**SDD (Software Design Description)**

**Author: Daniel Ivkovich**

1. **Abstract:**

The proposed system is a search engine that calculates movie similarity by title, genre, and plot summary, given a database of 25,000 movies.

Calculating the similarity between a single movie and all the others is a heavy computational operation to perform while the users are trying to retrieve data in real-time, therefore, a pre-processing algorithm is put in place in order to do these computations in advance and upload the results to the database.

The chosen database for this task was the NoSQL database MongoDB atlas. As the only requirement for the database is to hold a single type of object that is movie information by a title-name key, there is no need for a relational database.

In order to calculate the similarity of movies by text attributes, some NLP algorithms were used in the pre-processing stage of the project and the results were then uploaded to the database as an extension for each movie object.

A cache has been created in the search engine module in order to store requested movies and prevent recurring queries. If the cache is full, the least used searched item will be removed to make room for more relevant items.

1. **NLP Algorithms:**

**Preparing Data**

When dealing with text data we need to make some special adjustments to our dataset, such as tokenization, stemming, and vectorization.

**Tokenization**

Given a sequence of characters, tokenization is the process of breaking it in basic semantic units with a useful and basic meaning.

**Stemming**

We can reduce the inflectional forms of words into a root or base form.

In that way, the size of the vocabulary is reduced making it easier for the model to train on a limited dataset.

**Vectorization**

In order to realize any operations with text, we need first to transform the text into numbers. There are multiple ways of vectorizing text. Here we’ll be using Bag of Words and TF-IDF.

* Bag of Words(BoW): Each sentence is represented as a vector with a fixed size equal to the number of words in the vocabulary with each position representing one specific word. The value of this position is the number of occurrences of that specific word in the sentence.
* Term Frequency-Inverse Document Frequency(TF-IDF): TF-IDF consists in finding “importance” by a product of two terms: The TF term tells the frequency of the word in the document, while the IDF term is about how rarely are documents with that specific word.

**Cosine Similarity**

Since the sentences are converted into vectors, we can compute the cosine similarity between them and represent the “distance” between those vectors. The cosine similarity is calculated using the tfidf\_matrix and returns a matrix with dimensions (n\_movies, n\_movies) with the similarity between them. This is the main function that will calculate the similarity between 2 movie objects and was implemented by code. The 3 top similar movies for each movie are extracted and saved into each movie object for later usage.

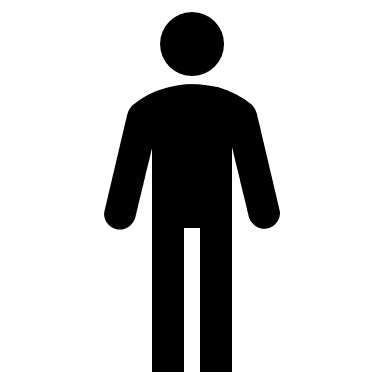
1. **Architecture:**

The system is a composition of sub-systems, each deals with a specific task. The basic flow of communication between the player and the system will be as follows:

* 1. The player writes a movie title name in the search box.
  2. The player can choose to click auto-complete for the title – this will incur a regular expression-based query to the database in order to find the best fit.
  3. The player pressed Search.
  4. The search engine reads information from the GUI search field and sends out a query for the requested movie and retrieves the relevant information including the top 3 similar movies.
  5. The search engine returns the results to the GUI for the player to see.

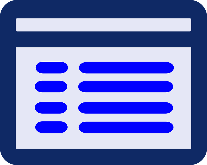
The diagram which describes this flow of high-level communication between the modules within the system is presented on the next page.

MongoDB Server



Searches for a movie

**User Interface**



Search Engine

NLP Algorithms

**System**

Output

Input

Fetch

Calculate Similarities

Update

**The main flow of the application is presented below:**

