npm install express jsonwebtoken bcrypt dotenv

**App.js**

let express = require('express');

let bodyParser = require('body-parser');

const session = require('express-session');

let connection = require('./dbcon.js');

let router = require('./routers.js');

//For JWT

let app = express();

app.use(bodyParser.json())

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

app.use(session({

    secret: 'your-secret-key', //used to sign the session ID cookie

    resave: false, //forces the session to be saved back to the session store, even if the session was never modified during the request

    saveUninitialized: false //forces a session that is "uninitialized" to be saved to the store

}))

app.use('/proj', router);

app.listen(3000, function(){

    console.log("Server is running");

})

**Routers.js**

let express = require('express');

let router = express.Router();

let connection = require('./dbcon.js');

let query = require('./queries.js');

router.post('/register/', function(req, res){

    const {U\_NAME,EMAIL,ROLE,PASSWORD} = req.body;

    if(U\_NAME == null || EMAIL == null || ROLE == null || PASSWORD == null){

        res.send("Please enter all the fields");

    }

    else{

        const values = [U\_NAME, EMAIL, ROLE, PASSWORD];

        connection.query(query.registerUserQuery, [values],(error, result)=>{

            if(error){

                if(error.code == 'ER\_DUP\_ENTRY'){

                    res.send("User Already Exists");

                }

                else{

                    console.error("Error while registering the user", error);

                    res.send("Error while registering the user");

                }

            }

            else{

                res.send("User Registered Successfully");

            }

        })

    }

})

//for login

router.post('/login/', function(req, res){

    const {EMAIL,PASSWORD} = req.body;

    if(EMAIL == null || PASSWORD == null){

        res.send("Please enter all the fields");

    }

    else{

        connection.query(query.loginUserQuery, [EMAIL, PASSWORD],(error, result)=>{

            if(error){

            console.log("Error while logging in", error);

            res.send("Error while logging in");

            }

            else if (result.length == 0){

                res.send("User does not exist");

            }

            else{

                const user = result[0];     //as email is unique, only one user will be returned

                req.session.userRole = user.ROLE;    //create a session for the user

                if(user.PASSWORD === PASSWORD){         //check if the password matches i.e case sensitive

                    res.send({message: "Login Successful", role: user.ROLE});

                    // if(ROLE === 'admin'){

                    //     res.redirect('/proj/insertBook/');

                    // }

                    // else{

                    //     res.send("Sorry, you are not an admin");

                    // }

                }

                else{

                    res.send("Invalid Credentials");

                }

            }

        })

    }

})

router.post('/insertBook/', function(req, res){

    const {BNAME, AUTHORS, GENRE} = req.body;

    // const userRole = req.body.role; // Get the user role from the request body

    const userRole = req.session.userRole;  // Get the user role from the session

    if(BNAME == null || AUTHORS == null || GENRE == null){

        res.send("Please enter all the fields");

    }

    else{

        if(userRole === 'admin'){

        const values = [BNAME, AUTHORS, GENRE];

        connection.query(query.insertBookQuery, [values],(error, result)=>{

            if(error){

                if(error.code === 'ER\_DUP\_ENTRY'){

                    res.send("Book Already Exists");

                }

                else{

                    console.error("Error while registering the user", error);

                    res.send("Error while registering the user");

                }

            }

            else{

                res.send("Book inserted successfully");

            }

        });

    }

    else{

        res.send("Sorry, you are not an admin");

    }

    }

});

router.delete('/delBook/', function(req, res){

    const {BID, BNAME, AUTHORS, GENRE} = req.body;

    // const userRole = req.body.role; // Get the user role from the request body

    const userRole = req.session.userRole;  // Get the user role from the session

    if(BID == null && BNAME == null && AUTHORS == null && GENRE == null){

        res.send("Please enter at least one field");

    }

    else{

        if(userRole === 'admin'){

        //const values = [BID, BNAME, AUTHORS, GENRE];

        connection.query(query.delBookQuery, [BID,BNAME,AUTHORS,GENRE],(error, result)=>{

            if(error){

                console.error("Error while deleting the book", error);

                res.send("Error while deleting the book");

            }

            else{

                res.send("Book deleted successfully");

            }

        });

    }

    else{

        res.send("Sorry, you are not an admin");

    }

    }

});

router.put('/updateBook/', function(req, res){

    const {BID, BNAME, AUTHORS, GENRE} = req.body;

    // const userRole = req.body.role; // Get the user role from the request body

    const userRole = req.session.userRole;  // Get the user role from the session

    if(BID == null || BNAME == null || AUTHORS == null || GENRE == null){

        res.send("Please enter all the fields");

    }

    else{

        if(userRole === 'admin'){

        //const values = [BNAME, AUTHORS, GENRE, BID];

        connection.query(query.updateBookQuery, [BNAME, AUTHORS, GENRE, BID],(error, result)=>{

            if(error){

                console.error("Error while updating the book", error);

                res.send("Error while updating the book");

            }

            else{

                res.send("Book updated successfully");

            }

        });

    }

    else{

        res.send("Sorry, you are not an admin");

    }

    }

})

router.get('/searchBook/', function(req, res){

    const {BNAME, AUTHORS, GENRE} = req.body;

    if(BNAME == null && AUTHORS == null && GENRE == null){

        res.send("Please enter at least one field");

    }

    else{

        connection.query(query.getBooksQuery, [BNAME, AUTHORS, GENRE],(error, result)=>{

            if(error){

                console.error("Error while searching the book", error);

                res.send("Error while searching the book");

            }

            else{

                res.send(result);

            }

        });

    }

})

module.exports = router;

**Queries.js**

const registerUserQuery = 'INSERT INTO USERS (U\_NAME,EMAIL,ROLE,PASSWORD) VALUES (?)';

const loginUserQuery = 'SELECT \* FROM USERS WHERE EMAIL = ? AND PASSWORD = ?';

const insertBookQuery = 'INSERT INTO BOOKS (BNAME, AUTHORS, GENRE) VALUES (?)';

const getBooksQuery = 'SELECT \* FROM BOOKS WHERE BNAME = ? OR AUTHORS = ? OR GENRE = ?';

const delBookQuery = 'DELETE FROM BOOKS WHERE BID = ? OR BNAME = ? OR AUTHORS = ? OR GENRE = ?';

const updateBookQuery = 'UPDATE BOOKS SET BNAME = ?, AUTHORS = ?, GENRE = ? WHERE BID = ?';

module.exports = {registerUserQuery, loginUserQuery, insertBookQuery, getBooksQuery,

                    delBookQuery, updateBookQuery};

**dbcon.js**

let mysql = require('mysql');

const connection = mysql.createConnection({

    host : 'localhost',

    user : 'root',

    password : 'password',

    database : 'bookdirectory'

});

connection.connect(error =>{

    if(error){

        console.log("There is some error while creating database connection", error);

        process.exit(1);

    }

    console.log("Database connected successfully")

})

module.exports = connection;