end client renders all the charts, diagrams, and plots to the user, and communicates with the server using the REST API calls. On the front-end, we have used ReactJS as the base framework since its virtual DOM manipulation allows our charts to be highly interactive in real-time. For the majority of charts themselves, we have used PlotlyJS. The network graphs have been created using D3.

We have used Django as the back-end. We have also used the Django REST Framework to help communicate and exchange data with the front-end. We use the ORM system provided by Django to interact with the database. We have also implemented caching to allow for higher performance - once the database pulls data for some particular query, it is then cached so that any subsequent fetches from the client can be served from the cache with low latency. In addition to this, we have also implemented cache warming - after the nightly updates, the application makes calls to all the API endpoints so that the cache is built up immediately. This ensures that our users are never left waiting for the plots to load and have a seamless experience.

D. Visual Representation

Fig. 5. MoViz Homepage

Fig. 6. Movie Genres vs. Budget. vs. Rating

MoViz provides a dashboard web application that displays a wide variety of plots to analyze different properties related to the metrics of success such as rating, popularity, and revenue. The user is initially displayed with a homepage seen in Fig. 5 that describes the general goals of the dashboard. The user can

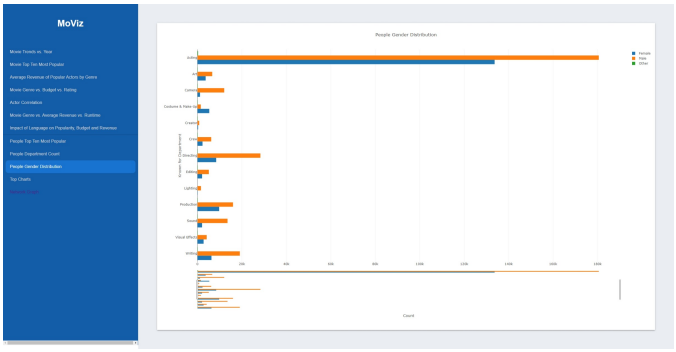

Fig. 7. Gender Distribution of People

then choose from the list of plots in the sidebar to analyze the different properties of movies and movie production. Some of the charts are displayed in Figure 7 and Figure 6.

Fig. 8. Actor Correlation

We have also provided the ability to track the correlation of the success of movies between pairs of actors. This allows users to understand how different actor pairings can affect the success of the movie in terms of revenue, budget, popularity, and ratings displayed in Figure 8.

MoViz plots a network graph where similar movies form clusters. This is done using TF-IDF on the plot of the movie and then using cosine similarity to determine the similarity between movies. This allows users to discover similar movies to those that they are analyzing. This can be seen in Figure 9.

MoViz also provides a top charts page allowing users to get a quick look at the most popular actors and the movies that lead the charts based on revenue, budget, and popularity. The MoViz top ten is created using a multidimensional ordering of movies based on their budget, revenue and popularity, allowing the users to get an idea of what the 'best' movies of all time are. This is shown in Figure 10.

E. Interactivity

Alongside the sidebar with a multitude of plots for the user to choose from, each plot has a level of interactivity that allows the user to filter certain properties to view a subset of the data where they would like to gain further insights. For example, the plots of general movie trends in Figure 11 provide the user

