**News and events Project**

What is MERN Stack?

MERN Stack stands for MongoDB/MySQL, Express, React, Node.js and this combined stack is known as MERN stack.

|  |  |
| --- | --- |
| **Stack** | **Detail** |
| **MongoDB/MySQL** | A document-oriented database programme based on NoSQL.  Structure oriented programming. |
| **Express** | It’s a robust Node.js web application framework that helps in creating powerful REST APIs. |
| **React** | A JavaScript library used to create beautiful and interactive user interfaces developed by Facebook and the community of individual developers. |
| **Node** | It’s a JavaScript runtime environment built on Google Chrome’s V8 engine, and it compiles js at the runtime. |

**Prerequisite**

HTML, CSS, JavaScript, Reactjs, nodejs, Expressjs, sql query

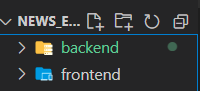
**Project steps:**

Whole project has 2 parts frontend and backend

**Step 1. Create project folder**

Create project folder using cmd command "mkdir news-and-events-my"

Name: news-and-events-my

Then two subfolders using cmd a) frontend b) backend

**Note: we add all the react code in frontend and express code in backend**

**SECTION-I**

**CREATING FRONTEND**

**Step 2: Create React Application**

To create react application go inside frontend folder using cd command

"cd frontend"

To create React application with use cmd command

"npx create-react-app ."

npm start

Note: This command opens the React project on the following URL: localhost: 3000

Now, you are all set to build the frontend of React news and event project

**Step 3: Integrating React Bootstrap with React App**

In the next step, install the React Bootstrap front-end framework in our MERN stack app. This framework will allow us to use the Bootstrap’s UI component in our React news and events app

React Bootstrap allows us to import individual UI components instead of importing the whole set of libraries.

npm install react-bootstrap

You have to import the Bootstrap into components and it will help you to create the UI components swiftly.

import "bootstrap/dist/css/bootstrap.css";



**Step 4: Creating router in app.js**

Create router using <Router> <routes> and <route path="" element="{}">

Four routes are created

1. root

2. dashboard

3. login

4. news form

<Router>

<Routes>

<Route path="/" element={<Login />}></Route>

<Route path="/login" element={<Login />}></Route>

<Route path="/form/:id" element={<Form />}></Route>

<Route path="/form" element={<Form />}></Route>

<Route path="/dashboard" element={<Dashboard />}></Route>

</Routes>

</Router>

**App.js**

import Login from "./pages/Authentication/Login";

import Form from "./pages/Dashboard/Form";

import Dashboard from "./pages/Dashboard/Dashboard";

import { BrowserRouter as Router, Routes, Route } from "react-router-dom";

export default function app() {

  return (

    <div>

      <Router>

        <Routes>

          <Route path="/" element={<Login />}></Route>

          <Route path="/login" element={<Login />}></Route>

          <Route path="/form/:id" element={<Form />}></Route>

          <Route path="/form" element={<Form />}></Route>

          <Route path="/dashboard" element={<Dashboard />}></Route>

        </Routes>

      </Router>

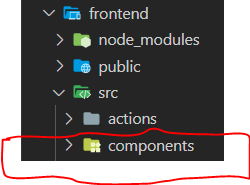
    </div>

  );

}

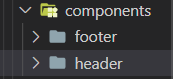
**Step 5: Creating components**

To create component we firstly create folder "component" in src

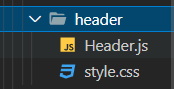


Next we create two more folder inside the component

1. Header and footer



1. Inside header folder create Header.js file and style.css



Header.js

import React from "react";

import "bootstrap/dist/css/bootstrap.min.css";

import "bootstrap/dist/js/bootstrap";

import { Link } from "react-router-dom";

export default function Header() {

  return (

    <nav className="navbar navbar-light bg-light">

      <div className="container-fluid">

        <img

          src="newslogo.png"

          alt=""

          width="100"

          height="70"

          className="d-inline-block align-text-top"

        />

        <Link to='/form'>Input</Link>

        <Link to='/dashboard'>Dashboard</Link>

        <span className="h12">News and Events</span>

      </div>

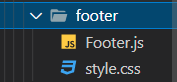
    </nav>

  );

}

Note : you can style accordingly in style.css

2. Inside footer folder create footer.js file and style.css



import React from "react";

import "bootstrap/dist/css/bootstrap.min.css";

import "bootstrap/dist/js/bootstrap";

import "./style.css";

export default function Footer() {

  return (

    <nav className="navbar navbar-light bg-light fnav">

      <div className="container-fluid tc">

        <div className="row">

          <div className="col">

            <span>Aptra Technologies pvt Ltd. </span>

          </div>

        </div>

      </div>

    </nav>

  );

}

Note : you can style accordingly in style.css

**Step 6: Creating Pages**

Create Pages folder to store our view, there are login form, dashboard, news entry form, the login form is created inside Authentication folder and rest are created inside dashboard folder.

