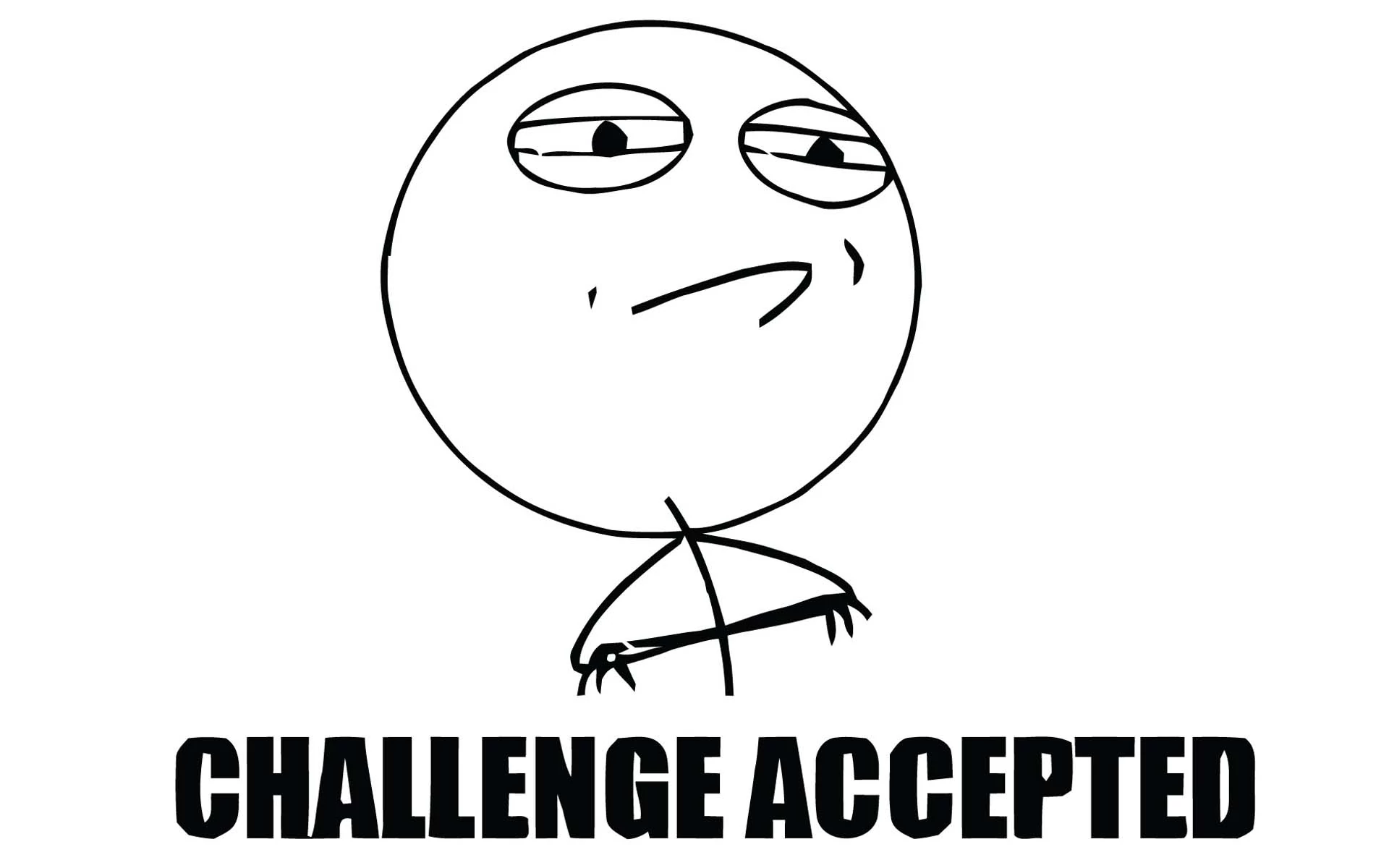
# Al Tayer Digital - Fullstack Case Study

This case study aims to assess how you approach a problem starting from the high level solution to the low level implementation details and code quality. You will not be penalized for asking questions, so don't hesitate to ask us if you need any clarification.