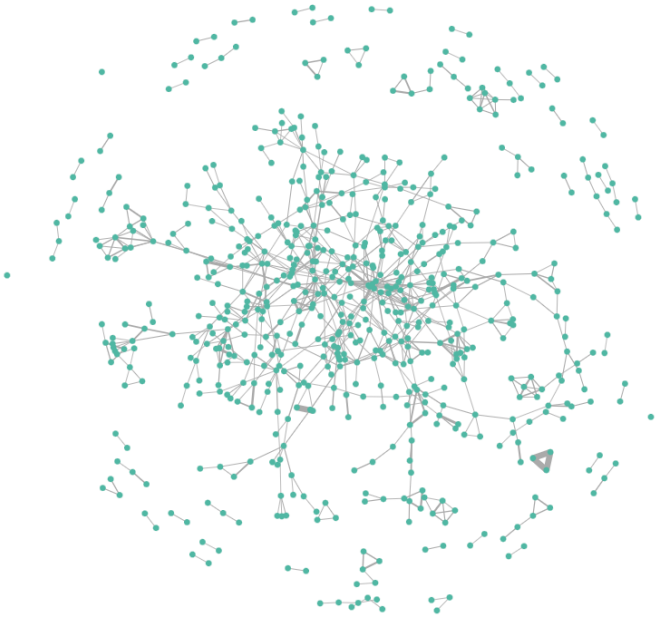

Fig. 9. Similar Movies Network Graph

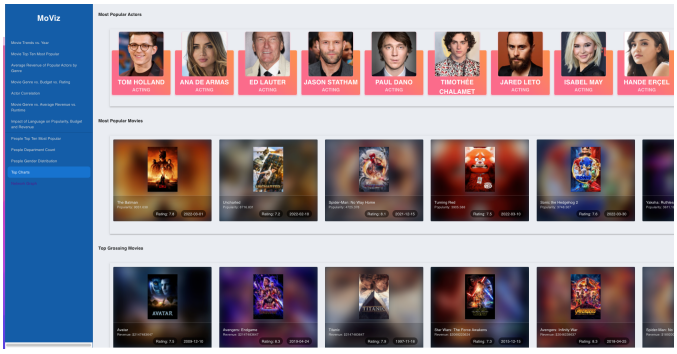

Fig. 10. Top Charts

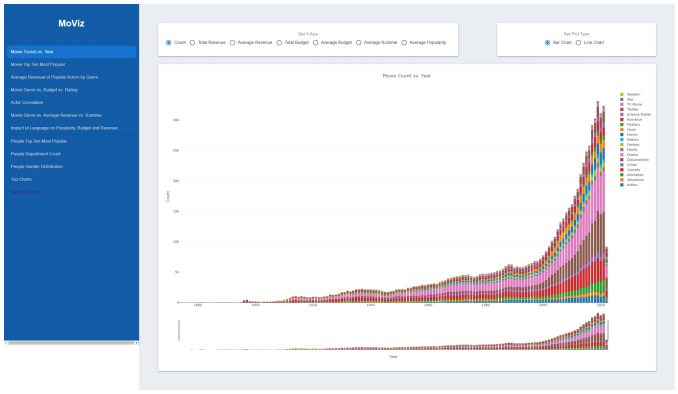

Fig. 11. Movie Trends over the Years across Genres

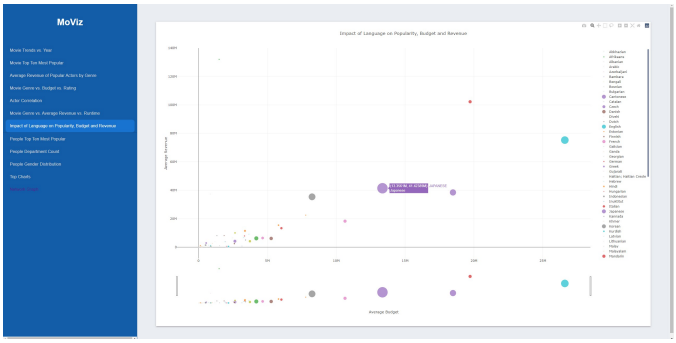

Fig. 12. Tooltip Details from Language vs. Popularity vs. Budget and Revenue

with an option to choose between the stacked bar charts and the line charts to view the trends from different perspectives. The list of different y-axis elements can be chosen for the plot to update and view the corresponding y-axis data. Every plot in the general trends page is shown over the x-axis of time, and the years can be narrowed down even further with the range selector for the x-axis, allowing the user to view specific periods of just a single year. This is an important feature to view how certain properties have trended in 'success' for the past century of movies. Furthermore, each plot in the sidebar includes tooltips to show additional information about the plot points when hovering over specific points or data in the plot as seen in Figure 12.

F. Development Documentation

The dataset builder module consists of Python [2] scripts running on an Amazon Elastic Compute Cloud (Amazon EC2) [3] instance. The database is MySQL [4] running on Amazon Relational Database Service (RDS) [5] to ensure high availability and elastic scalability of the system. The back-end server runs using Django [6], a Python web framework that is hosted on the Amazon EC2 instance. We have also used the Django REST Framework on our back-end. The front-end client uses HTML, CSS, and JavaScript [7] to display the plots and the charts to the user. To ensure a good design and high performance, we have used ReactJS as our front-end framework. The plots and charts use plotly.js [8] and D3 [9] to provide well-formed visualizations. This is rendered client-side on the user's machine.

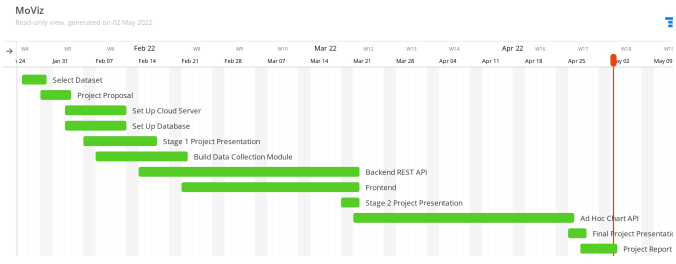

Fig. 13. MoViz Gantt Chart

We are using a Gantt chart from Instagantt [10] and a project board [11] to define and allocate our tasks, and to monitor our progress. The Gantt chart and project board are shown in Fig. 13–14.