Note: include header and footer component in these files.



1. Login form –it is the end point to the project: code snippet you can see below

return (

<>

<Header />

<div className="container-fluid log">

<div className="row">

<div className="col">

<div className="card text-dark bg-info mb-3">

<div className="card-body">

<center>

<span className="sp">Login in</span>

</center>

<form method="POST" onSubmit={handleSubmit}>

<div className="mb-3">

<label className="form-label">username</label>

<input

type="text"

className="form-control"

aria-describedby="emailHelp"

onChange={handleInputChange}

name="username"

value={inputs.username}

/>

</div>

<div className="mb-3">

<label className="form-label">Password</label>

<input

type="password"

className="form-control"

onChange={handleInputChange}

name="password"

value={inputs.password}

/>

</div>

<center>

<button type="submit" className="btn btn-primary">

Submit

</button>

</center>

<div style={{ marginLeft: 20 }}>

Not reistered?

<p className="but" onClick={handleRegister}>

click here!

</p>

</div>

</form>

</div>

</div>

</div>

</div>

</div>

<Footer />

</>

);

Whole Login file

import React, { useState } from "react";

import {CONFIG, LOGIN\_URL } from '../../utils/constants'

import "./style.css";

import axios from 'axios';

import Header from "../../components/header/Header";

import Footer from "../../components/footer/Footer";

import { useNavigate } from 'react-router-dom';

const Login = () => {

    const [inputs, setInputs] = useState({ username: "", password: "" });

    const navigate = useNavigate();

    const handleInputChange = (event) => {

        event.persist();

        setInputs((inputs) => ({

            ...inputs,

            [event.target.name]: event.target.value,

        }));

    };

    const handleRegister = () => {

        alert("clicked");

    };

    const handleSubmit = (event) => {

        if (event) {

            event.preventDefault();

            const body = JSON.stringify({

                username: inputs.username,

                password: inputs.password

            });

            try {

                axios.post(LOGIN\_URL, body, CONFIG )

                    .then((res) => {

                        console.log('client side')

                        console.log(res.data);

                        if (res.data.status === 'success') {

                            console.log('authenticate');

                            // return (<Dashboard/>

                            navigate('/form');

                        }

                        else {

                            console.log('failed');

                        }

                    })

            } catch (err) {

                console.log("error client side " + err);

            }

        }

    };

    return (

        <>

            <Header />

            <div className="container-fluid log">

                <div className="row">

                    <div className="col">

                        <div className="card text-dark bg-info mb-3">

                            <div className="card-body">

                                <center>

                                    <span className="sp">Login in</span>

                                </center>

                                <form method="POST" onSubmit={handleSubmit}>

                                    <div className="mb-3">

                                        <label className="form-label">username</label>

                                        <input

                                            type="text"

                                            className="form-control"

                                            aria-describedby="emailHelp"

                                            onChange={handleInputChange}

                                            name="username"

                                            value={inputs.username}

                                        />

                                    </div>

                                    <div className="mb-3">

                                        <label className="form-label">Password</label>

                                        <input

                                            type="password"

                                            className="form-control"

                                            onChange={handleInputChange}

                                            name="password"

                                            value={inputs.password}

                                        />

                                    </div>

                                    <center>

                                        <button type="submit" className="btn btn-primary">

                                            Submit

                                        </button>

                                    </center>

                                    <div style={{ marginLeft: 20 }}>

                                        Not reistered?

                                        <p className="but" onClick={handleRegister}>

                                            click here!

                                        </p>

                                    </div>

                                </form>

                            </div>

                        </div>

                    </div>

                </div>

            </div>

            <Footer />

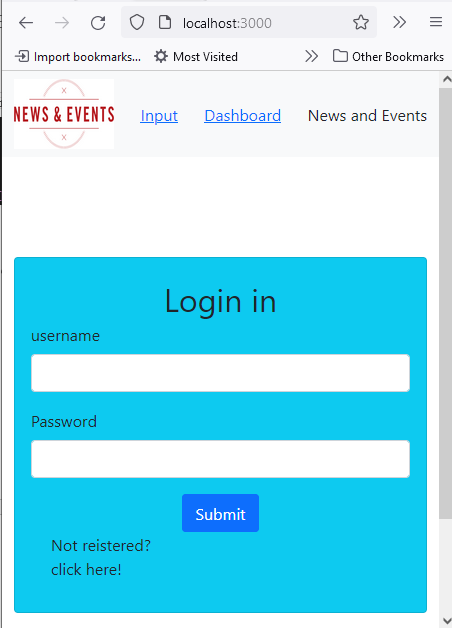
        </>

    );

