Backend Coding Task

*** Please note that candidates currently pursuing their graduation/studies are NOT eligible for this role. Such candidates MUST NOT submit the assignment.**

Instructions:

1. Make a GitHub public repository containing source code.
2. Create a new branch **test-task** and work in it. Make Pull Request from test-task to main branch.
3. You have to submit the Github link of the repository & a short screen recording of the application, and HTTP requests in Postman. Share the screen recording via a Google Drive link.
1. Submit the GitHub & Google Drive link, and fill in the relevant details, in this Google Form: https://forms.gle/qJrL8eLHsaVaePso7

Task:

The *data* folder contains 2 data files of 100 rows

movies.csv

tconst	titleType	primaryTitle	runtimeMinutes	genres
tt0000001	short	Carmencita	1	Documentary
tt0000002	movie	Le clown et ses chiens	5	Animation
tt0000003	short	Pauvre Pierrot	4	Animation
tt0000004	short	Un bon bock	12	Animation
tt0000005	movie	Blacksmith Scene	1	Comedy
tt0000006	short	Chinese Opium Den	1	Short
tt0000007	short	Corbett and Courtney Before the Kinetog	1	Sport
tt0000008	movie	Edison Kinetoscopic Record of a Sneeze	1	Documentary
tt0000009	movie	Miss Jerry	45	Romance
tt0000010	short	Leaving the Factory	1	Action

ratings.csv

tconst	averageRating	numVotes
tt0000001	5.7	1911
tt0000002	5.8	257
tt0000003	6.5	1716
tt0000004	5.6	169
tt0000005	6.2	2532
tt0000006	5.1	173
tt0000007	5.4	790
tt0000008	5.4	2054
tt0000009	5.2	199
tt0000010	6.9	6929

You can use any programming language & SQL database for this task.

1) Create SQL Tables `***movies***` & `***ratings***`, and populate the CSV data into them.

2) Create an HTTP server with the following routes

a) GET `/api/v1/longest-duration-movies`
This route returns as JSON the top 10 movies with the longest runTime
The output should contain tconst, primaryTitle, runtimeMinutes & genres

b) POST `/api/v1/new-movie`
This route takes JSON as input for new movie and saves it into the database
On successful save, it returns “success”

c) GET /api/v1/top-rated-movies
This route returns as JSON the movies with an averageRating > 6.0, in sorted order by averageRating
The output should contain tconst, primaryTitle, genre & averageRating.

d) GET /api/v1/genre-movies-with-subtotals
Show a list of all movies genre-wise with Subtotals of their numVotes.
The calculation of subtotals should be done in SQL query; not the API code
Output format :

Genre	**primaryTitle**	**numVotes**
Documentary	Carmencita	1911
Documentary	Edison Kinetoscopic Record….	2054
	TOTAL	3965
Animation	Le clown et ses cheins	257
Animation	Pauvre Pierrot	1716
	TOTAL	1973
. . .	. . .	. . .

e) POST /api/v1/update-runtime-minutes
Increment runtimeMinutes of all Movies using only SQL query (not in API code).
Increment runtimeMinutes by :
15 if genre = Documentary
30 if genre = Animation
45 for the rest