**Group Project - Report**

**Group No**: 16

**Group members**: Cai, Yi-Jia, Hassan, Omar, Smitas, Meirunas

**Introduction**

This report outlines the implementation of a Haskell program for harvesting movie details from a popular movie review website and saving it on a database.

**What does the app do?**

The “Top Movies” app retrieves the best movies of the current year up until December 2018. The app allows end users to look up movies by using different criterion's and displays the relevant output. For instance, a user may choose from one of the several options to display a movie by the director’s name. In addition, retrieve all movies by a certain rating or by a named actor.

**Use of web source**

The information is obtained from the popular review website known as Rotten Tomatoes. The format of the website is HTML. The structure of the website is fixed.

**How the information is extracted?**

The data is extracted by applying Haskell functions. Firstly, the program downloads the HTML file from the Rotten Tomatoes site. It proceeds by parsing the content for relevant information. Once the data is acquired, it is saved in a SQLite database in separate tables. The purpose of the database is to allow end users to query the data. This is achieved through access points for functions that allow querying the database to display the desired output.

**How to compile and run it?**

1. Extract the movie.zip file and save it on your desktop.
2. To build the app, open Terminal and change directory to the movie folder.
3. Type: /homes/YOUR\_ITL\_USERNAME/.local/bin/stack setup
4. Next type: /homes/YOUR\_ITL\_USERNAME/.local/bin/stack build
5. To execute the app, type: /homes/YOUR\_ITL\_USERNAME/.local/bin/stack exec topmovies-exe
6. You will be prompted with a menu screen to select from a variety of options.

**Design justification**

A many-to-many relationship exists in the database between the movie and starring table.

**Modules**

Main.hs

This is the main module where initial connection is established along with additional functions that facilitates user interaction with the app. A menu is provided allowing the user to choose from a selection and return desired results based on said selection.

Database.hs

This module enables establishing a database connection along with creating relevant tables to store the parsed data.

DatabaseQueries.hs

All queries are listed in this module which allows requesting for data from the database tables, including SELECT, DELETE, and JOIN among others. The data generated as results returned by SQL.

DownloadMovies.hs

This module makes “HTTP” request to download a copy of the website on to the disk, read the content in “ByteString” and prepare it for parsing.

MovieDataType.hs

This module outlines the custom data type that consists of: name, rank, year, rating, director, starring (actors). The use of the data type is to hold the extracted data as well as the data that is extracted from the database.

ParseResponse.hs

By applying a series of Haskell functions, the information is converted from the web source and stored in the custom data type defined in the module above.

SqliteToJson.hs

This module is responsible for dumping the database into a json file format.

**Extra features**

* Use of SELECT, DELETE, UPDATE and JOIN in SQL queries.
* Option Menu to make the app interactive with the end user.
* Multipe (3) tables to process the data.

**References**

<https://editorial.rottentomatoes.com/guide/best-movies-of-2018/>

<http://hackage.haskell.org/package/tagsoup>

<http://learnyouahaskell.com/chapters>