# TASK 6

The methods chosen to populate the database were scripts and APIs.

To create the database, we used 2 APIs. One for movies and one for tv series. For tv series, a loop was used to generate ids to call the TVMaze API and get information for the tv series. For movies, an IMDB database was used to get IMDB ids. Those ids were then put in an array and that array was looped through to call the OMDB API for information about movies. These were combined into the CONTENT table which had the general information. The TV\_SERIES database had series specific information, such as the series’ status and number of seasons. The MOVIES database had movie specific information such as awards, age rating and box office.

Included in the TVMaze and OMDB APIs was information about the crew, such as writers, directors and actors. While the TV\_SERIES and MOVIES databases are populated, the CREW, CREW\_ROLE and WORKS\_ON tables were being populated with the relevant information as well.

These were all done using loops so that each singular title was added fully before moving onto the next one to ensure the information pertaining to each title was relevant and accurate. The TVMaze and OMDB APIs were chosen because of their vast information. It included the relevant crew information, detailed summaries and images. Thus it allowed for the most functionality for our project as a whole as we had more information to work with.

So the CONTENT, MOVIES, TV\_SERIES tables were created using a JavaScript file to handle the loops and calls to the APIs, another API we created to populate the table and an HTML document to handle calling the functions which are in the JavaScript file.

The USERS, BANK\_ACCOUNT, INTERACTS and REVIEW tables were populated using Postman and the working API that interacted with the database.

For security purposes, each password had a salt added to it and was then hashed and the hashed password and salt were stored in the database. Due to the nature of hashing, the original password cannot be recreated from the server side, ensuring that the passwords are kept secure.

Foreign keys were added to databases to ensure relational integrity. Each user is linked to their userID and each content is linked to its contentID. This enforces the relationships found in the ER diagram and relational model.

So the USERS, BANK\_ACCOUNT, REVIEW, INTERACTS tables were created using an API we created and Postman as a workspace to make the calls to the API endpoints.