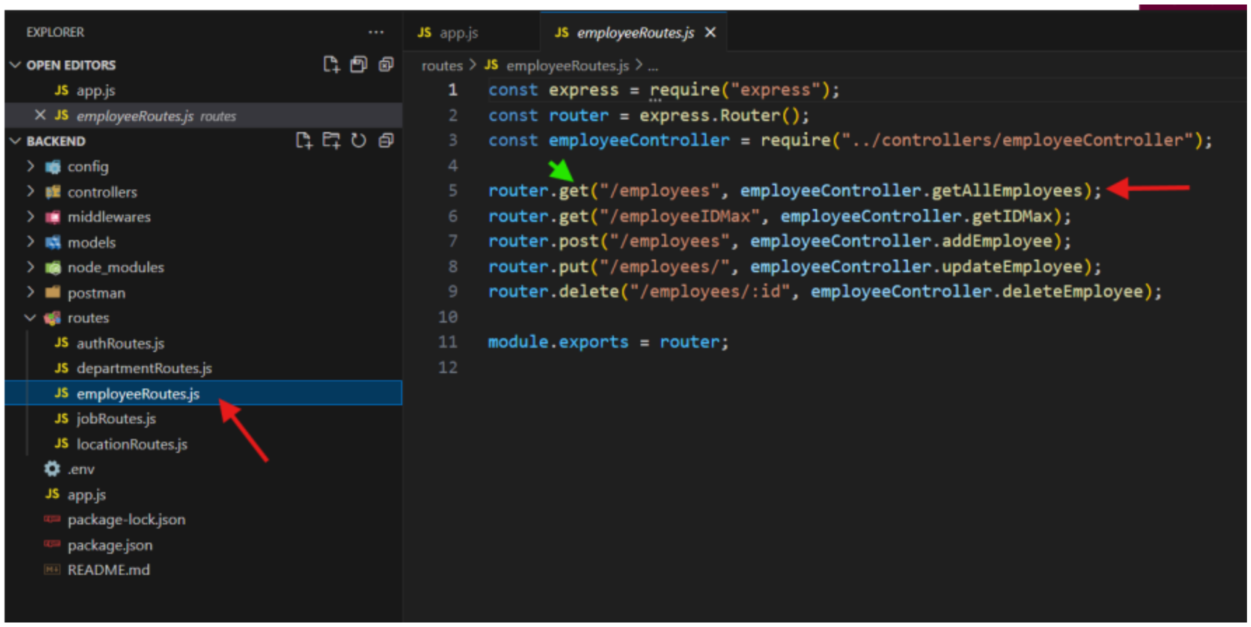
Name: zuha aqib

Erp: 26106

Task: databases lab 7

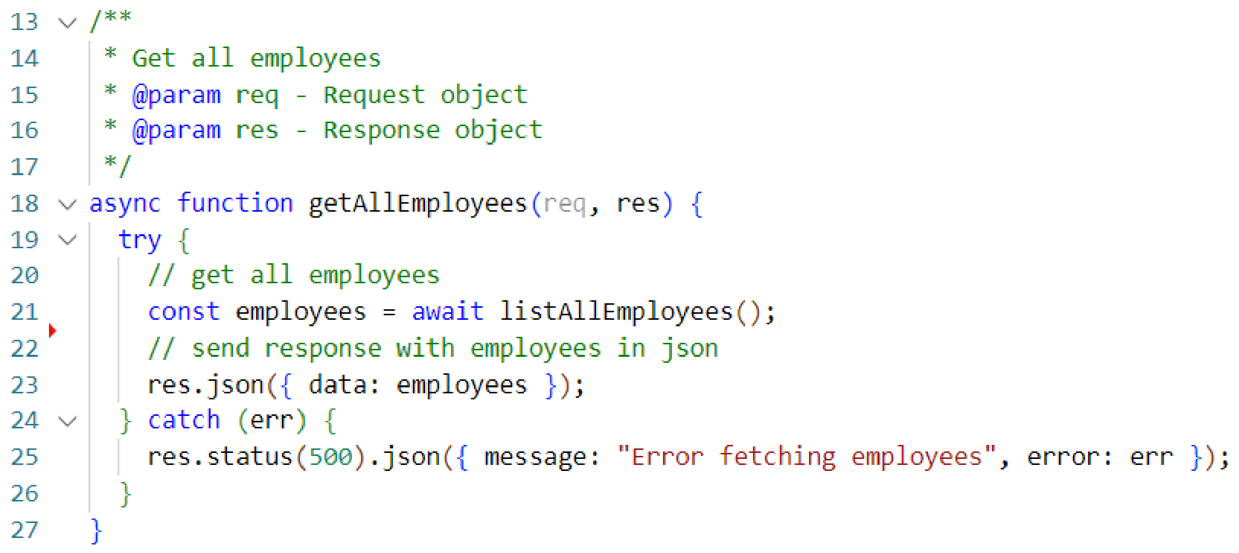
Documentation:

