# CS410 Project Proposal – Team DJTV

## Team Members

Danh Nguyen NetID: danhn2@illinois.edu

Justin Quach NetID: jmquach2@illinois.edu

Tik On (Johnson) Lui NetID: tlui2@illinois.edu

Vamshidhar Kommineni NetID: komminen@illinois.edu

Team captain: Vamshidhar Kommineni

## Topic Overview

Building a web application for a movie search engine that combines existing data sources including reviews to help users search for and find a good movie.

## Topic Details

### Task Description

Build a web application for intelligent movie search. This will include the ability to browse and search over a large set of movies with the user being able to view structured metadata around each movie. Search over the movie metadata is a key feature that pertains to the learnings in this course. Additionally, we will implement a "similar movie" recommender functionality that the user can use to browse for interesting movies. We will also the corpus of review data from our primary dataset to build a sentiment analysis of the reviews so the user can get a quick summary in addition to reading the reviews.

### Why is it important or interesting?

The primary reason we settled on this project was around application of the Information system retrieval and analysis techniques taught in the class. This also allows us to research and learn about the different Python libraries for sentiment analysis, summarization, suggestion etc. The presence of a research data set is very helpful for us to fine tune the parameters for sentiment analysis, similar to what was taught in class. Building the project as a web application allows us to learn that set of skills since none of us are front-end web developers or have much experience on this topic through prior coursework. Lastly, working in a group gives us valuable experience on working together on a larger project and coordinating across multiple people with different skillsets.

### Planned Approach

We plan to build the project in a modular way such that we have forward progress rather than waiting for things to come together at the end. With that in mind, here is our current plan:

- Use the Stanford research data set that has a corpus of movie reviews text from IMDB as the primary dataset

- Use the popular IMDBPy API (now known as CinemaGoer) to get structured metadata (name, year, genre, cast, etc.) back on each movie within the primary dataset
- Store the data above in appropriate storage (flat files, database, etc.)
- Design a web application that can access and display data
- Build search and browse functionality into the web application
- Build sentiment analysis, summarization and other advanced features, storing the data for access by the web application

## Tools, Systems and Datasets

- Programming languages – Python (for retrieval, algorithms and backend), Javascript (for web front-end)
- Information retrieval, analysis, etc. - Python
- Web application: HTML, JavaScript, CSS, Bootstrap/Material
- Primary Dataset (Movie reviews) - http://ai.stanford.edu/~amaas/data/sentiment/
- IMDBPy/CinemaGoer API - https://cinemagoer.github.io/
- (Stretch goal/Nice to have) Server side hosting of web application – Python, Django, Heroku or AWS

## Expected Outcome and Evaluation

At the end of this project, we expect to have the following deliverables using completion and quality of each for evaluation:

- Information retrieval: Query and store (in structured storage) metadata from IMDB to complement existing dataset of reviews
- Web application: Simple HTML, CSS, and JavaScript based front-end for the movie search, browse and list experiences
- Search features: Ability to search and retrieve movies using our custom implementation of the search engine
- Similarity features: Recommender system for similar movies
- Sentiment analysis: Analysis and display of sentiment from review data for a movie

In addition, we have two stretch goals to implement if we have the time:

- Server-side implementation and hosting on a cloud platform (Heroku, AWS< etc.)
- Implement relevance and/or implicit feedback into the search engine

## Workload Justification

Our initial estimate for work which is conservative and likely on the low end is at 116 hours. The details of this estimate are in the table below. We will track and report on actual time spent over the next few

weeks in the final report. In the unlikely event that we have additional time, we have included two stretch goals at the end of the table (they are not included in the 116 hr estimate).

| Task Type | Task | Description | Time estimate in hours (initial, low confidence) |
|---|---|---|---|
| Research | Data Review | Understanding the data structure of the primary movie dataset | 4 |
| Research | Package Review | Read and understand the documentation to understand how to use Cinemagoer API | 4 |
| Design | Webpage Design | Designing a simple web application | 4 |
| Implementation | Structured data storage | Build a file-based system (e.g. simple JSON files) to store the retrieved metadata | 8 |
| Implementation | Fetch and store metadata | Fetch all structured metadata from IMDB matching the movie dataset | 2 |
| Implementation | Web Application implementation | Implement the front end for different scenarios – Browse, search, etc. | 16 |
| Implementation | Movie Search Engine – **Uses CS410 learning** | Implement a movie search engine by using user's queries to get results from the database | 16 |
| Implementation | Sentiment Analysis – **Uses CS410 learning** | Implementing a sentiment analysis classifier to detect positive or negative reviews | 16 |
| Implementation | Similar Movie Feature – **Uses CS410 learning** | Implementing a Similar Movie Finder to get similar movies in movie detail page | 16 |
| E2E scenario testing | Integration and scenario testing | Writing wrap-up scripts to run and bundle all the required scripts and test the feature set | 16 |
| Documentation | Application Documentation | Writing a user-friendly readme.md to use and test our application per grading guidelines | 2 |
| Documentation | Presentation | Preparing a storyline and recording(s) a video to demonstrate our works | 4 |
| **Documentation** | Final Report | Final report with details on design, implementation, etc. | 8 |
| Total (Through here) | | | **116** |

| | | | |
|---|---|---|---|
| **Implementation** | Stretch goal – Host web application on backend | Use something like Django and Heroku/AWS to host the application. This is pre-step to store and implement things like relevance or implicit feedback | 32 |
| **Implementation** | Stretch goal – Implement relevance or implicit feedback | Update search engine relevance with direct relevance or implicit click-based feedback | TBD |