Exam back end developer

The qualifying process you are going through and this test, in particular, is confidential information that is part of organizational processes in Compie Technologies Ltd. Distribution, posting, or publishing of details or data related to the qualifying process or the test including, but not limited to distribution, posting, or publishing any part of the examination, oral or written, constitutes serious harm and damages to Compie Technologies Ltd. Answering the test shall be construed as consent and agreement to maintain the qualifying and the testing process in confidentiality, not distribute them or pass them to any other corporation or person, together and separately.

Time: up to 3 hours
*Using Google is allowed

In the following exam, you will be required to solve various problems utilizing all sorts of technologies.
Please make sure to spend your time wisely on each question (answer bonus sections only if you have finished questions fully), in case you are stuck it is recommended to move on to the next question.
Please remember that even if you don't have any prior experience with the following concepts try to solve them in the best possible manner.
Remember that it is important for us to see what you have accomplished and the approach you have taken for finding the solution.

Requirements & Guidelines

1. Choice of tools, frameworks, libraries, caching mechanisms, and architecture, etc. are completely free.
2. Attention should be given to clean coding best practices.
3. Please submit the task in the form of a zip file. If the program relies on any 3 rd party libraries that need to be installed manually – please either include them in the Zip file or provide a README file listing these packages.

Exam

Please solve the following tasks:

1. CSV Parser
    a. Inside the zip file, there should be a file named players.csv. It contains 9 rows with each row representing an NBA player. You should create an endpoint - http://localhost:port/player that parses and processes the CSV file line by line. Balldontlie's API (https://www.balldontlie.io/#players) should be used to acquire additional information about each player. The endpoint should return a CSV file with the player information (id, first name, last name) with any additional information derived from the API.

b. Add to the program some form of persistent caching (for example a local file in any chosen format, SQLite database, Redis, etc.)) The caching should be based on the player id.

c. Build a job that pulls the player details every 15 minutes and updates the persistent caching if necessary.

d. Create a Web Socket implementation, in any change of the players details please send a message with the Web Socket.

2. SQL **(Bonus)**

You can use https://sqliteonline.com/

a. Write a query to create a people Table:

name	width	height	birthday
Gal	65	178	1991-06-17
Shlomo	75	195	1992-09-17
Dan	65	185	1990-10-17
Tom	90	191	1994-11-17

And insert all the rows.

b. Write a query to get the name of the people ordered by their width and if they have the same width order by the height descending.

c. Write a query that returns the average height of all the people born between 1990-01-01 - 1993-01-01

d. Write a query that creates a view table with all the people that born between 1990-01-01 - 1993-01-01 And add for each person the proportion for his height to the average. Call the table proportion and the column proportion_height

Explanation: avg=182 Gal hight=178 result: 178/182

e. Write a query that joins the view with the people table and shows all the people their width bigger than 75 and their height bigger than 185 with their proportion_height if they have