So to display all the employees in frontend, we have this in employeesRoutes.js,



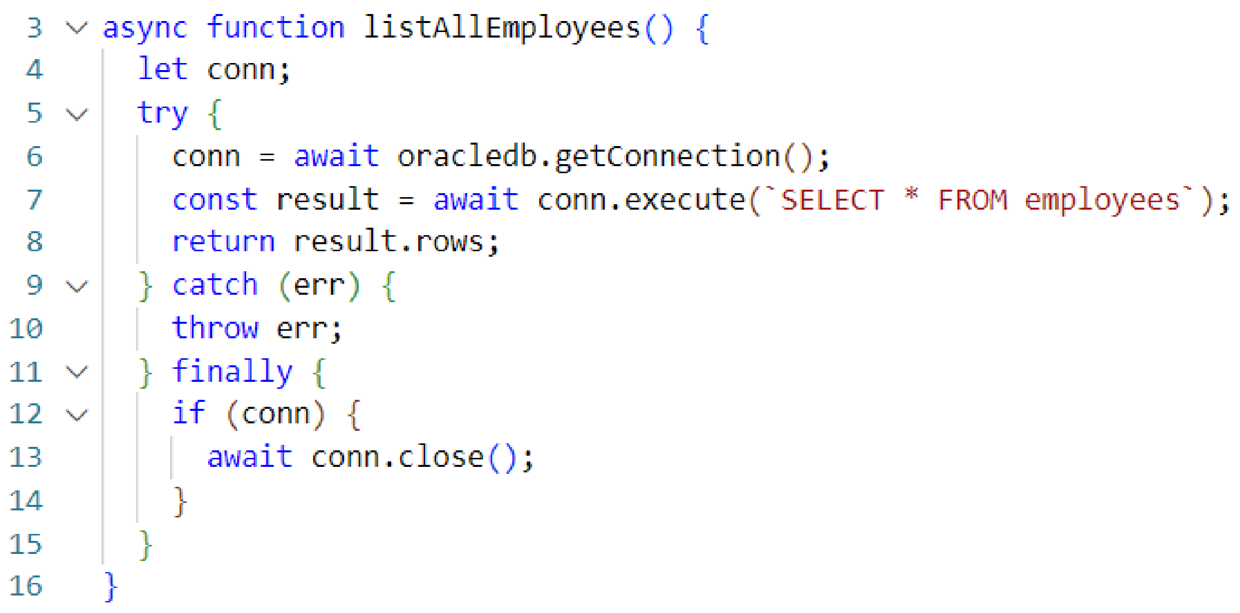
```javascript
const express = require("express");
const router = express.Router();
const employeeController = require("../controllers/employeeController");

router.get("/employees", employeeController.getAllEmployees);
router.get("/employeeIDMax", employeeController.getIDMax);
router.post("/employees", employeeController.addEmployee);
router.put("/employees/", employeeController.updateEmployee);
router.delete("/employees/:id", employeeController.deleteEmployee);

module.exports = router;
```

This means that we call the getAllEmployees function in employeeController. If we navigate to that, we have this:



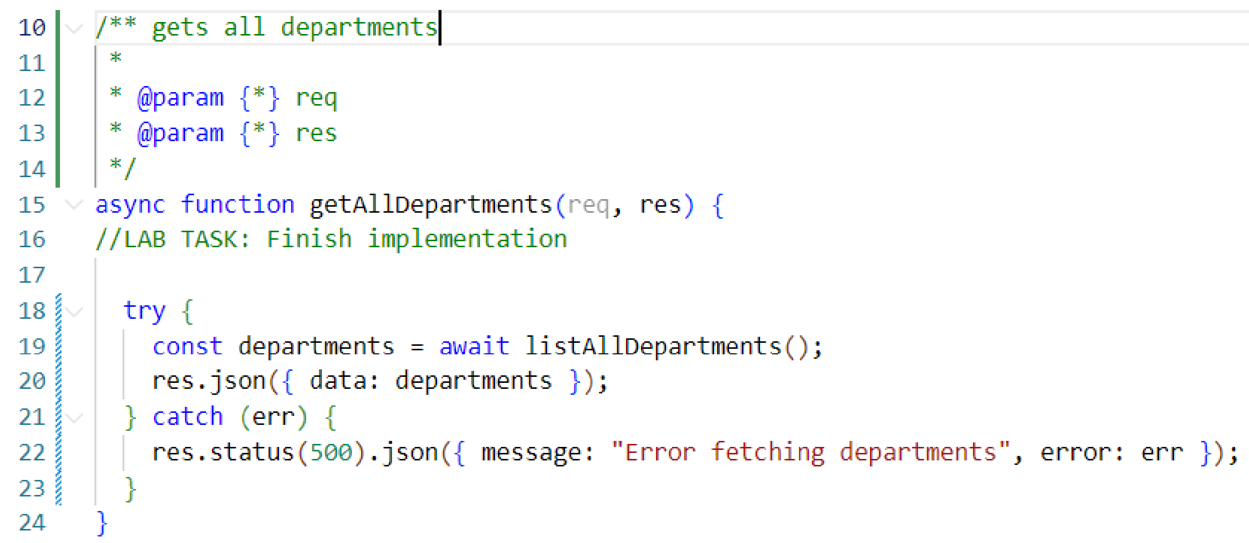
```javascript
/**
 * Get all employees
 * @param req - Request object
 * @param res - Response object
 */
async function getAllEmployees(req, res) {
  try {
    // get all employees
    const employees = await listAllEmployees();
    // send response with employees in json
    res.json({ data: employees });
  } catch (err) {
    res.status(500).json({ message: "Error fetching employees", error: err });
  }
}
```

Which gets all the employees. This also uses a function of listAllEmployees() which is in employeeModel.js,



