...A competitive tournament with multiple participants has just finished -
the results are available.
Your task is to build a single-page application that displays the
tournament's results.

We provide you with an API to retrieve the tournament data. The tournament participants (the "players") have each achieved a score, ranging from 0 up to 150 points.

API specification:
Each player has four properties
"id" - the player's numeric identifier, its value is unique to the player
"name" - player's name
"level" - player's competition category, either a rookie, an amateur, or a pro
"score" - player's tournament result

The API is available via the following endpoints:

```
GET /api/v1/players?start=<num>&n=<num>
```

The response is a list of players in JSON format, as demonstrated by the example list below:

```
[
{
   "name": "alice",
   "level": "rookie",
   "score": 84,
   "id": 1
},
{
   "name": "bob",
   "level": "pro",
   "score": 136,
   "id": 2
}
]
```

The two obligatory query parameters are:
"start" - position of the first player to include in the response
"n" - count, number of players to include in the response

In addition to that, there are 2 more optional query parameters
supported:
"level" - filter by player level (level=rookie returns only players who are
rookies)
"search" - free search on all the player's attributes - name, level, id and score

The total count of players that fit the query filters is returned in the response header,
via the field x-total.

`GET /api/v1/players/suspects`

The response is a list, in JSON format, of player IDs - which represents the players that are suspected for cheating by the tournament's organizers.

Example response:
[34, 69, 124, 140, 218, 973, 1180]

Running the provided server:
** Linux
Extract the executable "serverLx" from the archive - then, from within the directory it has been extracted to, run *./serverLx*

Windows **
Extract "serverWin.txt" from the archive, change its file suffix from .txt to .exe - then, from within the directory it has been extracted to, run *serverWin*

The server is listening on http://localhost:20000

Adding client side files to the server:
All client side files should reside in the same folder as the server app. The server will automatically serve any file that reside under the base folder. Example (not obligatory) for a simple app structure:
/home/omer/task/serverLx
/home/omer/task/index.html
/home/omer/task/style.css

The web application :

- has to contain a table to display the results, a row for each player, with 4 columns showing all player's attributes - id, name, level and score.

- If you find that it serves the purpose of displaying the tournament's results - you may add more columns with other info, as you wish.
- the table must have pagination
- the table must have an option of filtering by level, to display only the players that have a certain level. Please make the filter selection available on the corresponding column header.
- the table should have a free search input, to allow the user to type a search phrase. The table displays only these players that the search phrase is a substring of their name, their id, their level or their score.
- Player names should always be capitalized
- Players that are suspected of cheating need to have a clear table indication - the entire row containing the player's details should be "highlighted". Freely choose how to visually create the distinguishing effect of cheaters (colors, fonts etc...) to make it clear the player is a suspect.
- In addition to the table, the page should contain the title "Tournament 101 - Final Results". Place and size it to your best judgment.

Misc:

- For testing, going to use latest Chrome and FF only. There is no need to support other or not up to date browsers.
- You can use AngularJS, React, or JQuery for your task. You can also use none of those if you prefer.
- For styling, you can use Bootstrap, including Bootstrap themes such as Material. You can also choose doing it by hand, which is OK as well. The page should have style to serve the purpose of displaying the information clearly and pleasantly.
- Keep in mind the various resolutions and window sizes that might be used (within the reasonable range, down to 720p width).

Server files:
https://www.dropbox.com/s/tvkeuapk856zrkc/server.zip?dl=0