Assessment Test

Task 1 (GroupBy)

Task should be uploaded to GitHub account.

You need to create a function, which will group an array of objects by a key.
Requirements:

1. Ability to group by any key (in the example “universe” key is used).
2. Return an error if a key is not provided.
3. Return an error if an array is not provided.
4. Return an empty object if the provided key does not exist.
5. The function should be immutable (return a new array).

Function example:

```
export const groupBy = (arr, key) => {}
```

Input data:

```
[
  { id: 1, universe: "marvel", name: "Spider Man" },
  { id: 2, universe: "marvel", name: "Iron Man" },
  { id: 3, universe: "dc", name: "Aqua Man" },
  { id: 4, universe: "dc", name: "Bat Man" },
  { id: 5, universe: "marvel", name: "Hulk" }
]
```

Result:

```
{
  marvel: [
    { id: 1, universe: "marvel", name: "Spider Man" },
    { id: 2, universe: "marvel", name: "Iron Man" },
    { id: 5, universe: "marvel", name: "Hulk" }
  ],
  dc: [
    { id: 3, universe: "dc", name: "Aqua Man" },
    { id: 4, universe: "dc", name: "Bat Man" }
  ]
}
```

Task 2 (To-Do List)

Task should be uploaded to GitHub account.

You need to implement the “To-Do List” using the provided design and the requirements described below.
General Requirements:

1. The list should have a Create function: new items are created in the text input after clicking on the “Add” button.
2. The list should have a Delete function: already created items are deleted after clicking on the “Delete” button.
3. The layout should be adaptive (both desktop and mobile versions should be supported).

Additional Requirements (nice to have):

1. Use React.js for this task.
2. Use one of the CSS preprocessors.
3. Add validation for the text input. New items should not be empty.
4. Add the “Completed” state for the list items.

- Clicking on the related checkbox should mark the item as “Completed”.
- Completed items should be moved to the “Completed” list.
- Clicking on the checkbox near the completed item should change its state back.
- Unmarked items should be moved back to the end of the list.

5. The list should have an Edit function.

- For this, you need to use the creation form (text input with related “Add” button).
- Only uncompleted tasks can be edited.
- Clicking on the “Edit” button in the related item line should move the text from it to the text input.
- The “Add” button should be changed to “Save”.
- Clicking on the “Save” button should update the line item accordingly.
- After that the text input should become empty.
- The “Save” button should be changed back to “Add”.

6. Each operation (Get/Create/Delete/Edit) should be implemented using AJAX. For this, you need to use the following API. Initial data for the list should be loaded from there. For example, when creating a new item, you need to send the POST request to the API and after the response a new item should be added.
7. Requested examples are provided here. Note: the resource will not actually be updated on the server.