};

export default Login;

The UI output of form is shown below:



2. Dash board.js

import React, { useEffect, useState } from "react";

import {

  Table,

  Button,

  Col,

  Container,

  Form,

  FormControl,

  InputGroup,

  Row,

} from "react-bootstrap";

import { useNavigate } from "react-router-dom";

import { delete\_from, getNews, update } from "../../actions/newsAction";

import Footer from "../../components/footer/Footer";

import Header from "../../components/header/Header";

const Dashboard = () => {

  const [newsData, setNewsData] = useState([]);

  let result;

  let sn = 1;

  const navigate = useNavigate();

  useEffect(() => {

    result = getNews();

    result.then((res) => {

      setNewsData(res);

      // console.log(res);

    });

  }, []);

  console.log(newsData);

  const handleEdit = (id) => {

    navigate(`/form/${id}`);

  };

  const handleDelete = (id) => {

    const res = window.confirm("delete row ?");

    if (res) {

      delete\_from({ id: id });

      window.location.reload();

    }

  };

  const handleView = (id) => {

    navigate(`/form/${id}`);

  };

  return (

    <>

      <Header />

      <Table striped hover bordered size="sm">

        <thead>

          <td>sl. no</td>

          <td>Title</td>

          <td>Type</td>

          <td>News</td>

          <td>Action</td>

        </thead>

        <tbody>

          {newsData?.map((obj) => {

            return (

              <>

                <tr>

                  <td>{sn++}</td>

                  <td>{obj.title}</td>

                  <td>{obj.type}</td>

                  <td>{obj.news}</td>

                  <td>

                    <Button variant="link" onClick={() => handleView(obj.id)}>

                      View

                    </Button>

                    <Button variant="link" onClick={() => handleEdit(obj.id)}>

                      Edit

                    </Button>

                    <Button variant="link" onClick={() => handleDelete(obj.id)}>

                      Delete

                    </Button>

                  </td>

                </tr>

              </>

            );

            sn++;

          })}

        </tbody>

      </Table>

      <Footer />

    </>

  );

};

export default Dashboard;

dashboard snippet:

<>

<Header />

<Table striped hover bordered size="sm">

<thead>

<td>sl. no</td>

<td>Title</td>

<td>Type</td>

<td>News</td>

<td>Action</td>

</thead>

<tbody>

{newsData?.map((obj) => {

return (

<>

<tr>

<td>{sn++}</td>

<td>{obj.title}</td>

<td>{obj.type}</td>

<td>{obj.news}</td>

<td>

<Button variant="link" onClick={() => handleView(obj.id)}>

View

</Button>

<Button variant="link" onClick={() => handleEdit(obj.id)}>

Edit

</Button>

<Button variant="link" onClick={() => handleDelete(obj.id)}>

Delete

</Button>

</td>

</tr>

</>

);

sn++;

})}

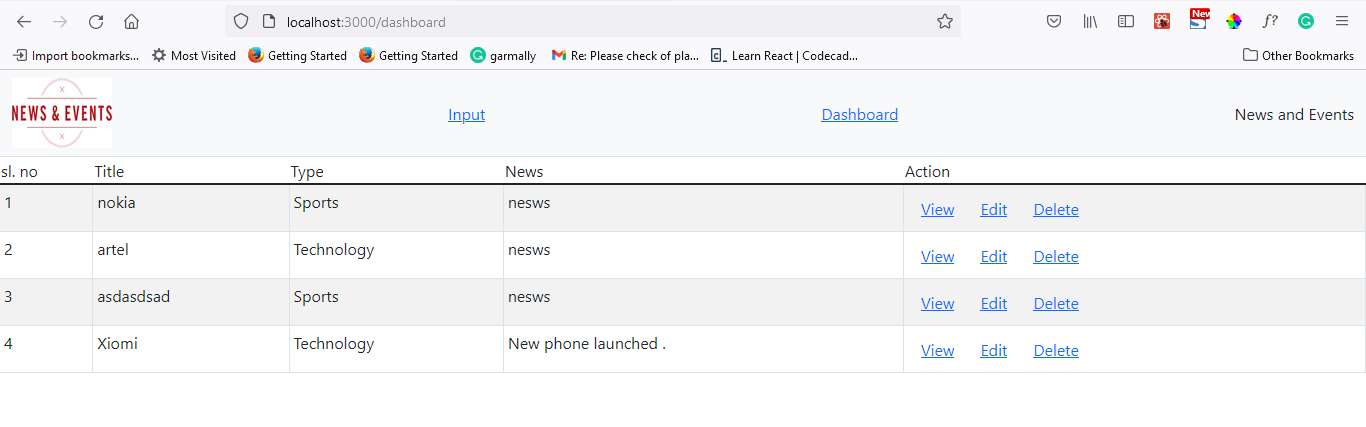
</tbody>

</Table>

<Footer />

</>

The UI output of dashboard is shown below:



3. news form is created in form.js

import React, { useEffect, useState } from "react";

import {

  Button,

  Col,

  Container,

  Form,

  FormControl,

  InputGroup,

  Row,

} from "react-bootstrap";

import Header from "../../components/header/Header";

import Footer from "../../components/footer/Footer";

import { NEWS\_URL, CONFIG } from "../../utils/constants";

import { useNavigate, useParams } from "react-router-dom";

import axios from "axios";

import { getRow, update } from "../../actions/newsAction";

const Dashboard = () => {

  const { id } = useParams();

  const [inputs, setInputs] = useState({

    title: "",

    type: "select",

    news: "",

  });

  const navigate = useNavigate();

  const handleInputChange = (event) => {

    event.persist();

    setInputs((inputs) => ({

      ...inputs,

      [event.target.name]: event.target.value,

    }));

  };

  useEffect(() => {

    // window.location.reload();

    if (id) {

      var res = getRow({ id: id });

      console.log(res);

      res.then((value) => {

        console.log(value[0]);

        setInputs(value[0]);

      });

    }

  }, []);

  const handleSubmit = (event) => {

    console.log(inputs);

    if (event) {

      event.preventDefault();

      const obj = {

        id: id,

        title: inputs.title,

        type: inputs.type,

        news: inputs.news,

      };

      if (!id)

        try {

          const body = JSON.stringify(obj);

          axios.post(NEWS\_URL, body, CONFIG).then((res) => {

            console.log("client side news");

            console.log(res.data);

            if (res.data.status === "success") {

              console.log("submitted successful");

              alert("insert successful ");

            } else {

              console.log("failed");

              alert("failed");

            }

          });

        } catch (err) {

          console.log("error client side " + err);

        }

      if (id) {

        update(obj);

        navigate(`/dashboard`);

      }

    }

  };

  return (

    <>

      {/\* <GoBackNavBar title={'New Inward Record'} /> \*/}

      <Header />

      <Form onSubmit={(e) => handleSubmit(e)}>

        <Container>

          <Row className="pt-5">

            {/\* -------- Title  -------- \*/}

            <InputGroup className="mb-3 mt-4">

              <InputGroup.Text>Title : </InputGroup.Text>

              <FormControl

                type="text"

                placeholder="title"

                name="title"

                value={inputs.title}

                onChange={(value) => handleInputChange(value)}

              />

            </InputGroup>

          </Row>

          <Row>

            {/\* -------- Type -------- \*/}

            <InputGroup className="mb-3 mt-4">

              <InputGroup.Text>News Type : </InputGroup.Text>

              {

                <select

                  name="type"

                  className="dropdown-border"

                  id="dropdown"

                  onChange={(val) => {

                    handleInputChange(val);

                  }}

                >

                  <option value="">{inputs.type}</option>

                  <option vlaue="">Sports</option>

                  <option vlaue="">News</option>

                  <option vlaue="">Technology</option>

                </select>

              }

            </InputGroup>

          </Row>

          <Row>

            <Col xs={12} sm={12} md={12} lg={4}>

              {/\* -------- News  -------- \*/}

              <InputGroup className="mb-3 mt-4">

                <InputGroup.Text>News : </InputGroup.Text>

                {/\* <FormControl

                                    type='textarea'

                                    placeholder=""

                                    name='news'

                                    // value={formData.nature}

                                    onChange={(value) => handleInputChange(value)}

                                /> \*/}

                <textarea

                  name="news"

                  value={inputs.news}

                  onChange={(value) => handleInputChange(value)}

                ></textarea>

              </InputGroup>

            </Col>

          </Row>

          <Row>

            {/\* -------- Form control Buttons -------- \*/}

            <Col

              lg={{ span: 3, offset: 4 }}

              md={{ span: 4, offset: 4 }}

              sm={{ span: 5, offset: 4 }}

            >

              {!id && (

                <Button className="m-4" type="submit" variant="success">

                  Save

                </Button>

              )}

              {id && (

                <Button className="m-4" type="submit" variant="success">

                  Update

                </Button>

              )}

            </Col>

          </Row>

        </Container>

      </Form>

      <Footer />

    </>

  );

