CS545 WAA – Second Exam – Jan 23

In this exam you need to develop a part of React project for the Employee Management Application. Assume having an application with a Restful API for the Employee Management Application.
Based on the projects from the following repositories:

https://github.com/muhyidean/SecondExamJan23-Frontend.git
https://github.com/muhyidean/SecondExamJan23-Backend.git

You must build a Single-Page-Application by using **React Router (V6).** You need to build the following components:

1. Employee	**2. Employees**	**3. EmployeeDetails**	**4. NewEmployee**	**5. Project**	**6. ManageProjects**
7. Dashboard	**8. Header**	**9. App**			

1. Employee: This component will display two fields of the employee object. (*Name* and *ID). *It should be clickable that is associated with a link to display the **EmployeeDetails**.

Clicking on the **Employee** component will route to the **EmployeeDetails** (separate page) component that displays all fields of the **Employee** object.

2. Employees: You need to create this component to render **Employee** components.

3. EmployeeDetails: This component will be in a separate page, it will display full information about an employee (*Id, Name, Projects*) and contain the following:

i. *Delete* *-> this will delete the current employee and navigate back to the displayed employees*

ii. *Manage Projects* *-> this will display the projects in a separate page, this component is described below*

iii. *Back* *-> this will navigate back to the displayed employees*

4. NewEmployee: This component will be in a separate page and takes the input from the user and sends the data to the server, then it should navigate to the **Employees** component when submitted.

5. Project: This component could be used in the **EmployeeDetails** and **ManageProjects** component, will display two fields of the employee object. (*Name* and *ID). *If the Employee has projects, they should be printed out in the **EmployeeDetails** component. Otherwise, you should display "`No projects assigned`". Refer to the images for this part.

6. ManageProjects: This component will be in a separate page and it should fetch all projects and show which ones have been assigned or not for a particular employee. From this page, the user can add/remove an employee from a project.
- (Bonus) Optional: Make the two textfields filter the projects. They should send to a criteria query endpoint (which is the projects with query parameters for example localhist:8080/api/v1/projets?name=''&location='').

7. Dashboard: This component has the **Header** component and contains the required routes for all the components (**Employees, EmployeeDetails, NewEmployee, ManageProjects**).

i. The default route ('/') should render the **Employees** component

8. Header: This component will be holding two links that will go the following pages:

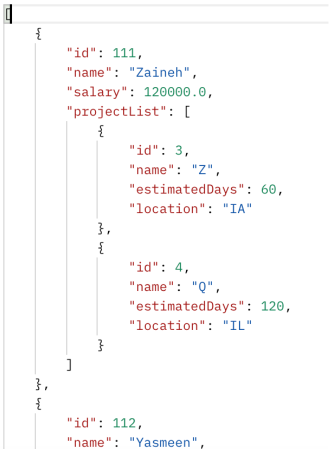
ii. Employee
iii. Add Employee

9. App: This component contains the **Dashboard** component.

YOU MAY ADD/REMOVE ANY COMPONENT AS LONG AS YOU GET THE JOB DONE

BackEnd Endpoints

- **GET** request to ‘**localhost:8080/api/v1/employees**’ returns all the employees with their associated projects in the following JSON object:



```
[
    {
        "id": 111,
        "name": "Zaineh",
        "salary": 120000.0,
        "projectList": [
            {
                "id": 3,
                "name": "Z",
                "estimatedDays": 60,
                "location": "IA"
            },
            {
                "id": 4,
                "name": "Q",
                "estimatedDays": 120,
                "location": "IL"
            }
        ]
    },
    {
        "id": 112,
        "name": "Yasmeen",
```

- **GET** request to ‘**localhost:8080/api/v1/employees /{id}**’ returns all information for one specific employee
- **GET** request to ‘**localhost:8080/api/v1/employees /{id}/projects**’ returns all projects for one specific employee
- **DELETE** request to ‘**localhost:8080/api/v1/employees /{id}**’ removes an Employee object.
- **POST** request to ‘**localhost:8080/api/v1/employees**’ persists an Employee object.

  Request Body ->

```
{
"name":"john",
"salary":120000.      }
```

IMPORTANT NOTE: (You can use these enpoints to make the project Add/Remove feature)

- **PUT** request to ‘**localhost:8080/api/v1/employees/{eid}/projects/{pid}**’ will add the project to the employee
- **DELETE** request to ‘**localhost:8080/api/v1/employees/{eid}/projects/{pid}**’ will remove the project to the employee

- **GET** request to ‘**localhost:8080/api/v1/projects**’ returns all the projects.