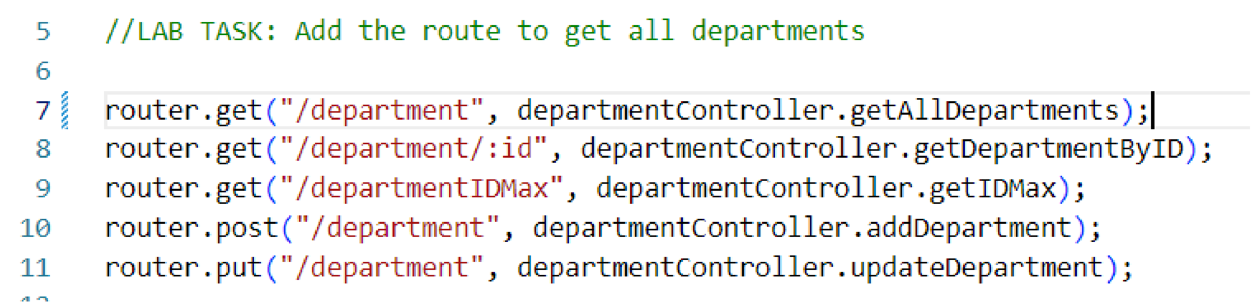
```
async function listAllEmployees() {
  let conn;
  try {
    conn = await oracledb.getConnection();
    const result = await conn.execute(`SELECT * FROM employees`);
    return result.rows;
  } catch (err) {
    throw err;
  } finally {
    if (conn) {
      await conn.close();
    }
  }
}
```

My first task is to add a similar function of getAllEmployees() in employeesController.js in backend to getAllDepartments() in departmentsController.js in backend.



```
/** gets all departments
 *
 * @param {*} req
 * @param {*} res
 */
async function getAllDepartments(req, res) {
//LAB TASK: Finish implementation

  try {
    const departments = await listAllDepartments();
    res.json({ data: departments });
  } catch (err) {
    res.status(500).json({ message: "Error fetching departments", error: err });
  }
}
```

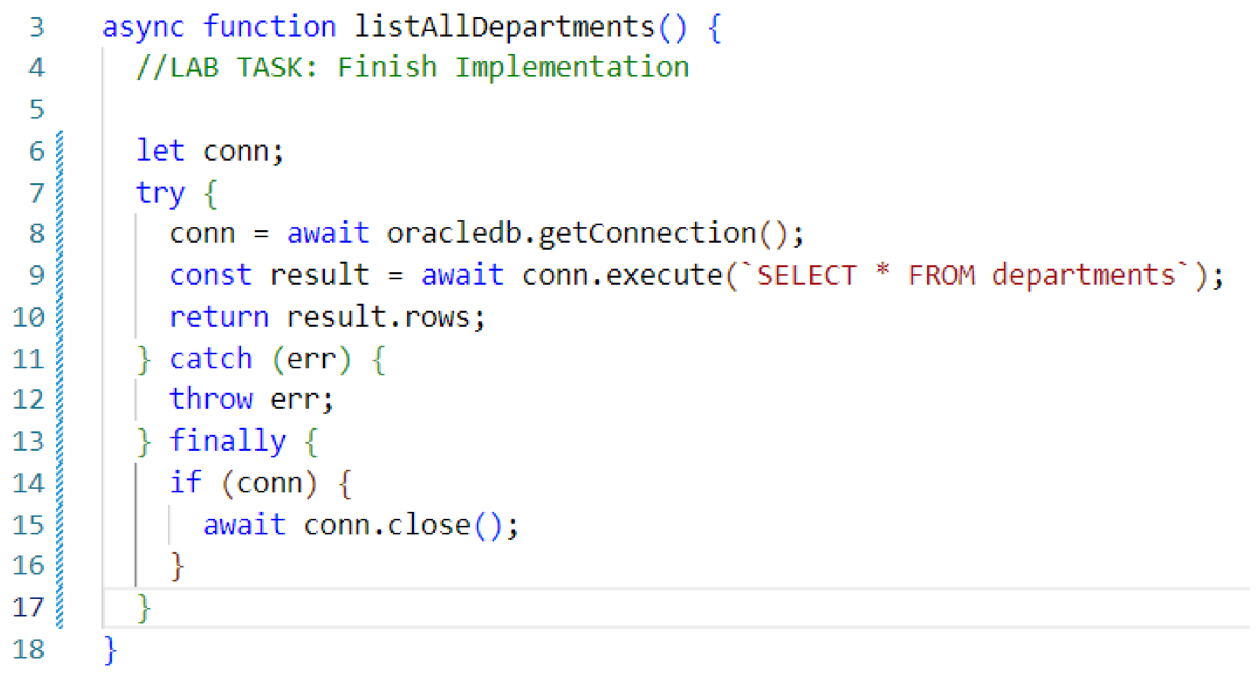
Now my second task is to call this function in departmentRoutes.js,



```
5   //LAB TASK: Add the route to get all departments
6
7   router.get("/department", departmentController.getAllDepartments);
8   router.get("/department/:id", departmentController.getDepartmentByID);
9   router.get("/departmentIDMax", departmentController.getIDMax);
10  router.post("/department", departmentController.addDepartment);
11  router.put("/department", departmentController.updateDepartment);
```

Line 7 was added

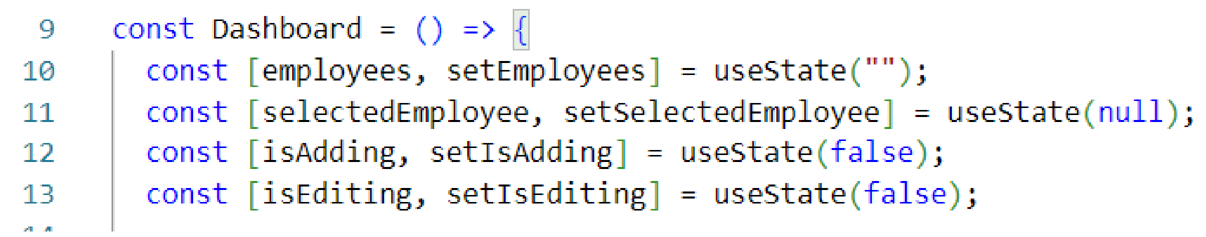
Now my third task is to edit and add a function of listAllDepartments() in departmentModel.js,



```
async function listAllDepartments() {
  //LAB TASK: Finish Implementation

  let conn;
  try {
    conn = await oracledb.getConnection();
    const result = await conn.execute(`SELECT * FROM departments`);
    return result.rows;
  } catch (err) {
    throw err;
  } finally {
    if (conn) {
      await conn.close();
    }
  }
}
```