# 🍿 Movie Search Site

You are asked to build a simple website that has a search box where a user will be able to search for movies by name and see the list of the results. The results will be fetched from an external API on the backend. The frontend & backend should be in the same repo for the sake of simplicity.

## External API

Use OMDb API to get movie search results

API docs: <http://www.omdbapi.com/>

Example request: [http://www.omdbapi.com/?apikey=[API\_KEY]&s=[KEYWORD]&page=[PAGE](http://www.omdbapi.com/?apikey=%5BAPI_KEY%5D&s=%5BKEYWORD%5D&page=%5BPAGE)]

where

* API\_KEY: you need to get a free API key from here <http://www.omdbapi.com/apikey.aspx>
* KEYWORD: the keyword to search by
* PAGE: page number for the results, starting from 1

If you have trouble getting an API key, please reach out for us

## 🛠 Functional Requirements

### Frontend

Build a single page app with **React & Redux** with the following functionality

1. Display a search bar where a user can type a search keyword
2. Display the results as a grid of 3 columns and responsive, for each movie display the movie poster returned from API and the movie title below it

|-----------| |-----------| |-----------|

| | | | | |

| Poster | | Poster | | Poster |

| | | | | |

|-----------| |-----------| |-----------|

Title Title Title

|-----------| |-----------| |-----------|

| | | | | |

| Poster | | Poster | | Poster |

| | | | | |

|-----------| |-----------| |-----------|

Title Title Title

.

.

.

1. On mobile screens (width <= 768px), the grid will be 2 columns

|-----------| |-----------|

| | | |

| Poster | | Poster |

| | | |

|-----------| |-----------|

Title Title

|-----------| |-----------|

| | | |

| Poster | | Poster |

| | | |

|-----------| |-----------|

Title Title

.

.

.

1. You should start to fetch and display the results from the backend once the user types 3 characters or more in the search bar
2. If the user is typing, don't make any API calls until they stop typing for 300ms
3. If a user searches for the same keyword again within 30 seconds, the request should not go the server again, use HTTP cache.Consider all the possible UI states: initial, loading, error,... and present them to the user clearly. No fancy UI required
4. There are no design requirements, but we expect you to use minimal CSS to structure the page and make it a little pleasant. Follow any CSS convention you prefer like BEM, SUITCSS,... Don’t use external CSS or UI frameworks like bootstrap, bulma, ant.

### Backend

Build a NodeJS API server that will feed the data to the frontend

1. Endpoint /api/search?keyword=foo which will take the search keyword as a query param and return the matching results from the external API
2. This endpoint should return the first 20 results in 1 request, that means you need to make 2 requests to the external API as it will return only 10 results each time. Find the most efficient way to implement this and explain your solution
3. Each search result should be cached on the server indefinitely and served from the cache if requested again. Use any cache library or store if you prefer.
4. Endpoint **/api/clear** will flush the backend cache.

## 

## ✨Additional Requirements

1. README.md file explaining your high level solution and any decisions you made and the reasons behind them
2. Applicable variables should be able to be easily changed based on the environment
3. Please make sure that we are able to run your application (frontend & backend) with a single command. Using docker is preferred
4. We will assess the quality of your code. Use modern ES6+ syntax, async/await, elegant & readable code
5. Good unit tests are a big plus
6. Feel free to use any boilerplate to start the project, but mention the boilerplate you’re using and why you chose it. Also remove unused code before submitting the case study
7. Include the following in your README.md:
   * The time you spent on the case study
   * What would you change in your submission to make it production ready?
   * What would you do differently if you had more time?
8. Clean git log is appreciated but not required

## Submission

Once you have completed the solution

1. Delete any files that are not tracked in git, like: build, node\_modules, …
2. Create a zip file with the solution folder. Include .git directory
3. Upload the zip file to google drive and share the link with the recruiter
4. You are not allowed to post the solution publicly (eg: on github or bitbucket) or share it with anyone