};

export default Dashboard;

code snippet

<>

<Header />

<Form onSubmit={(e) => handleSubmit(e)}>

<Container>

<Row className="pt-5">

{/\* -------- Title -------- \*/}

<InputGroup className="mb-3 mt-4">

<InputGroup.Text>Title : </InputGroup.Text>

<FormControl

type="text"

placeholder="title"

name="title"

value={inputs.title}

onChange={(value) => handleInputChange(value)}

/>

</InputGroup>

</Row>

<Row>

{/\* -------- Type -------- \*/}

<InputGroup className="mb-3 mt-4">

<InputGroup.Text>News Type : </InputGroup.Text>

{

<select

name="type"

className="dropdown-border"

id="dropdown"

onChange={(val) => {

handleInputChange(val);

}}

>

<option value="">{inputs.type}</option>

<option vlaue="">Sports</option>

<option vlaue="">News</option>

<option vlaue="">Technology</option>

</select>

}

</InputGroup>

</Row>

<Row>

<Col xs={12} sm={12} md={12} lg={4}>

{/\* -------- News -------- \*/}

<InputGroup className="mb-3 mt-4">

<InputGroup.Text>News : </InputGroup.Text>

{/\* <FormControl

type='textarea'

placeholder=""

name='news'

// value={formData.nature}

onChange={(value) => handleInputChange(value)}

/> \*/}

<textarea

name="news"

value={inputs.news}

onChange={(value) => handleInputChange(value)}

></textarea>

</InputGroup>

</Col>

</Row>

<Row>

{/\* -------- Form control Buttons -------- \*/}

<Col

lg={{ span: 3, offset: 4 }}

md={{ span: 4, offset: 4 }}

sm={{ span: 5, offset: 4 }}

>

{!id && (

<Button className="m-4" type="submit" variant="success">

Save

</Button>

)}

{id && (

<Button className="m-4" type="submit" variant="success">

Update

</Button>

)}

</Col>

</Row>

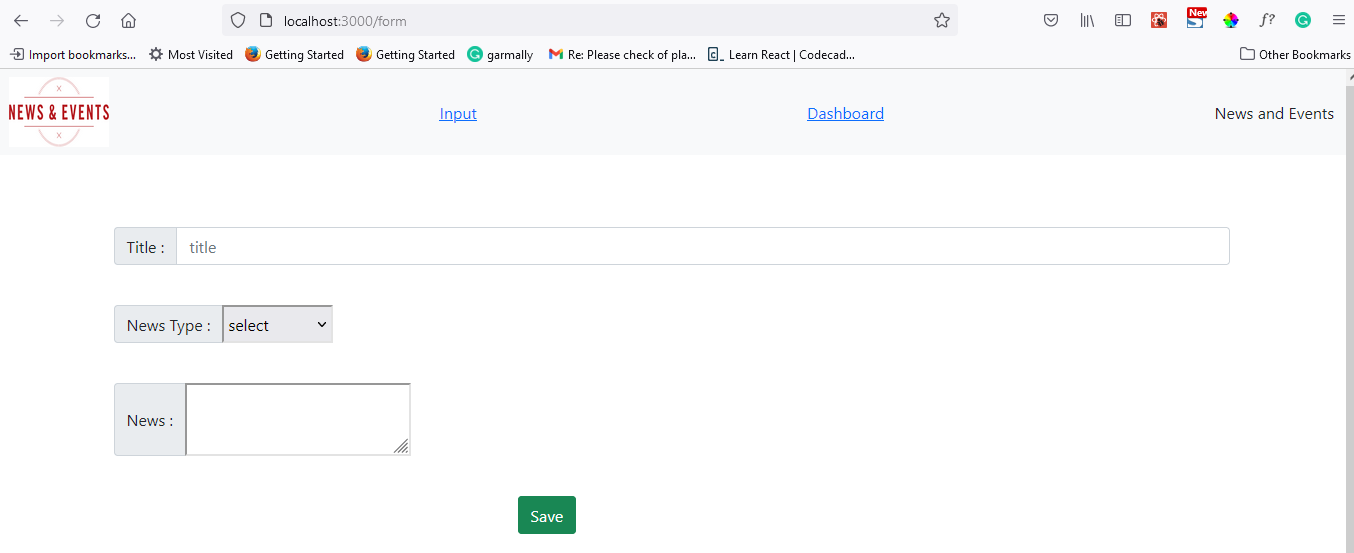
</Container>

</Form>

<Footer />

</>

The UI of news form is shown below:



**SECTION-II**

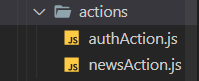
**Transferring data from frontend to backend**

**Step 7: Creating data transfer actions**

To create actions firstly create separate folder "actions" inside src



1. To transfer login form data and news form data to backend we create authaction.js and newsAction.js file inside actions folder