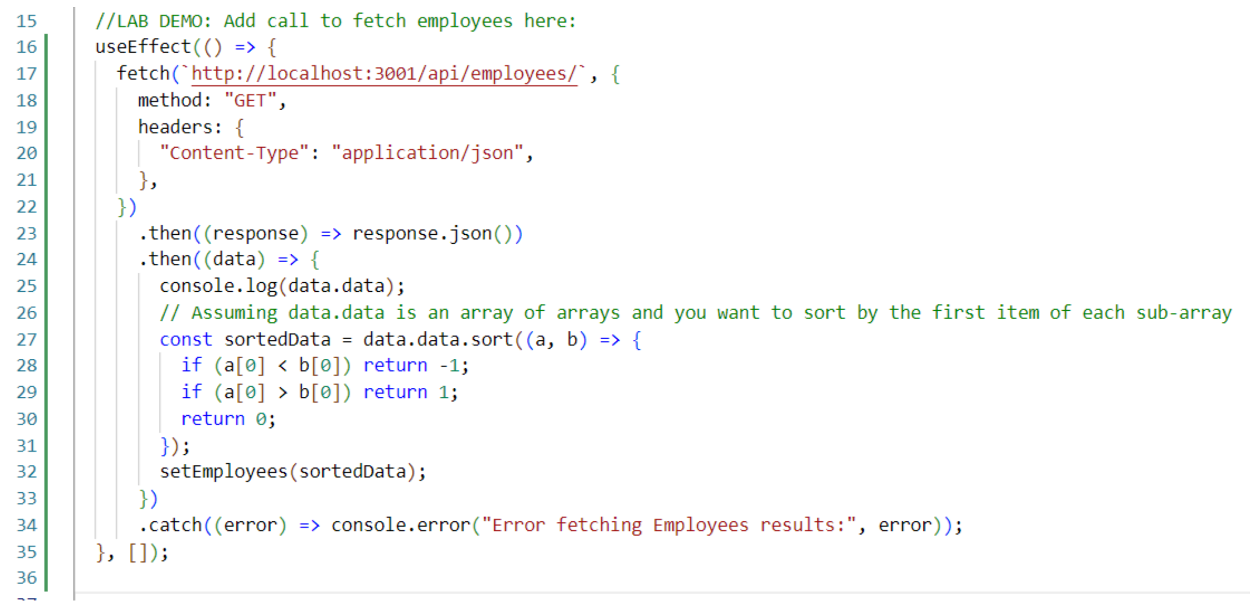
Done.

Now that backend is done, lets move to frontend. In frontend/src/components/Dashboard/employees.js, we had this code,



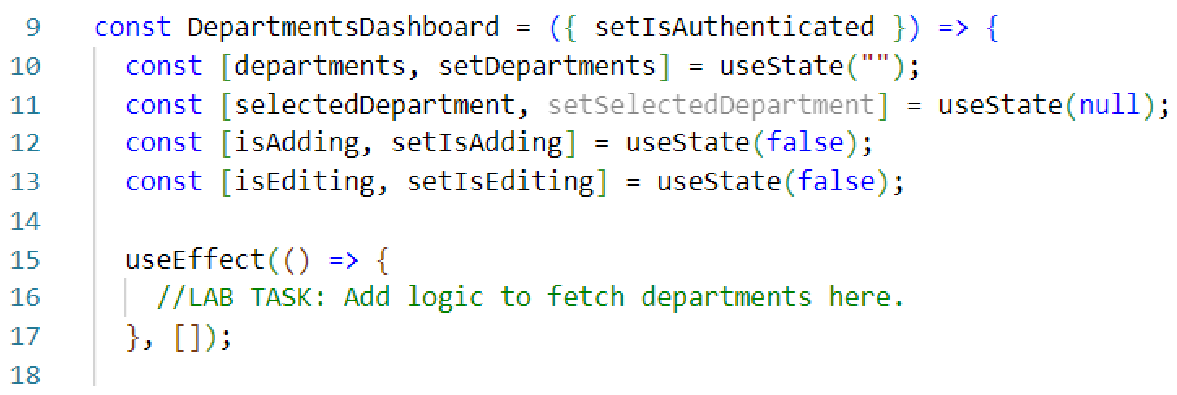
```
const Dashboard = () => {
  const [employees, setEmployees] = useState("");
  const [selectedEmployee, setSelectedEmployee] = useState(null);
  const [isAdding, setIsAdding] = useState(false);
  const [isEditing, setIsEditing] = useState(false);
```

