Project Proposal

Team Members

- **Captain**: Michael Haines (mhaines2)
- Kevin Eveker (keveker2)

Topic & Motivation

Applying NLP to extract entities from queries to answer relevant questions and surface search results from an IMDB movie dataset

IMDB offers search functionality as a primary entry point for finding movies in their database. This feature does a great job at matching titles based on keyword search. For example, searching `Billy Madison` returns several titles, with the 1995 Adam Sandler classic as the top hit. This is the most relevant result, and matches what I would expect. However, the search also returns several people using a simple keyword search based on the query terms. I would expect `Adam Sandler` to be the first result, but in fact the first person is former Playboy bunny `Holly Madison`. Note that search by actor performs well - searching `Adam Sandler` returns the actor as the most relevant person, a TV special called `Adam Sandler: 100% Fresh` as the most relevant title, with `Hubie Halloween` and `Grown Ups` as the second most relevant (both of which are movies starring and written by Adam Sandler the person).

Secondly, the search feature does not really support natural language queries. For example, if I type `most popular movie from 2020` into the search bar, the search results do not answer the question at all. The top result is a 2020 podcast episode titled `What Is the Most Popular Christmas Movie in Every State?`, indicating that the search seems to be doing a simple keyword search based on the query terms.

In our project, we will use the TMDB movie dataset from Kaggle and prototype an improved search functionality. Our objective is two-fold:

- Improve the relevance of search results from keyword search for a movie by displaying the cast members for that movie (i.e. return "Adam Sandler" for the same `Billy Madison` query)
- Support natural language queries, providing a concrete results set wherever applicable

Technological Approach

To accomplish these objectives, we will use a combination of technologies to build a solution that satisfies our objectives:

- SQLite database to store Kaggle data
- GPT-3 for SQL query generation from natural language queries
- Metapy search engine with indexes for `titles` and `people`
- Python backend running on Flask server for runtime search execution
- HTML and Javascript app running in a web browser for query UX

Tasks & Effort Required

This solution will be delivered through completion of the following tasks, with the following time estimates (including investigation, design, implementation, debugging, and validation):

- `/scripts/init_db.py` - Create script to initialize SQLite database in `/data/db/` directory from Kaggle dataset - *4 hours*
- `/scripts/init_search_engine.py` - Create script to initialize search indexes in `/data/search/` directory - *4 hours*
- `/data/sql_queries.json` - Create bank of SQL queries that will be used to prime GPT-3 - *4 hours*
- `/scripts/prime_gpt3.py` - Develop script to prime GPT-3 for SQL generation - *4 hours*
- `/scripts/start_app.sh` - Create simple script to start Flask app for local development - *4 hours*
- `/src/query_handler.py` - Develop the application backend - *12 hours*
- `/src/index.html` & `/src/query.js` - Develop the HTML & JS front end - *12 hours*
- `README.md` - Document how to run the solution locally - *1 hour*
    - Initializing local Python environment with required runtime dependencies
    - (Optional) Rebuilding the SQLite database and search indexes
    - Creating and configuring an OpenAI API key for GPT-3 integration
    - Priming GPT-3 for SQL generation
    - Starting and accessing the application

Initial estimations account for more than 40 hours of work, and likely will require additional effort to stitch all of the moving pieces together and tune the query behavior based off the performance of the search engine.

Acceptance Criteria

We will evaluate the success of our solution by anecdotally assessing performance on the following queries:

- What other movies was the actor that played Jack Sparrow in?
    - Figure out that Johnny Depp plays Jack Sparrow and request list of movies with his name as keyword
- Which actress played She Hulk and what else was she in?
    - Figure out that character is played by 2 actresses (Tatiana Maslany and Malia Arrayah) and request list of movies with their names as keywords
- Who is the girl from Ex Machina and what else is she in?
    - Figure out that there were 2 primary female characters (Alicia Vikander starring role, Sonoya Mizuno supporting role), request list of movies with their names, present both to user
- What movies are the villain from Thor Love and Thunder in?
    - Figure out that character name is Gorr, played by Christian Bale, and request list of movies with his name as keyword
- Who directed best movie of 2022?
    - Figure out what the best movies are from that year (multiple awards organizations), use movie name(s) as keywords and request directors
- What is the best movie with Rachel from Friends?
    - Figure out Rachel was played by Jennifer Aniston, request list of her movies with ratings, sort to return top N
- What are top RomComs / Dramas that Adam Sandler starred in?
    - request list of his movies with ratings, figure out category of movies, sort to return top N by category

- When did Hagrid die, what was cause of death, and what other movies was he in?
    - Figure out that character played by Robbie Coltrane, request list of his movies with his personal information

Non-scope

As this is intended as a prototype, we will focus our energy on delivering the aforementioned features. We will not focus so much on:

- Improving runtime performance, beyond baseline usability of the solution
- Providing rich UX; the web page will be bare bones HTML
- Supporting dynamically changing data; the prototype will be built on a static dataset
- Providing a hosted solution; running the solution will require following the instructions documented in this `README.md`