2. code snippet of authAction.js

await axios.post(NEWS\_SELECT\_URL, body, CONFIG)

.then((res) => {

console.log('client side')

console.log(res.data);

if(res.data.status === 'success'){

console.log('authenticate');

}

else {

console.log('failed');

}

})

import axios from 'axios';

import { CONFIG, NEWS\_SELECT\_URL } from '../components/utils/constants';

export const loginAction = async ({ username, password }) => {

    const body = JSON.stringify({

        username: username,

        password :password

    });

    try {

        await axios.post(NEWS\_SELECT\_URL, body, CONFIG)

        .then((res) => {

            console.log('client side')

            console.log(res.data);

            if(res.data.status === 'success'){

                console.log('authenticate');

            }

            else {

                console.log('failed');

            }

        })

    } catch (err) {

        console.log("error client side " + err);

    }

}

3. Code snippet of newsActions.js for sending data adding, updating and deleting data is shown below

Sending---------------------

const row = axios.post('http://localhost:5000/select/row', body, CONFIG)

.then((res) => {

// console.log(res.data.selectRow);

console.log('single row retrived successful')

// sessionStorage.setItem('selectRow', JSON.stringify({rowData : res.data.selectRow }));

// Next()

return res.data.selectRow;

})

Deleting-------------------

await axios.post(NEWS\_DELETE\_URL, body, CONFIG)

.then((res) => {

console.log(res.data);

if (res.data.status === 'success') {

}

})

Updating-----------------------

await axios.post(NEWS\_EDIT\_URL, body, CONFIG)

.then((res) => {

console.log(res.data);

if (res.data.status === SUCCESS);

})

Adding-------------------

const newsTable = axios.post(NEWS\_SELECT\_URL, CONFIG)

.then((res) => {

// console.log('client side news action')

// console.log(res.data.row);

return res.data.row;

})

return newsTable;

import axios from 'axios';

import { NEWS\_SELECT\_URL, CONFIG, SUCCESS, NEWS\_EDIT\_URL, NEWS\_DELETE\_URL } from '../utils/constants';

export const getNews = () => {

    try {

        const newsTable = axios.post(NEWS\_SELECT\_URL, CONFIG)

            .then((res) => {

                // console.log('client side news action')

                // console.log(res.data.row);

                return res.data.row;

            })

        return newsTable;

    } catch (err) {

        console.log("error client side " + err);

    }

}

// ----------------- Update Row handler -----------------

export const update = async ({ id, title, type, news }) => {

    console.log(' Edit news ');

    const body = JSON.stringify({

        id: id,

        title: title,

        type: type,

        news: news

    });

    console.log(body);

    try {

        await axios.post(NEWS\_EDIT\_URL, body, CONFIG)

            .then((res) => {

                console.log(res.data);

                if (res.data.status === SUCCESS);

            })

    }

    catch (err) {

        console.log(err);

    }

}

// ----------------- Retrive specific row handler -----------------

export const getRow = ({ id }) => {

    const body = JSON.stringify({

        id: id

    })

    try {

        const row = axios.post('http://localhost:5000/select/row', body, CONFIG)

            .then((res) => {

                // console.log(res.data.selectRow);

                console.log('single row retrived successful')

                // sessionStorage.setItem('selectRow', JSON.stringify({rowData : res.data.selectRow }));

                // Next()

                return res.data.selectRow;

            })

        // .then((res) => {

        // console.log(res.data);

        // store.dispatch(setRowData());

        // return res.data.selectRow;

        // } );

        return row;

    }

    catch (err) {

        console.log(err);

        return 'error';

    }

}

// ----------------- delete handler -----------------

export const delete\_from = async ({ id }) => {

    const body = JSON.stringify({ id: id });

    try {

        await axios.post(NEWS\_DELETE\_URL, body, CONFIG)

            .then((res) => {

                console.log(res.data);

                if (res.data.status === 'success') {

                }

            })

        // delete successful alert msg.

    }

    catch (err) {

        console.log(err);

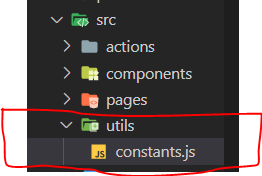
    }

}

**SECTION-III**

**Necessary Constants**

To set and use necessary constants from one place create a folder "utils" inside src and create a file constans.js inside it



1. Define the constants inside constants.js

// ------------------------------ URLs ------------------------------

const BASE\_URL = "http://localhost:5000"

export const LOGIN\_URL = BASE\_URL+ "/login";

export const NEWS\_URL = BASE\_URL+ "/insert/news";

export const NEWS\_SELECT\_URL = BASE\_URL+ "/select/news";

export const NEWS\_EDIT\_URL = BASE\_URL+ "/edit/news";

export const NEWS\_DELETE\_URL = BASE\_URL+ "/delete/news";

export const SUCCESS = 'success';