we added this code to call the API to get and display the employees,



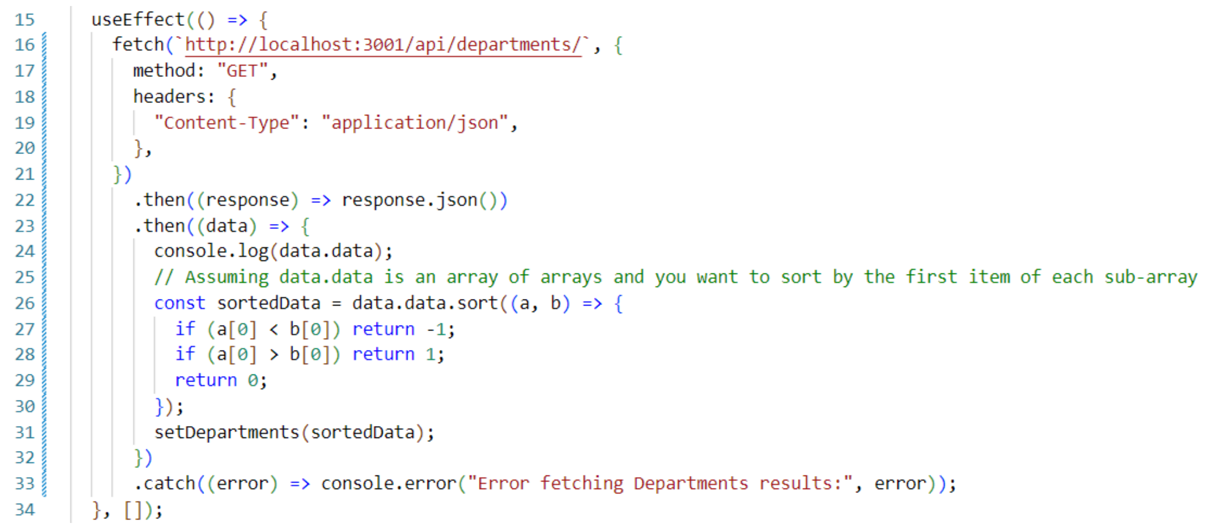
```
//LAB DEMO: Add call to fetch employees here:
useEffect(() => {
  fetch(`http://localhost:3001/api/employees/`, {
    method: "GET",
    headers: {
      "Content-Type": "application/json",
    },
  })
    .then((response) => response.json())
    .then((data) => {
      console.log(data.data);
      // Assuming data.data is an array of arrays and you want to sort by the first item of each sub-array
      const sortedData = data.data.sort((a, b) => {
        if (a[0] < b[0]) return -1;
        if (a[0] > b[0]) return 1;
        return 0;
      });
      setEmployees(sortedData);
    })
    .catch((error) => console.error("Error fetching Employees results:", error));
}, []);
```

So we will do the same in departments.js, we have this existing code,



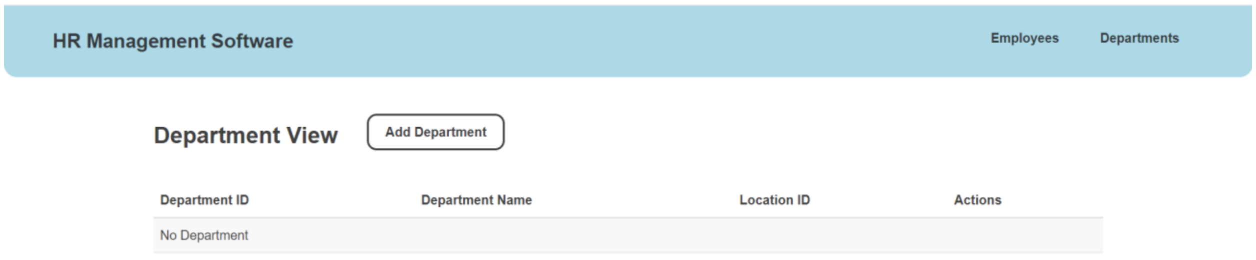
```
const DepartmentsDashboard = ({ setIsAuthenticated }) => {
  const [departments, setDepartments] = useState("");
  const [selectedDepartment, setSelectedDepartment] = useState(null);
  const [isAdding, setIsAdding] = useState(false);
  const [isEditing, setIsEditing] = useState(false);

  useEffect(() => {
    //LAB TASK: Add logic to fetch departments here.
  }, []);

```

And we have to add the useEffect() function,



```
useEffect(() => {
  fetch(`http://localhost:3001/api/departments/`, {
    method: "GET",
    headers: {
      "Content-Type": "application/json",
    },
  })
    .then((response) => response.json())
    .then((data) => {
      console.log(data.data);
      // Assuming data.data is an array of arrays and you want to sort by the first item of each sub-array
      const sortedData = data.data.sort((a, b) => {
        if (a[0] < b[0]) return -1;
        if (a[0] > b[0]) return 1;
        return 0;
      });
      setDepartments(sortedData);
    })
    .catch((error) => console.error("Error fetching Departments results:", error));
}, []);
```

