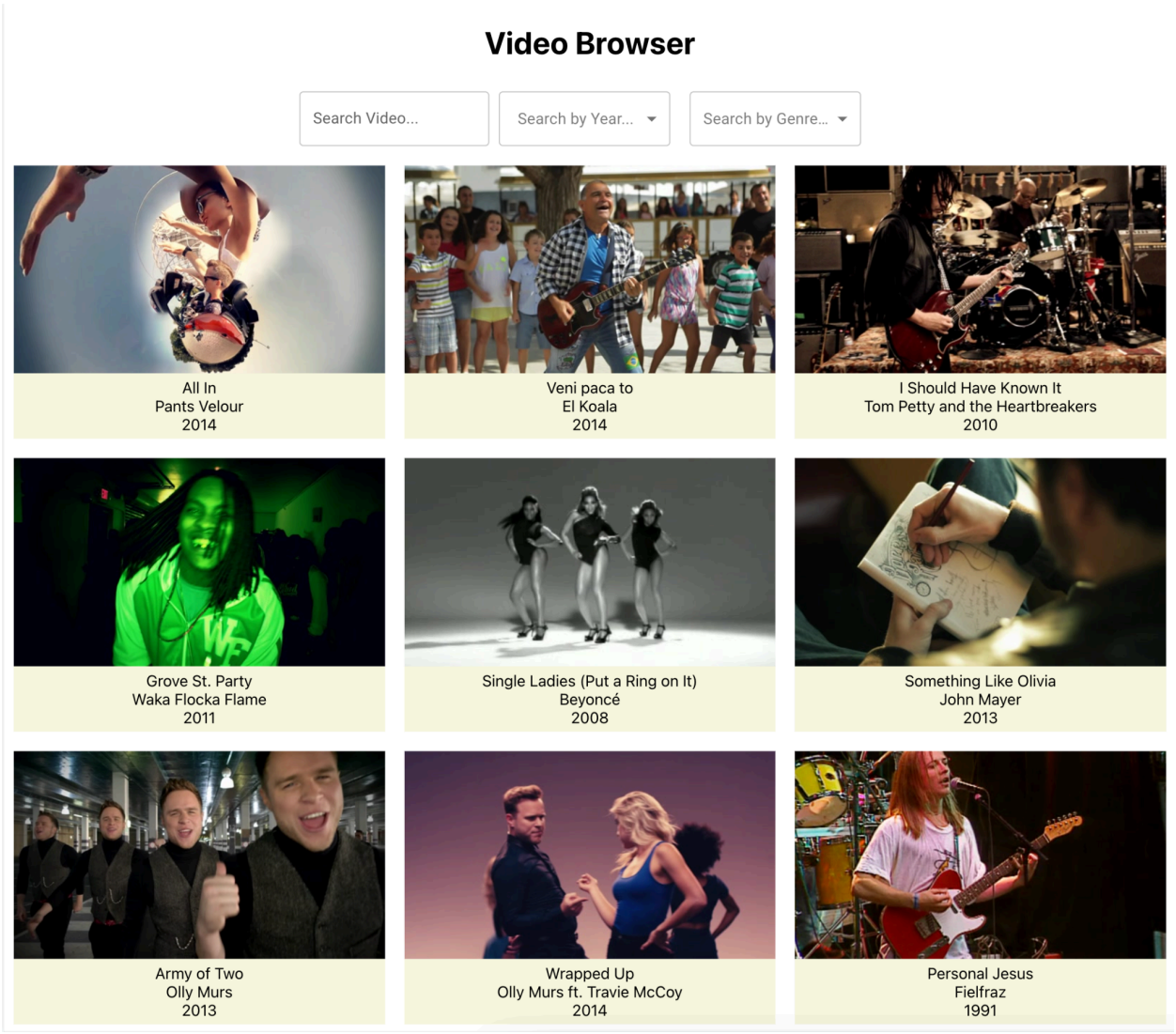
Hi,
Thanks for taking the time to work on this assignment!

This assignment is meant to assess your FE development skills, in which you'll build a **Video Browser** app, which allows a user to browse videos and filter them by parameters.

Your task is to implement this page and the components inside it according to the following design (see explanations in the next page):



The page consists of 2 parts:

1. The header panel - displaying:
    a. The app header “Video Browser”
    b. Search Video filter - a text input field (no need for autocomplete) which allows a user to filter videos by artist name **OR** title. While typing, the list will get automatically filtered (no submit button).
    c. Search by Year filter - a dropdown of available years to filter from, only one year can be selected at a time.
    d. Search by Genre filter - a dropdown of available genres, multiple genres can be selected - but you can’t select the same genre more than once.

    Important:
    - the filters can be mixed, for example one can search for “John Mayer”, year 2013 and Rock genre.
    - No need to update the dropdown filters (year/genre) according to the filtered videos by search term (But it’s a nice bonus if you do :))
    - If no videos match the filters, show a friendly message to the user saying “No videos were found”.

2. Scrollable video card list:
    a. Gets auto updated by the selection of filters
    b. Shows a minimum of 1 video per row, maximum 3
    c. Every video card shows the video image, title, artist and year (genre is only used for filtering)

You will get the video data from an API call to this URL:
https://raw.githubusercontent.com/XiteTV/frontend-coding-exercise/main/data/dataset.json

The JSON is an object which contains arrays of **genres** and **videos**, note that a video is linked to a genre via a genre ID but the genre filter will show the genre **name**.

Important: Please don’t copy the file into your project, instead use the URL as if you were calling a BE API for this data, including loading indicators and error handling.

Guidelines:

- This page must be implemented using React (advantage: implement using Typescript)
- The implementation should follow the design, as much “pixel perfect” as possible
- The implementation should be as clean, readable and efficient as possible - you should be able to explain the division into components and any other design decisions
- **You have to implement the list and video card components yourself**, but you can use an external library for the dropdowns.
- You can use any online resource you find appropriate, and reach out to us for consultation.

Good Luck!