**SOURCE CODE FOR THE CSBC1010-NOTES-APP-SERVER**

1. **Setting up a successful connection**

var mysql = require('mysql');

var connection = mysql.createConnection({

host : 'localhost',

user : 'root',

password : ''

});

connection.connect(function(err) {

if (err) {

console.error('error connecting: ' + err.stack);

return;

}

console.log('connected as id ' + connection.threadId);

});

1. **Fetching all notes from the database**

query = [{SELECT id, text, dateCreated, lastModified FROM notes};]

connection.query(query, function (error, results, fields) {

if (error) {

console.error (query, error);

return next(error);

}

res.send(results);

});

1. **Create a new note**

module.exports = (req, res, next) =>{

const token = req.headers.authorization.split(' ')[1];

const decoded = jwt.verify(token, process.env.SECRET);

const userId= decoded.user.\_id

try{

if(req.body.userId && req.body.userId !== userId){

throw 'invalid user ID';}}else{

req.jwtPayload = decoded // add payload here to req

next();

}

}catch{

res.status(401).json({ error : 'not authenticated !'});

}

exports.createNote = (req, res, next) =>{

const { jwtPayload } = req

// do stuff with jwtPayload

let note = new Note(req.body);

note.user = req.user;note.save((err) =>{

SELECT id, text, dateCreated, lastModified FROM notes

if(err){

res.status(401);

return next(err)

}

return res.status(201).json('note created successfully!')

})

1. [**Update an existing note**](https://github.com/vivienfan/csbc1010-notes-app-server/blob/master/src/routes/note.js#L62-L119)

import React, { useState } from "react";

// import Header from "./Header";

// import Footer from "./Footer";

import Note from "./Note";

import CreateArea from "./CreateArea";

function App() {

const [notes, setNotes] = useState([]);

function addNote(newNote) {

setNotes((prevNotes) => {

return [...prevNotes, newNote];

});

}

function editNote(id, text, dateCreated, lastModified){

const tobeUpdated = notes.find(singleNote => singleNote.id === id)

tobeUpdated.id = id

tobeUpdated.text = text;

tobeUpdated.dateCreated = dateCreated;

tobeUpdated.lastModified = lastModified;

setNotes([...notes])

}

function deleteNote(id) {

setNotes((prevNotes) => {

return prevNotes.filter((noteItem) => {

return noteItem.id !== id;

});

});

}

console.log(notes);

return (

{notes.map((noteItem) => {

return (

key={noteItem.id}

id={noteItem.id}

title={noteItem.title}

content={noteItem.content}

onDelete={deleteNote}

onEdit={editNote}

/>

);

})}

);

}

export default App;

import React, { useState } from "react";

// //import AddIcon from "@material-ui/icons/Add";

// import Fab from "@material-ui/core/Fab";

// import Zoom from "@material-ui/core/Zoom";

function CreateArea(props) {

const [isExpanded, setExpanded] = useState(false);

const [note, setNote] = useState({

id: "",

title: "",

content: ""

});

function handleChange(event) {

const { name, value } = event.target;

setNote((prevNote) => {

return {

...prevNote,

[name]: value

};

});

}

function submitNote(event) {

event.preventDefault();

setNote({

id: Math.floor(Math.random() \* 100),

title: "",

content: ""

});

props.onAdd(note);

}

function expand() {

setExpanded(true);

}

return (

{isExpanded && (

name="title"

onChange={handleChange}

value={note.title}

placeholder="Title"

/>

)}

name="content"

onClick={expand}

onChange={handleChange}

value={note.content}

placeholder="Take a note..."

rows={isExpanded ? 3 : 1}

/>

Submit

);

}

export default CreateArea;

import React, { useState } from "react";

//import DeleteIcon from "@material-ui/icons/Delete";

import "./styles.css";

function Note(props) {

const [displayForm, setForm] = useState(false);

function handleClick() {

props.onDelete(props.id);

}

function handleEdit(e) {

e.preventDefault();

const title = e.target.children[0].value;

const content = e.target.children[1].value;

props.onEdit(props.id, title, content);

setForm(false)

}

return (

{props.title}

{props.content}

onSubmit={handleEdit}

className={`${displayForm ? "show" : "hide"}`}

>

X

setForm(!displayForm)}>Edit

);

}

export default Note;

.App {

font-family: sans-serif;

text-align: center;

}

.show{

display: block;

}

.hide{

display: none;

}

1. [**Delete an existing note**](https://github.com/vivienfan/csbc1010-notes-app-server/blob/master/src/routes/note.js#L121-L163)

db.notes.remove({ note: note }, function(err, results) {

console.log(results);

next();

});

db.notes.update({ "notes.note": note },

{ $pull: { notes: { note: note } } },

function(err, results) {

console.log(results);

next();

});

1. [**Search notes by content**](https://github.com/vivienfan/csbc1010-notes-app-server/blob/master/src/routes/notes.js#L61-L116)

exports.getSearch = (req, res, next) => {

const { title } = req.query;

Note.find({title: { $regex: title, $options: "i" }})

.then(noteName => {

res.render('note/notes-list', {

prods: noteName ,

pageTitle: 'All Notes',

path: '/notes'

});

})

.catch(err => {

console.log(err);

});

}

1. [**Bulk delete all existing notes**](https://github.com/vivienfan/csbc1010-notes-app-server/blob/master/src/routes/notes.js#L118-L156)

const valuesToDelete = ['text'];

let attributes = [id, text, dateCreated, lastModified

] //your attributes data

attributes = attributes.filter(({ trait\_type }) => !valuesToDelete.includes(trait\_type));