But it is not displaying on the screen,

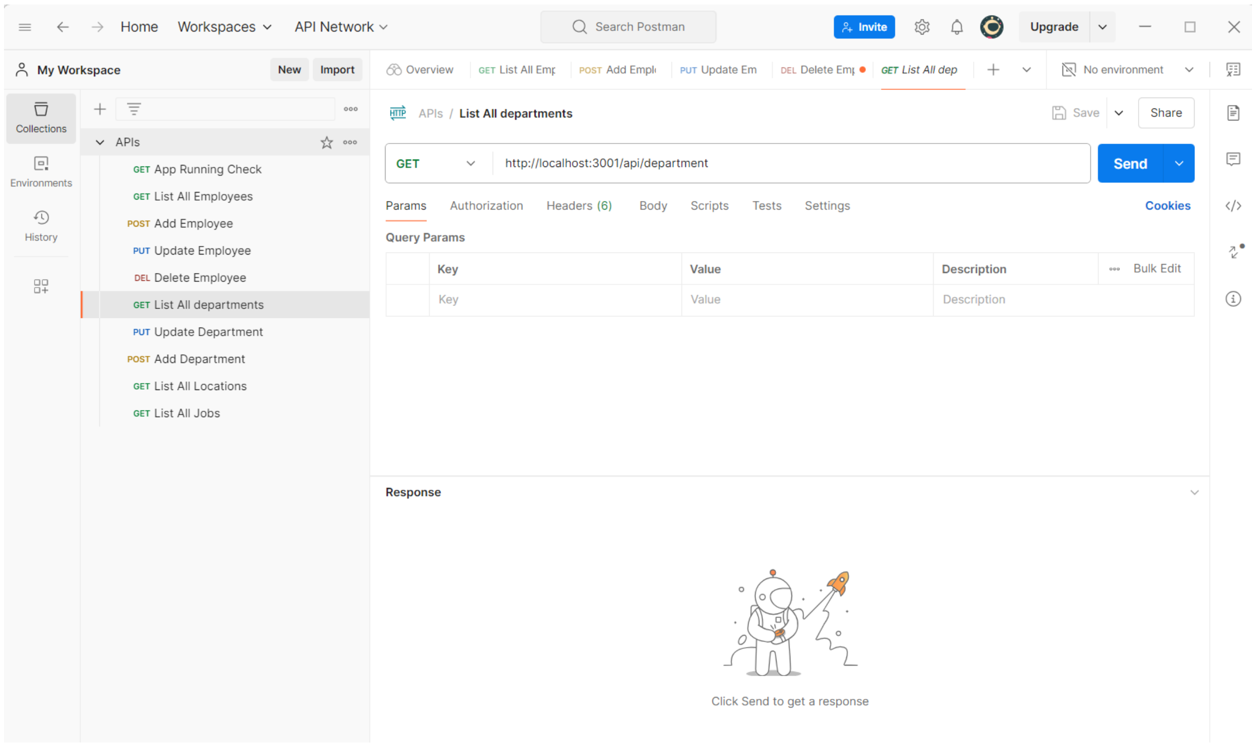

HR Management Software

Employees Departments

Department View Add Department

| Department ID | Department Name | Location ID | Actions |
|---|---|---|---|
| No Department | | | |

So we will open PostMan app and see if listAllDepartments() works there,



APIs
- GET App Running Check
- GET List All Employees
- POST Add Employee
- PUT Update Employee
- DEL Delete Employee
- GET List All departments
- PUT Update Department
- POST Add Department
- GET List All Locations
- GET List All Jobs

APIs / List All departments

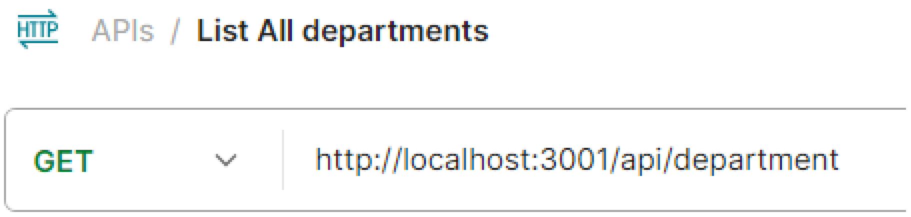
GET http://localhost:3001/api/department

Query Params

| Key | Value | Description |
|---|---|---|
| Key | Value | Description |

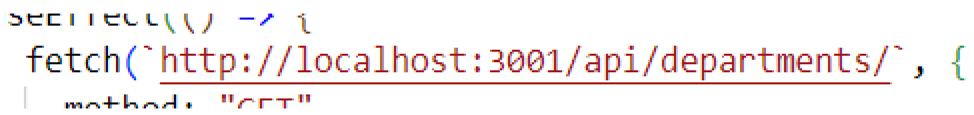
Response

And we found this error, that,

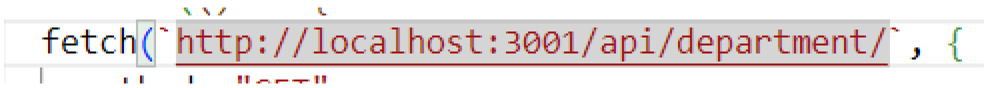

APIs / List All departments

GET http://localhost:3001/api/department

And ours is,


```
fetch(`http://localhost:3001/api/departments/`, {
  method: "GET"
```

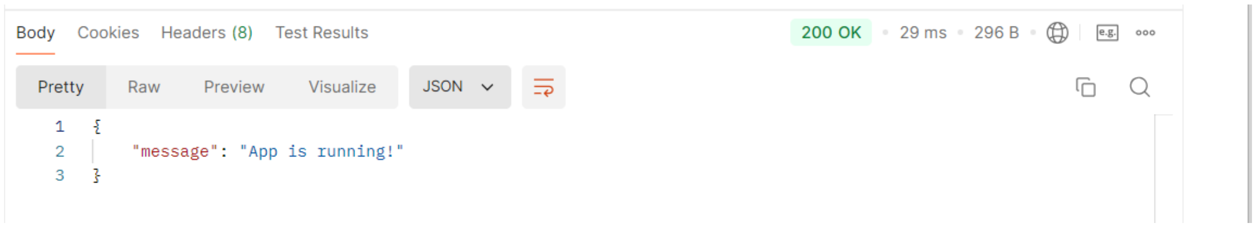
So we have to remove the 's' from departments,



```
fetch(`http://localhost:3001/api/department/`, {
```

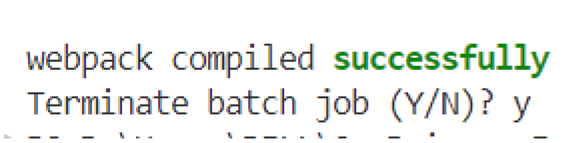
But still it did not work.

