Project Proposal

1. Team Members
    a. Captain: Akarsh Vankayalapati (akarshv2)
    b. Other Members: John Armgardt (johnra2), Ian Goodwin (img3), Spencer Sullivan-Hayes (sjs7), Aditya Mansharamani (adityam5)
2. The topic we chose was an Intelligent Browsing Chrome extension that based on searching for a movie on Google will return a list of movies based on similar critic site ratings. This extension will aggregate scores from Rotten Tomatoes, Metacritic, and IMDb to generate an overall rating for a movie. Then, the extension will find the movies with the closest overall rating to the searched movie. This is a problem because users often feel overwhelmed by the surplus of movies available to them, coined "analysis by paralysis," so it will benefit users to have a way to have movies recommended to them based on searching for a specific movie. This relates to the theme and the class because it involves the pull method of browsing and document ranking based on closest matches from the searched movie and similar movies.
3. The databases we will be using are Imdb API and web scraping packages like BeautifulSoup and Selenium to get data. Doing this, we can dynamically extract data for movies to build our database, where data will be coded into an internal SQL database. The algorithm we will use is a derivation of the vector space model to compare movies with the closest movies in the vector space.
4. We will demonstrate our approach will work as expected by using a testing set that we will use to see if the movies returned were the same as what was expected and by gaining user feedback on how well they would use a tool like this to help them choose what movies to watch next.
5. The programming language we plan to use is Javascript with the possibility of having Python for back-end functionality. Because we will be developing a Chrome extension, we will be using Javascript because of the capabilities it has for this function. Moreover, because we have some members of the team who are more familiar with Python, we will try to incorporate their skills for more of the backend functions.
6. Our team is made up of 5 members, meaning there will be at least 100 hours of work to be done. Moreover, the main tasks to be completed will be creating the front-end interface and extension, creating the database and calling the APIs, creating the ranking function, and making sure all parts of the project are connected. Because of this, we will split the project into the following split: 1 member working on the front-end (20 hours), 1 member working on creating the database and calling the APIs (20 hours), 2 members working on the ranking function (40 hours), and 1 member working on connecting all the parts and testing (20 hours). Since there are many different parts involved with the project, we believe this is the best split and will take at least 100 hours to complete.