  - **Get projects with Criteria Query**: This will return all the projects that either have a name or location that is equal to any given input

    GET ‘localhost:8080/api/v1/projects?name=X&location=FL

Help from your professor:

* When you read the values from the filtering options. You can add the request parameters to the call using the following: *Even if there are no parameters, it will work.*

```
axios.get('http://localhost:8080/api/v1/employees', {
          params: {
              name: // [PLACE DROPDOWN DATA HERE…]
              location: // [PLACE INPUT DATA HERE…]
          }
      })
```

* When navigating from **EmployeeDetails** to **ManageProjects**, you can add the following code on your navigate command to send the ‘id’ over→ `navigate("/manage-projects", {state:{id: params.id }})`

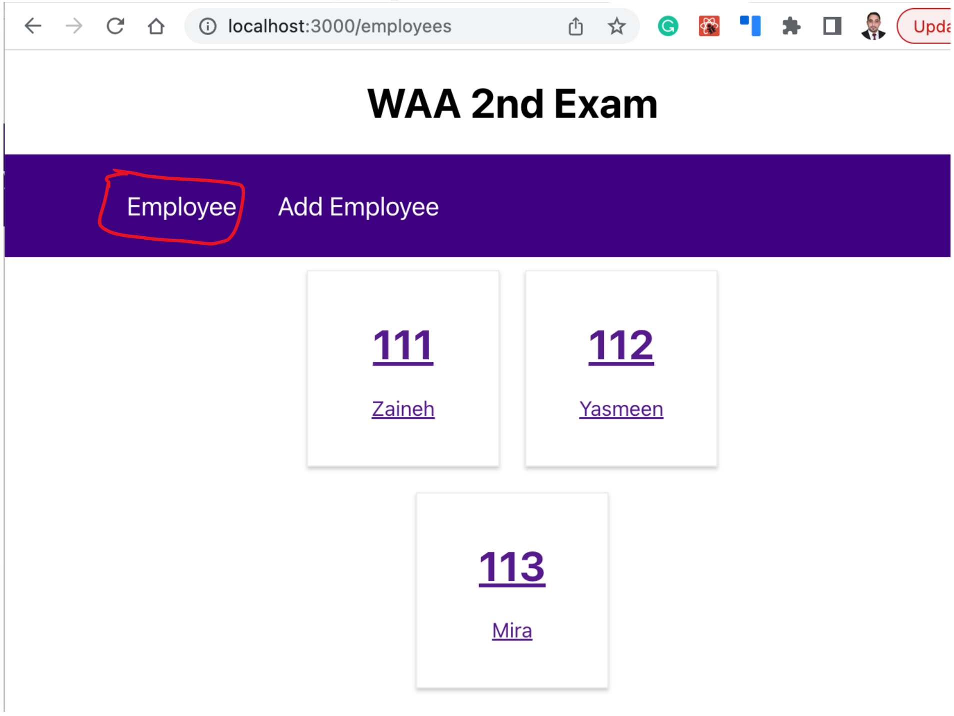
Add you can retrieve the id from adding this code in the **ManageProjects**→ `const {state}= useLocation();`

---

* You can use this for the dropdown. (Don’t forget to create a useRef() to link it.)

For more clarification, refer to the following screenshots. (You are NOT obliged to implement the same design!)