So now we will try and run postman,

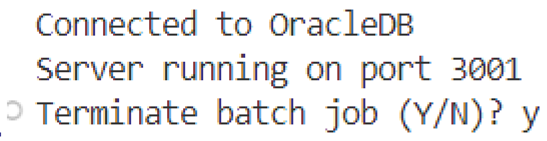


```
{
    "message": "App is running!"
}
```

So I think we should save all these files and then run. So lets press Ctrl+C, stop running backend and frontend,

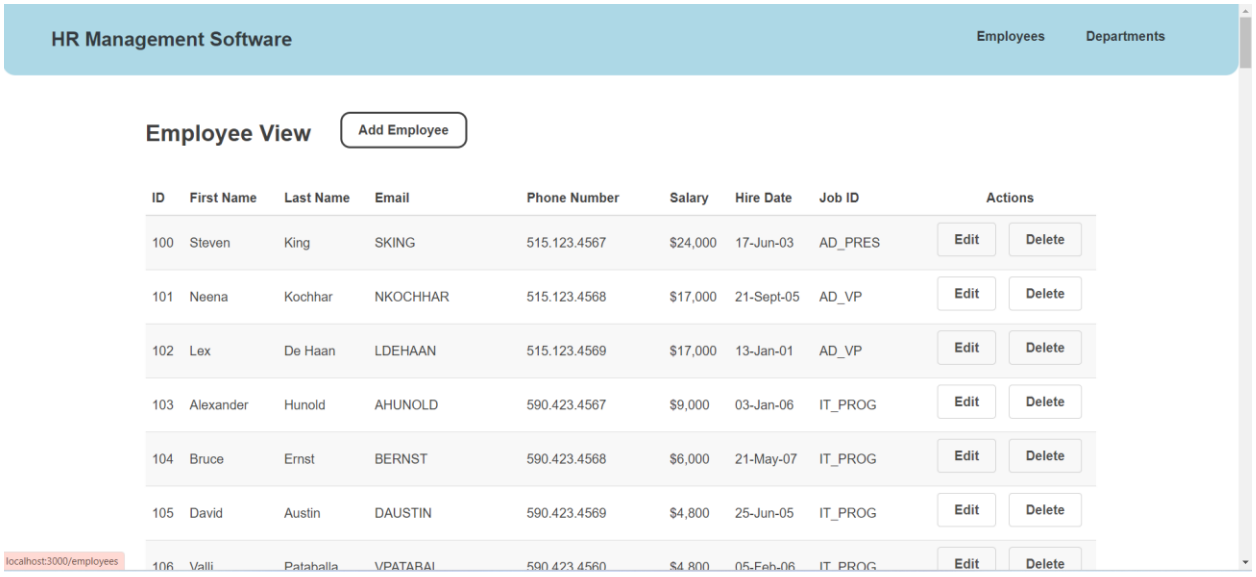


```
webpack compiled successfully
Terminate batch job (Y/N)? y
```



```
Connected to OracleDB
Server running on port 3001
Terminate batch job (Y/N)? y
```

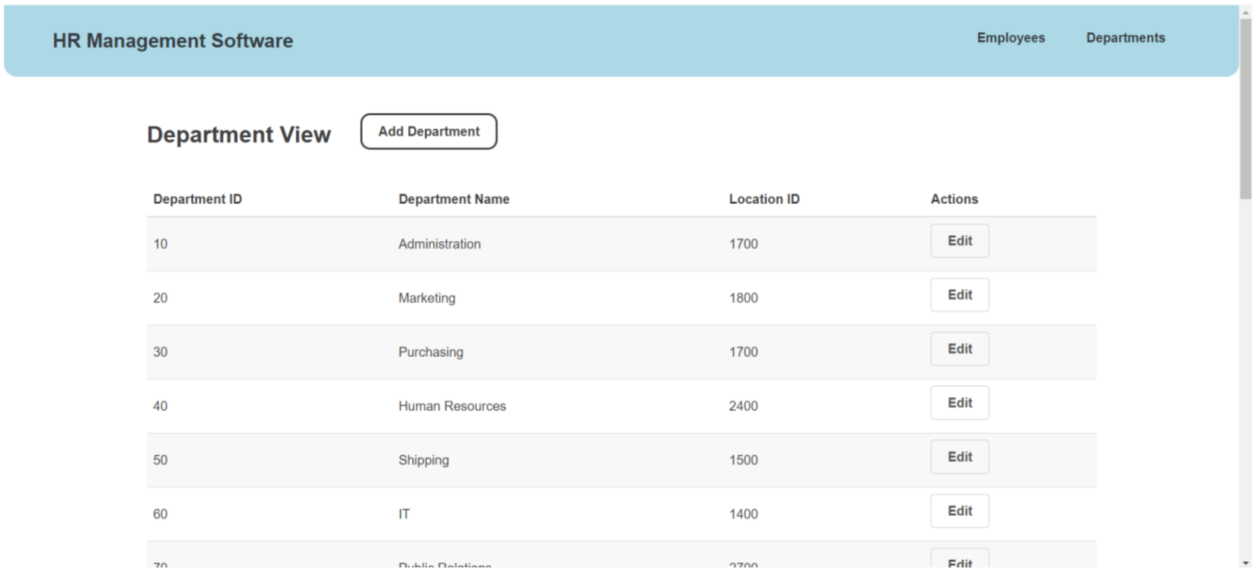
And then re-run by doing "npm run start" in backend folder and "npm start" in frontend folder and reload it,



