CSSE3101 – ADVANCED WEB TECHNOLOGIES
MERN STACK DEVELOPMENT

Objective:

In this lab activity, you are going to Read data from the MongoDb database and display on the React application. You will read all the documents and specific documents based on some criteria.

PART 1- Express Route-Server-side script retrieve all students documents

Create the Express GET route for retrieving all the records from the database.

- `.find()` method – retrieves all the records
- `.countDocuments()` method – counts the number of records

```
//express GET route for retrieving all the records from the database
app.get("/getAllStudents", async (req, res) => {
        const students = await StudentModel.find()
        const count = await StudentModel.countDocuments({});
        res.send({ students, count});
});
```

PART 2- APPLICATION SETUP

1. In the client folder, update `ShowStudents.js` component by doing the following:
   a. Import the following:

```
import React, { useState, useEffect } from "react";
import Axios from "axios";
```

   b. Create state variables using `useState` hook.

```
const [listOfStudents, setlistOfStudents] = useState([]);
const [countRecords, setcountRecords] = useState(0);
```

   c. Create a `useEffect` hook to accept the response from the server.

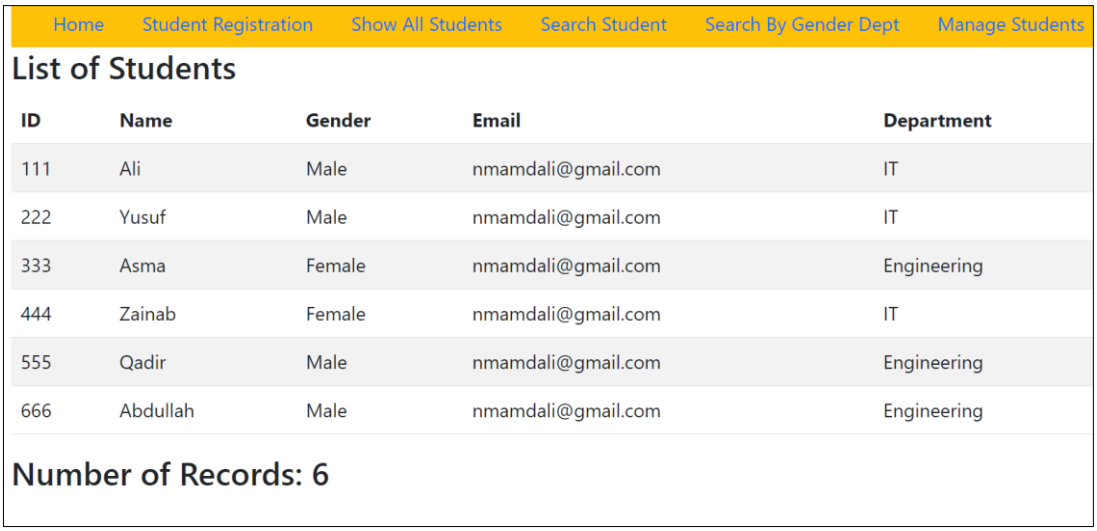
```
useEffect(() => {
     Axios.get("http://localhost:3001/getAllStudents")
        .then((response) => {
                setlistOfStudents(response.data.students);
                setcountRecords(response.data.count);
            })
        .catch((err) => {
                console.log(err);
        });
 }, []);
```

   d. In the `<tbody>` use the `map()` function to iterate over the response and display the data.

```
        listOfStudents.map((s) => {
                                return (
                                    <tr>
                                        <td>{s.studId}</td>
                                        <td>{s.studName}</td>
                                        <td>{s.gender}</td>
                                        <td>{s.email}</td>
                                        <td>{s.dept}</td>
                                    </tr>
                                )
                            }
                            )
```

e. Display the number of records.

```
<div>
  <h3>Number of Records: {countRecords}</h3>
 </div>
```



f. Update the **App.js** and add the following **link** and **routes**:

- `/home` – render the `Home.js` component
- `/stdreg` – render the `StudentRegister.js` component
- `/stdshow – render the ShowStudents.js component`

PART 3- Express Route– Writing Server-side script to retrieve a specific document

Update the index.js Write **"/getStudent"** Express route to retrieve a specific student document based on the student id from the MongoDb collection. Use the mongoose method to retrieve the documents based on the product brand chosen by the user.

```
//express GET route for a specific student document
app.get("/getStudent/:id", async (req, res) => {
    try {
        const id = req.params.id;
        const student = await StudentModel.find({studId: id});
        res.send({student});
        } catch (err) {
        console.error(err);
        }
});
```

PART 4- APPLICATION SETUP

In the folder client/components, create a new component named `GetStudentById.js.` The component Includes:

a) Input screen to get student id.

b) Create the state variables using the appropriate Reach Hook for the following:

```
const [studId,setStudId] = useState("")
const [student, setStudent] = useState([]);
```

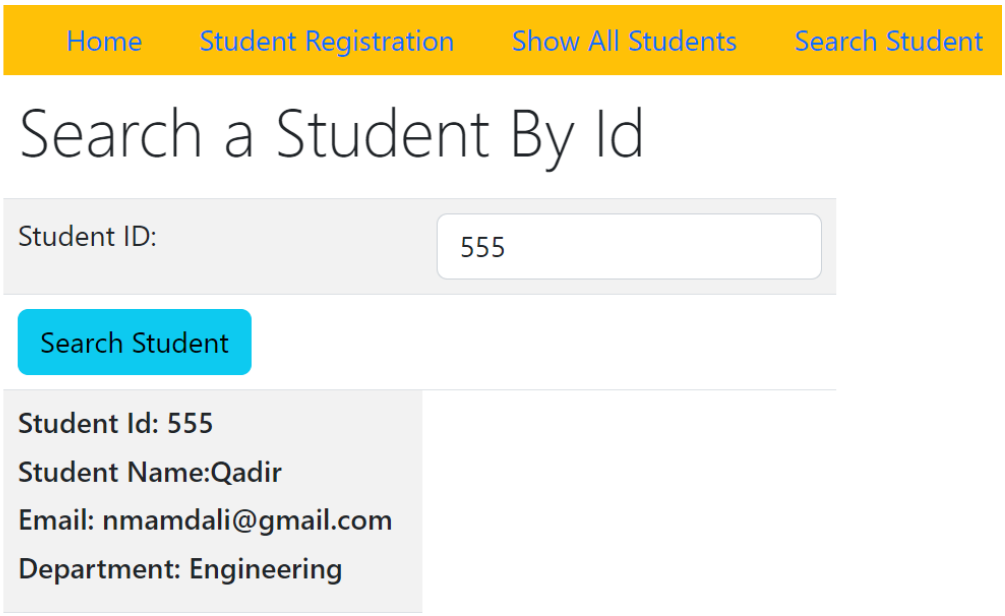
c) Add the corresponding event handlers to set the value of the respective state variable dynamically based on user input.

d) Write the code to make the function call to the `getStudent()` function when the user clicks the **"Search Student"** button. Send an **Axios get request** to the server to retrieve a student document of Student whose id is entered on input control. Service end point is "**/getStudent**".

```
const getStudent = async () => {
  Axios.get(`http://localhost:3001/getStudent/${studId}`)
      .then((response) => {
          console.log(response.data.student);
          setStudent(response.data.student);
        }
        );
 };
```

e) Update the **App.js** and add the **link** and **route**:

`/stdsearch` – render the `GetStudentById.js` component



Required Submission.

Once you complete the lab activity, you are required to upload the database model file `student.js`, `server` folder and `src` folder of the client app.