// ------------------------------ Configuration data ------------------------------

export const CONFIG = {

    headers: {

        "Access-Control-Allow-Headers": "\*",

        "Access-Control-Allow-Methods": 'OPTIONS,POST,GET',

        'Content-Type': 'application/json'

      }

}

**SECTION-IV**

**Backend Processing**

**Step 8: Create a file dbserver.js inside frontend folder**

1. code snippet for login processing

/ ---------------------- Login ----------------------

app.use('/login', (req, res) => {

const sql = `SELECT password FROM user WHERE username=?`;

const username = req.body.username.trim();

const password = req.body.password.trim();

console.log("username: ", username, password);

try {

db.query(sql, [username], (err, rows, fields) => {

try {

console.log(rows[0].password);

if (rows[0].password === password) {

res.send({ status: 'success' });

}

else res.send({ status: 'failed' });

}

catch (err) {

console.error(err);

}

})

}

catch (err) {

console.error(err);

}

})

2. code snippet for inserting news form data

// ---------------------- (insert query) news ----------------------

app.post('/insert/news', (req, res) => {

const NEWS\_INSERT = `insert into news\_tb (title, type, news) values(?, ?, ?) `;

const title = req.body.title.trim();

const type = req.body.type.trim();

const news = req.body.news.trim();

console.log(title, type, news);

try {

db.query(NEWS\_INSERT, [title, type, news], (err, rows, fields) => {

try {

// console.log(rows);

if (err) res.send({ status: 'error' });

else res.send({ status: 'success' });

}

catch (err) {

console.error(err);

}

})

}

catch (err) {

console.error(err);

}

})

// ---------------------- (select query) news ----------------------

app.post('/select/news', (req, res) => {

const NEWS\_SELECT = `select \* from news\_tb ;`;

try {

db.query(NEWS\_SELECT, (err, rows, fields) => {

try {

// console.log(rows);

if (err) res.send({ row: ['error'] });

else res.send({ row: rows });

}

catch (err) {

console.error(err);

}

})

}

catch (err) {

console.error(err);

}

})

3. code snippet for read data from db

/ ---------------------- (select specific query) news ----------------------

app.post('/select/row', (req, res) => {

const SELECT\_NEWS\_ROW = `select \* from news\_tb where id = ? ;`;

const id = req.body.id;

console.log(id);

db.query(SELECT\_NEWS\_ROW, [id], (err, rows, fields) => {

if (err) res.send({ selectRow: 'error' });

else {

res.send({ selectRow: rows })

}

})

})

4. code to update data in db

app.post('/edit/news', (req, res) => {

const NEWS\_UPDATE = `update news\_tb set title=?, type=?, news=? where id=? ;`;

const title = req.body.title;

const type = req.body.type;

const news = req.body.news;

const id = req.body.id;

try {

db.query(NEWS\_UPDATE, [title, type, news, id], (err, rows, fields) => {

try {

// console.log(rows);

if (err) res.send({ status: 'error' });

else res.send({ status: 'success' });

}

catch (err) {

console.error(err);

}

})

}

catch (err) {

console.error(err);

}

})

5. code snippet to delete data in database

app.post('/delete/news', (req, res) => {

console.log('delete news');

const DELETE\_NEWS = `delete from news\_tb where id = ? ;`;

const id = req.body.id;

console.log(id);

try {

db.query(DELETE\_NEWS, [id], (err, rows, fields) => {

if (err) {

res.send({ status: 'error' });

console.log(err);

}

else res.send({ status: 'success' });

})

}

catch (err) {

console.log(err);

}

Over all code in dbserver is shown in server

import dotenv from 'dotenv';

dotenv.config();

import express from 'express';

const app = express()

import mysql from 'mysql';

import cors from 'cors';

const DB\_HOST = process.env.DB\_HOST

const DB\_USER = process.env.DB\_USER

const DB\_PASSWORD = process.env.DB\_PASSWORD

const DB\_DATABASE = process.env.DB\_DATABASE

const DB\_PORT = process.env.DB\_PORT

const port = process.env.PORT

const CREDENTIALS = {

    connectionLimit: 100,

    host: DB\_HOST,

    user: DB\_USER,

    password: DB\_PASSWORD,

    database: DB\_DATABASE,

    port: DB\_PORT

}

const corsOptions = {

    origin: 'http://localhost:3000',

    credentials: true,

    optionSuccessStatus: 200

}

const db = mysql.createPool(CREDENTIALS);

app.use(cors(corsOptions));

app.use(express.json());