Employee View

| ID | First Name | Last Name | Email | Phone Number | Salary | Hire Date | Job ID | Actions |
|---|---|---|---|---|---|---|---|---|
| 100 | Steven | King | SKING | 515.123.4567 | $24,000 | 17-Jun-03 | AD_PRES | Edit Delete |
| 101 | Neena | Kochhar | NKOCHHAR | 515.123.4568 | $17,000 | 21-Sept-05 | AD_VP | Edit Delete |
| 102 | Lex | De Haan | LDEHAAN | 515.123.4569 | $17,000 | 13-Jan-01 | AD_VP | Edit Delete |
| 103 | Alexander | Hunold | AHUNOLD | 590.423.4567 | $9,000 | 03-Jan-06 | IT_PROG | Edit Delete |
| 104 | Bruce | Ernst | BERNST | 590.423.4568 | $6,000 | 21-May-07 | IT_PROG | Edit Delete |
| 105 | David | Austin | DAUSTIN | 590.423.4569 | $4,800 | 25-Jun-05 | IT_PROG | Edit Delete |
| 106 | Valli | Pataballa | VPATABAL | 590.423.4560 | $4,800 | 05-Feb-06 | IT_PROG | Edit Delete |

And then we click departments,

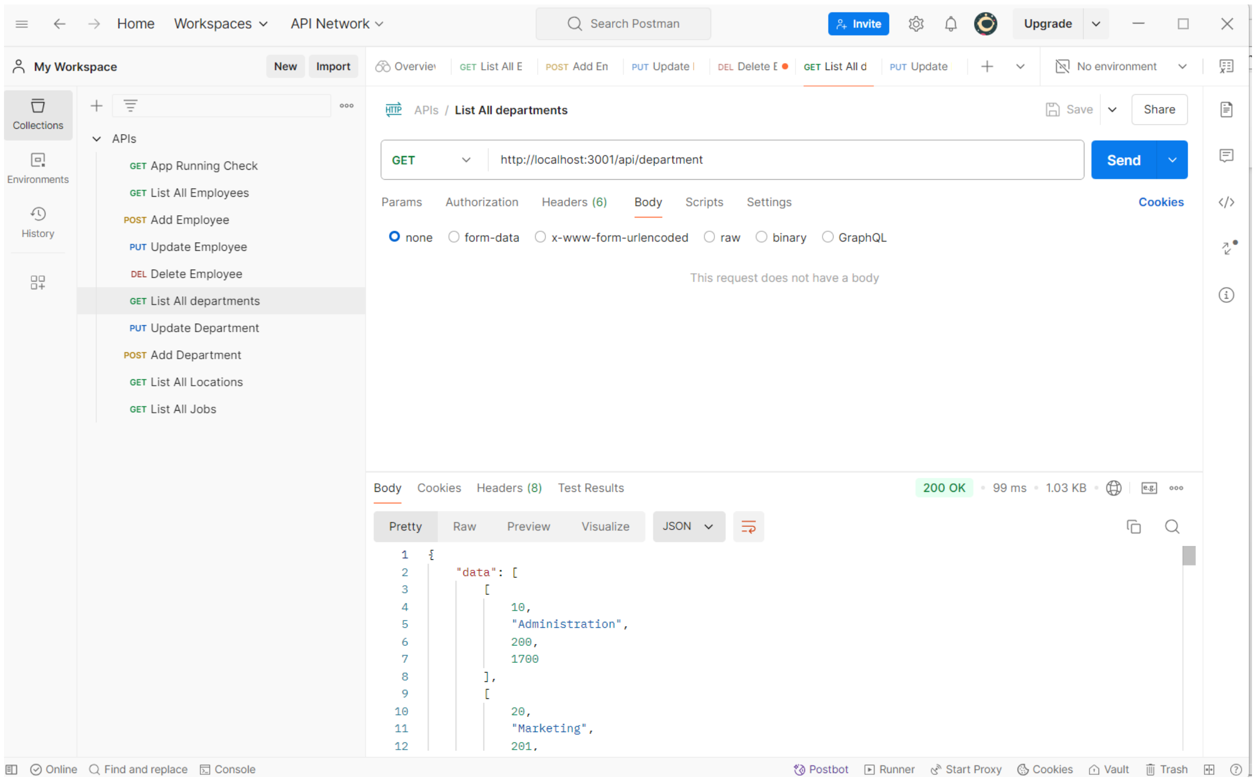

HR Management Software

Department View

| Department ID | Department Name | Location ID | Actions |
|---|---|---|---|
| 10 | Administration | 1700 | Edit |
| 20 | Marketing | 1800 | Edit |
| 30 | Purchasing | 1700 | Edit |
| 40 | Human Resources | 2400 | Edit |
| 50 | Shipping | 1500 | Edit |
| 60 | IT | 1400 | Edit |

And it worked! Alhamdullilah.

Lets run it in postman:



GET http://localhost:3001/api/department

Body: [✓ none] form-data x-www-form-urlencoded raw binary GraphQL

This request does not have a body

Body Cookies Headers (8) Test Results — 200 OK • 99 ms • 1.03 KB

```
{
    "data": [
        [
            10,
            "Administration",
            200,
            1700
        ],
        [
            20,
            "Marketing",
            201,
```

It worked!