# Milestone 4

### Route 1

**Route:** /search
**Method:** GET
**Description:** Returns all videos with matched titles with search text
**Route Parameter(s):** None
**Query Parameter(s):** search_text(string)
**Route Handler:** search(req, res)
**Return Type:** JSON Array
**Return Parameters:** { title(string), tconst(string) }
**Expected (Output) Behavior:**

- If the seach_text is provided, return *tconst* and *primaryTitle* for a videos with either primary title or the videos's AKAs matching the search text.
- If the text is empty, return all videos's tconst and primaryTitle.

### Route 2

**Route:** /type/:type
**Method:** GET
**Description:** Return basic information of all movies with the same type
**Route Parameter(s):** type(string)
**Query Parameter(s):** None
**Route Handler:** get_type(req,res)
**Return Type:** JSON Array
**Return Parameters:** {tconst(string), primaryTitle(String), startYear}
**Expected (Output) Behavior:**

- Return an array of objects with the provided text. (Type will be given in the frontend interface, thus will not be null)

### Route 3

**Route:** /filter_movie
**Method:** GET
**Description:** Returns an array of videos with all their properties matching the search query
**Route Parameter(s):** None
**Query Parameter(s):** idAdult(boolean), startYear(int), endYear(int), runtimeMinutesLow(int), runtimeMinutesHigh(int), genre(string)
**Route Handler:** filter_movie(res, req)
**Return Type:** JSON Array
**Return Parameters:** {tconst(string), primaryTitle(String)}
**Expected (Output) Behavior:**

- Case 1: if the genre is not selected, return all matching objects with all other properties matched with the query parameters
- Case 2: if the genre is selected, split the passed parameter by ",", and count the number of genres passed. If the number of genres is less than or equal to 3, return

all matched objects with search query, otherwise return empty array (each movie has no more than three genres)

### Route 4

**Route:** /distinct_genres<br>
**Method:** GET
**Description:** Return all distinct genres recorded in the database
**Route Parameter(s):** None
**Query Parameter(s):** None
**Route Handler:** get_distinct_genres(res, req)
**Return Type:** JSON Array
**Return Parameters:** {genre(string)}
**Expected (Output) Behavior:**

- Return all genres ordered by genre ascending (helper function to generate search page)

### Route 5

**Route:** /distinct_types
**Method:** GET
**Description:** Return all distinct types recorded in the database
**Route Parameter(s):** None
**Query Parameter(s):** None
**Route Handler:** get_distinct_types(res,req)
**Return Type:** JSON Array
**Return Parameters:** {title(string)}
**Expected (Output) Behavior:**

- Return all types ordered by genre ascending (helper function to generate search page)

### Route 6

**Route:** /video/:tconst
**Method:** GET
**Description:** Return all information of a videos
**Route Parameter(s):** tconst(string)
**Query Parameter(s):** None
**Route Handler:** get_video_info(res,req)
**Return Type:** JSON Object
**Return Parameters:** {StartYear(int), EndYear(int), title(string), language(string), isAdult(boolean), titleType(string)}
**Expected (Output) Behavior:**

- If a valid tconst is provided, return the relevant information from the database as specified in return parameters, otherwise the behavior can be undefined (although it is best just to return an empty object)

### Route 7

**Route:** /video_crew/:tconst

**Method:** GET
**Description:** Return all information of a crew
**Route Parameter(s):** tconst(string)
**Query Parameter(s):** None
**Route Handler:** get_video_crew(res,req)
**Return Type:** JSON Object
**Return Parameters:** {category(string), characters(string), C.primaryName(string), birthYear(int), deathYear(int)}
**Expected (Output) Behavior:**

- If a valid tconst is provided, return the relevant information from the database as specified in return parameters, otherwise the behavior can be undefined (although it is best just to return an empty object)

### Route 8

**Route:** /top5/:year/:type
**Method:** GET
**Description:** Return five highest-rated videos in given year and type.
**Route Parameter(s):** year(int), type(string)
**Query Parameter(s):** None
**Route Handler:** get_top5(res,req)
**Return Type:** JSON Array
**Return Parameters:** {primaryTitle(string), averageRating(float), numVotes(int)}
**Expected (Output) Behavior:**

- If the year and type are valid, return five videos with highest averagerating from the database dependents on year and type
- If year is valid but type is not, return five videos with highest averagerating from the database depending on year in all types
- If type is valid but year is not, return five videos with highest averagerating from the database depending on type of all time
- If both are invalid, return five videos with highest averagerating from the database of all time in all types

### Route 9

**Route:** /movie_pop_crew
**Method:** GET
**Description:** Users are able to get the movies information with the crew (actor or actress who have displayed in > 5 movies, stored as a list/array) information provided.
**Route Parameter(s):** None
**Query Parameter(s):** None
**Route Handler:** movie_pop_crew(req, res)
**Return Type:** JSON Object
**Return Parameters:** {tconst(string), Title(string), startYear(int), runtimeMinutes(int), genres(string separated by comma), averageRating(float), crew(string)}
**Expected (Output) Behavior:**

- Return all movies information with the crew (actor or actress who have displayed in > 5 movies, stored as a list/array) information.

### Route 10

**Route:** /rating_trend/:crew
**Method:** GET
**Description:** For an actor/actress, average rating of all the movies starred by the person over years.
**Route Parameter(s):** crew (string)
**Query Parameter(s):** None
**Route Handler:** rating_trend(req, res)
**Return Type:** JSON Object
**Return Parameters:** {PersonName(string), startYear(int), People(string), average_rating(float), currentAge(int), mainMedia(string separated by ";")}
**Expected (Output) Behavior:**

- If the provided string for crew name is valid, or the string is contained in crew's name, then return the average rating of the media for that crew for each year. The result contains a trend of rating for a specific crew over the year.

### Route 11

**Route:** /top1000
**Method:** GET
**Description:** Top 1000 rating work for each genre, type and year, with movie information and directors provided.
**Route Parameter(s):** None
**Query Parameter(s):** None
**Route Handler:** top1000(req, res)
**Return Type:** JSON Object
**Return Parameters:** {tconst(string), title(string), titleType(string), startYear(int), runtimeMinutes(int), genres(string), averageRating(float), directors(string separated by comma)}
**Expected (Output) Behavior:**
● Return all movie information with the directors' information with different genre, type and year groups.

### Route 12

**Route:** /pop_people_media
**Method:** GET
**Description:** Find actors and actresses who played in shows with displaying area greater than or equal to 5 and all the movie/tv series names they've played with rating greater than 6.
**Route Parameter(s):** None
**Query Parameter(s):** None
**Route Handler:** pop_people_media(req, res)
**Return Type:** JSON Object
**Return Parameters:** {PersonName(string), startYear(int), People(string), average_rating(float), currentAge(int), mainMedia(string separated by ";")}
**Expected (Output) Behavior:**

- Return a table of the actors and actresses who played in shows with displaying area greater than or equal to 5 and all the movie/tv series names they've played with rating greater than 6.