// ---------------------- Login ----------------------

app.use('/login', (req, res) => {

    const sql = `SELECT password FROM user WHERE username=?`;

    const username = req.body.username.trim();

    const password = req.body.password.trim();

    console.log("username: ", username, password);

    try {

        db.query(sql, [username], (err, rows, fields) => {

            try {

                console.log(rows[0].password);

                if (rows[0].password === password) {

                    res.send({ status: 'success' });

                }

                else res.send({ status: 'failed' });

            }

            catch (err) {

                console.error(err);

            }

        })

    }

    catch (err) {

        console.error(err);

    }

})

// ---------------------- (insert query) news ----------------------

app.post('/insert/news', (req, res) => {

    const NEWS\_INSERT = `insert into news\_tb (title, type, news) values(?, ?, ?) `;

    const title = req.body.title.trim();

    const type = req.body.type.trim();

    const news = req.body.news.trim();

    console.log(title, type, news);

    try {

        db.query(NEWS\_INSERT, [title, type, news], (err, rows, fields) => {

            try {

                // console.log(rows);

                if (err) res.send({ status: 'error' });

                else res.send({ status: 'success' });

            }

            catch (err) {

                console.error(err);

            }

        })

    }

    catch (err) {

        console.error(err);

    }

})

// ---------------------- (select query) news ----------------------

app.post('/select/news', (req, res) => {

    const NEWS\_SELECT = `select \* from  news\_tb  ;`;

    try {

        db.query(NEWS\_SELECT, (err, rows, fields) => {

            try {

                // console.log(rows);

                if (err) res.send({ row: ['error'] });

                else res.send({ row: rows });

            }

            catch (err) {

                console.error(err);

            }

        })

    }

    catch (err) {

        console.error(err);

    }

})

// ---------------------- (select specific query) news ----------------------

app.post('/select/row', (req, res) => {

    const SELECT\_NEWS\_ROW = `select \* from news\_tb where id = ? ;`;

    const id = req.body.id;

    console.log(id);

    db.query(SELECT\_NEWS\_ROW, [id], (err, rows, fields) => {

        if (err) res.send({ selectRow: 'error' });

        else {

            res.send({ selectRow: rows })

        }

    })

})

// ---------------------- (update query) news ----------------------

app.post('/edit/news', (req, res) => {

    const NEWS\_UPDATE = `update news\_tb set title=?, type=?, news=? where id=? ;`;

    const title = req.body.title;

    const type = req.body.type;

    const news = req.body.news;

    const id = req.body.id;

    try {

        db.query(NEWS\_UPDATE, [title, type, news, id], (err, rows, fields) => {

            try {

                // console.log(rows);

                if (err) res.send({ status: 'error' });

                else res.send({ status: 'success' });

            }

            catch (err) {

                console.error(err);

            }

        })

    }

    catch (err) {

        console.error(err);

    }

})

app.post('/delete/news', (req, res) => {

    console.log('delete news');

    const DELETE\_NEWS = `delete from news\_tb where id = ? ;`;

    const id = req.body.id;

    console.log(id);

    try {

        db.query(DELETE\_NEWS, [id], (err, rows, fields) => {

            if (err) {

                res.send({ status: 'error' });

                console.log(err);

            }

            else res.send({ status: 'success' });

        })

    }

    catch (err) {

        console.log(err);

    }

});

app.listen(port, () => console.log('server started at port ' + port))

NOTE: .env file creating inside frontend folder and setting database parameters

DB\_HOST = localhost

DB\_USER = root

DB\_PASSWORD = ''

DB\_DATABASE = news

DB\_PORT = 3306

PORT = 5000

NOTE: necessary library packages used the project backend

"author": "Srinath",

"license": "ISC",

"dependencies": {

"async": "^3.2.3",

"cors": "^2.8.5",

"dotenv": "^16.0.0",

"express": "^4.17.3",

"mysql": "^2.18.1",

"pg-promise": "^10.11.1"

}

NOTE: necessary library packages for frontend

{

"name": "frontend",

"version": "0.1.0",

"private": true,

"dependencies": {

"@popperjs/core": "^2.11.2",

"@testing-library/jest-dom": "^5.16.2",

"@testing-library/react": "^12.1.3",

"@testing-library/user-event": "^13.5.0",

"axios": "^0.27.2",

"bootstrap": "^5.1.3",

"react": "^17.0.2",

"react-bootstrap": "^2.4.0",

"react-dom": "^17.0.2",

"react-popper": "^2.2.5",

"react-router-dom": "^6.3.0",

"react-scripts": "5.0.0",

"web-vitals": "^2.1.4"

},

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*END\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

You can find this code at github link