`localhost:3000/employees`



WAA 2nd Exam

Employee   Add Employee

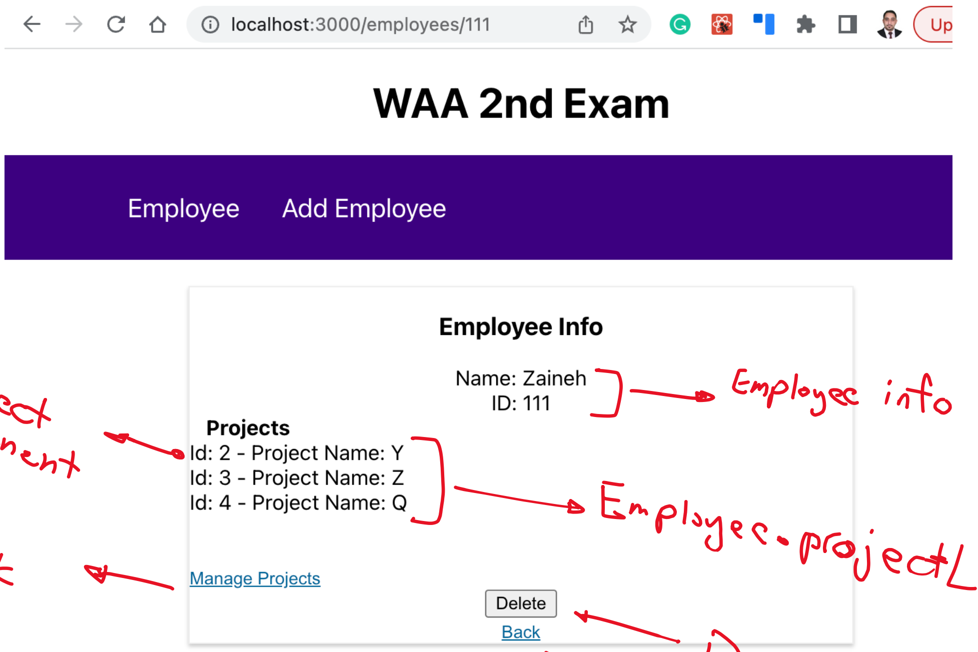
111
Zaineh

112
Yasmeen

113
Mira

should be default

`localhost:3000/employees/111`



WAA 2nd Exam

Employee   Add Employee

Employee Info

Name: Zaineh
ID: 111

Projects
Id: 2 - Project Name: Y
Id: 3 - Project Name: Z
Id: 4 - Project Name: Q

Manage Projects

Delete

Back

Employee info

Project component

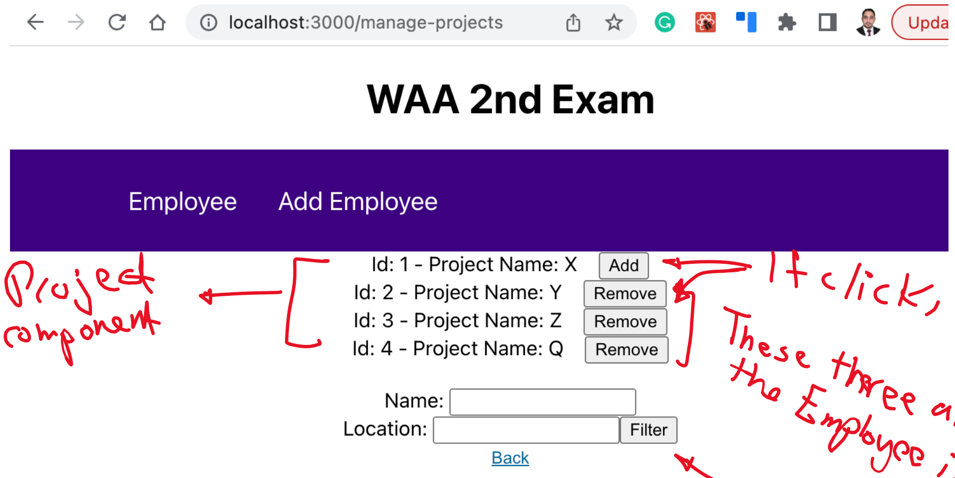
Employee.projectList

Link

Delete Employee

Link

localhost:3000/manage-projects



WAA 2nd Exam

Id: 1 - Project Name: X [Add]

Id: 2 - Project Name: Y [Remove]

Id: 3 - Project Name: Z [Remove]

Id: 4 - Project Name: Q [Remove]

Name:

Location:

Project component

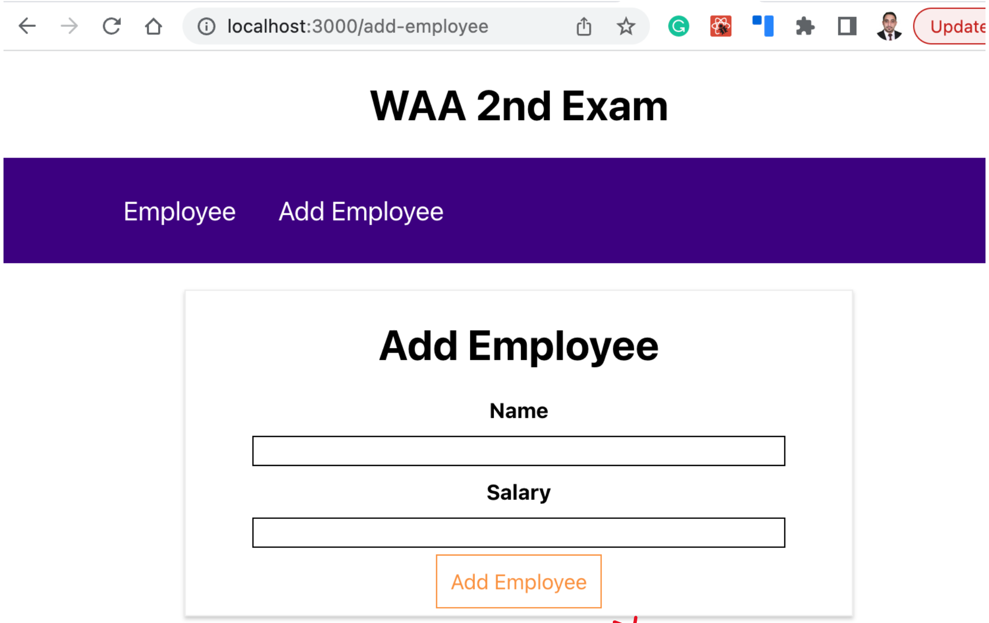
When you click from EmployeeDetails

If click, should change DB

These three are 'Remove' because the Employee is already assigned

This should filter

localhost:3000/add-employee



WAA 2nd Exam

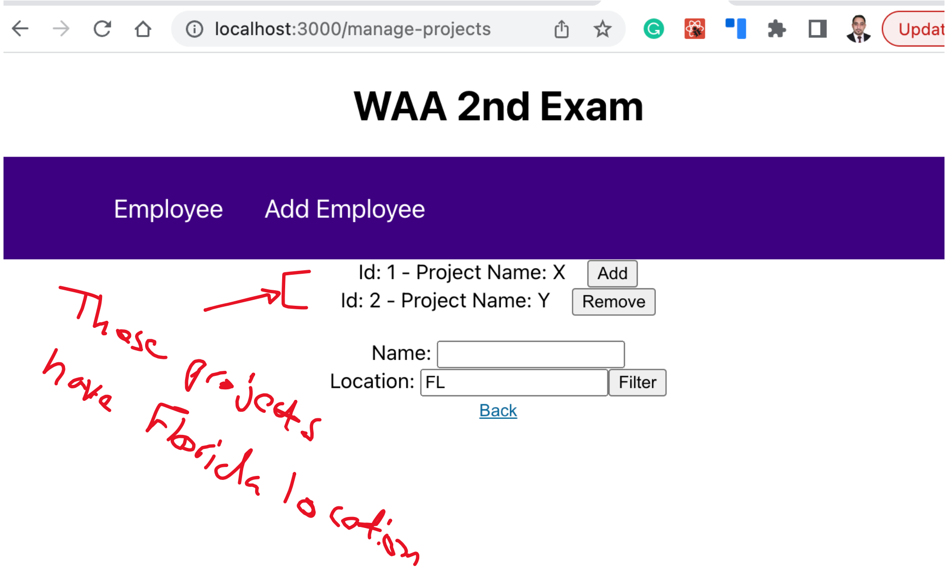
Add Employee

Name

Salary

Save to DB

Example of employees after applying filter:



localhost:3000/manage-projects

Updat

WAA 2nd Exam

Employee Add Employee

Id: 1 - Project Name: X Add

Id: 2 - Project Name: Y Remove

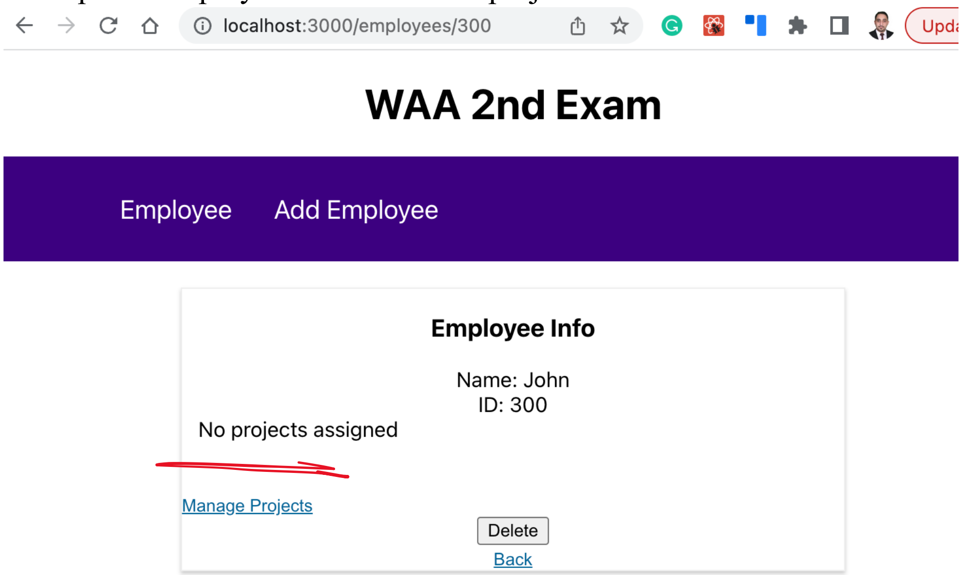
Name:

Location: FL Filter

Back

These projects have Florida location

Example of employee details with no projects



localhost:3000/employees/300

Upda

WAA 2nd Exam

Employee Add Employee

Employee Info

Name: John

ID: 300

No projects assigned

Manage Projects

Delete

Back