CRUD-for-React-and-MYSQL

Stage 1

This project is composed by two parts:The frontend and the backend.

Frontend

For the frontend, we will use react.js as our structure. Here is how to setup and initial a react project: If this this the first time for you to use react, you may also need to install node.js first.

```
sudo npm install create-react-app -g

create-react-app <projectName>

cd <projectName>

npm start
```

By doing this, the webpage will be open automatically. Also, you may need to

```
npm install axios --save
```

for this project.

In this demoproject, under the demoproject/src/, we will only need to keep the

- App.css
- App.js
- index.js

for our frontend development. To run locally, the default page for frontend is http://localhost:3001/

Backend

To start the backend, you can run

```
npm init
```

and press enter for all the questions

Then, you might need to install those for this demoproject:

```
npm install express body-parser mysql nodemon cors
```

After that, Create and Change index.js in backend as:

```
const express = require('express');
const app = express();


app.get('/', (require, response) => {
    response.send("Hello world");
})


app.listen(3002, () => {
    console.log("running on port 3002");
})
```

Run: `node index.js` for this program.

To see the updates, you might need to run `node index.js` after your changes. To simplify this process, we can make the following improvements:

In the package.json: make change:

```
"scripts": {
    "start": "node index.js",
    "devStart": "nodemon index.js",
    "test": "echo \"Error: no test specified\" && exit 1"
  },
```

Then run:

```
npm run devStart
```

By doing those, we can refresh the page to load changes.

To run locally, the default page for backend is http://localhost:3002/

Mysql

To to this, we use the MYSQLWorkbench to manage our dataset. You might need to install mysql and MYSQLWorkbench on your local mahcine.

Stage two:

For the details, please see the video. After you finish stage two, the code should looks like: frontend:

```
import './App.css';
import React, {useState, useEffect} from "react";
import Axios from 'axios';

function App() {
  return (
  <div className="App">
      <h1> CRUD APPLICATIONS</h1>

      <div className="form">
        <label> Movie Name:</label>
        <input type="text" name="movieName" />
        <label> Review:</label>
        <input type="text" name="Review" />

        <button> Submit</button>



      </div>

  </div>
  );
}

export default App;
```

backend:

```
const express = require("express");
const app = express();
const mysql = require("mysql");

var db = mysql.createConnection({
    host:'localhost',
    user: 'root',
    password:'mypassword',
    database:'411demo',
})

app.get('/', (require, response) => {
    const sqlInsert = "INSERT INTO `movie_reviews` (`movieName`,
`movieReview`) VALUES ('Spider2', 'good movie');";
    db.query(sqlInsert, (err, result) => {
        response.send("Hello world!!!");
```

```
    })
})

app.listen(3002, () => {
    console.log("running on port 3002");
})
```

Stage3:

For the details, please see the video. After you finish stage three, the code should looks like:

Changes can be made in App.css for the better format:

frontend:

```
import './App.css';
import React, {useState, useEffect} from "react";
import Axios from 'axios';

function App() {
  const [movieName, setMovieName] = useState('');
  const [Review, setReview] = useState('');

  const submitReview = () => {
    Axios.post('http://localhost:3002/api/insert', {
      movieName: movieName,
      movieReview: Review
    }).then(() => {
      alert('success insert')
    })

  };


  return (
    <div className="App">
      <h1> CRUD APPLICATIONS</h1>

      <div className="form">
        <label> Movie Name:</label>
        <input type="text" name="movieName" onChange={(e) => {
          setMovieName(e.target.value)
        } }/>
        <label> Review:</label>
        <input type="text" name="Review" onChange={(e) => {
          setReview(e.target.value)
        }}/>

        <button onClick={submitReview}> Submit</button>
```

```
      </div>

    </div>
  );
}

export default App;
```

backend:

```
const express = require("express");
const bodyParser = require("body-parser");
const app = express();
const mysql = require("mysql");
const cors = require("cors");


var db = mysql.createConnection({
    host:'localhost',
    user: 'root',
    password:'mypassword',
    database:'411demo',
})

app.use(cors());
app.use(bodyParser.urlencoded({ extended: true }));
app.use(express.json());

// app.get('/', (require, response) => {
//     const sqlInsert = "INSERT INTO `movie_reviews` (`movieName`,
`movieReview`) VALUES ('testMovie', 'cool movie!');";
//     db.query(sqlInsert, (err, result) => {
//         response.send("Hello world??");
//     })
// })

app.post("/api/insert", (require, response) => {
    const movieName = require.body.movieName;
    const movieReview = require.body.movieReview;

    const sqlInsert = "INSERT INTO `movie_reviews` (`movieName`,
`movieReview`) VALUES (?,?)";
    db.query(sqlInsert, [movieName, movieReview], (err, result) => {
        console.log(error);
    })
});

app.listen(3002, () => {
```

```
    console.log("running on port 3002");
})
```

stage4:

The final stage code can check the files.