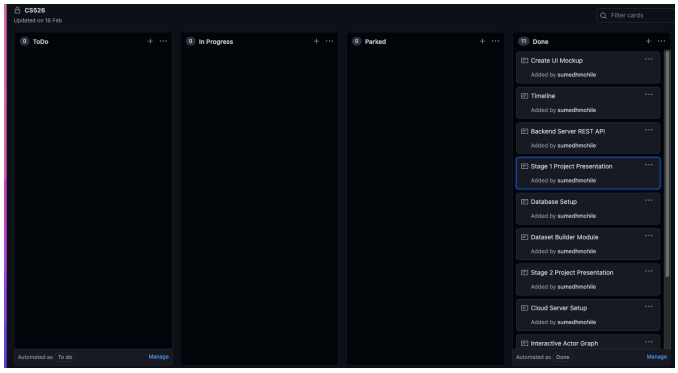

Fig. 14. MoViz Project Board

III. Conclusion

Revisiting some of the questions that we asked previously and at the beginning of the project, we can now utilize the plots created to gain some insight into the measure of success of movies.

- **Genre vs. Revenue:**
  Throughout the past two decades, the movies that generate the most revenue have been mainly the action and adventure genres. These two genres are closely related and many movies that are considered adventure are also considered action genres, which explains why both have seen such success in the past two decades. Before the previous two decades, the highest revenue genres included dramas and comedies, with action also being a high revenue-generating genre. Generally, we can also see that the margin of revenue is higher for action and adventure in the previous two decades, compared to the previous success of dramas and comedies. This could be attributed to the access to movies generally being easier in the past two decades and the integration of movies into popular culture, aided by the use of social media and technology.

- **Actors vs. Genre:**
  Actors tend to find success in popularity and movie revenue in a few genres. Across all genres, actors do not tend to find the same relative success outside of the one or two genres they are known for.

- **Budget vs. Rating:**
  There is no strong correlation between the budget and the corresponding rating and popularity of a movie. Many movies, with a high vote count in rating, fall into the average rating value of around 8 but all vary vastly in budget. Movies such as Avengers: Endgame and Back to the Future both scored an 8.3 rating but Back to the Future had a much lower budget, being made in 1985, as compared to Avengers: Endgame, being made in 2019

with much more technology, special effects, and the large cast of extremely popular actors. The rating does not seem to be affected by the budget of a movie and a lower budget does not necessarily mean a movie will not be liked by general audiences.

- **Duration vs. Revenue:**
  There is no strong correlation between the duration of a movie and the corresponding generated revenue of the movie. Although the graph shows the general distribution of movies around 150-200 minutes with the largest average amount of revenue, this can be explained by the count of movies being made within that duration was more than any other duration of time. If most movies fall under that period of time in duration, the average revenue around those times will be generally larger.

- **Language vs. Revenue:**
  The popularity of movies in different languages, according to TheMovieDB, is somewhat surprising, with English and Mandarin movies created with the most budget but Mandarin movies having less popularity but more average revenue. This can be explained by the total count of produced English movies being much higher than the number of Mandarin movies, causing the average revenue to generally be less. On the other hand, the popularity of movies in Korean and Japanese was surprisingly large, even surpassing English in the case of Japanese movies. These values are of current polarities in TheMovieDB and so can be attributed to the current release of movies in theaters.

We believe that by using the MoViz system, producers will be able to optimize their movies for the best performance at the box office. They will be able to allocate their resources more efficiently to projects that will ensure a successful showing of their movies. Writers will be able to write scripts that are more likely to succeed and also make casting decisions based on highly accurate real-world data. Overall, we believe that the MoViz application will lead to movies that are favorable to audiences and production companies alike.

IV. Future Work

In the future, we aim to expand support to TV shows and other media as well. Supporting additional media like Television, music, and theatre will allow MoViz to provide utility to a much larger audience and aggregate far more data. With this, we also hope to support the ability to analyze and visualize data across different media types, such as modeling the relationships between movies and the music they contain or understanding the popularity of actors who perform in both movies and TV shows. We believe that the addition of these features will make MoViz a one-stop shop for all performing media analysis and visualization.

References

[1] The Movie Database API. [Online]. Available: https://developers.themoviedb.org/
[2] Welcome to Python.org. [Online]. Available: https://python.org/
[3] Secure and resizable cloud compute - Amazon EC2 - Amazon Web Sevices. [Online]. Available: https://aws.amazon.com/ec2/
[4] MySQL. [Online]. Available: https://mysql.com/
[5] Amazon RDS — Clous Relational Database — Amazon Web Services. [Online]. Available: https://aws.amazon.com/rds/
[6] The web framework for perfectionists with deadlines — Django. [Online]. Available: https://djangoproject.com/
[7] JavaScript.com. [Online]. Available: https://javascript.com/
[8] Plotly javascript graphing library in JavaScript. [Online]. Available: https://plotly.com/javascript/
[9] Interactive javascript charts library. [Online]. Available: https://highcharts.com/
[10] Online Gantt Chart Software for Project Management — Instagantt. [Online]. Available: https://instagantt.com/
[11] About project boards - Github Docs. [Online]. Available: https://docs.github.com/en/issues/organizing-your-work-with-project-boards/managing-project-boards/about-project-boards