//Create all of creates, reads, deletes etc.. here...

const createError = require('http-errors');

const {MongoClient, ServerApiVersion, ObjectId} = require ('mongodb');

// const uri = "mongodb+srv://RBennett:pw5489@cluster0.qkqez51.mongodb.net/?retryWrites=true&w=majority";

// const client = new MongoClient (uri, {useNewUrlParser: true, useUnifiedTopology: true, serverApi: ServerApiVersion.v1});

const {Book} = require ('./models/books')

// let booklist = []

// let idnumber = 0

exports.index = function (req, res) {

    Book.find()

    .then(books =>

        res.send(books))

}

    // client.connect (async (error) => {

    //     const findResult = client.db('myFirstDatabase').collection('books').find();

    //     const result = await findResult.toArray()

    //     res.send(result)

    // })

    // res.send(booklist)

//CREATE

//put in some errors alerts to make sure people type in the correct formats

exports.create = function (req, res, next) {

    if(!req.body.title) { //name could be changed to Author/Title etc...

        return(next(createError(400, "name is required")))

    }

    const book = new Book({title: req.body.title, author: req.body.author, available: req.body.available})

    book.save()

    .then(() => res.send({result:true}))

}

    //using Mongo Database connection

    // client.connect ((error) => {

    //     client.db('myFirstDatabase').collection('books').insertOne({

    //         // id: idnumber,

    //         title: req.body.title,

    //         author: req.body.author,

    //         available: req.body.available

    //     })

    //     .then(() => {res.send({result:true})})

    // })

    // booklist.push({id: idnumber, title: req.body.title,  author: req.body.author,  available: req.body.available}) //push all variables that are required, e.g. title, author, read/not read, ID number etc...

    // res.send({result: "true"})

    // idnumber++ //everytime we create new name/data the id will increase by 1

//FIND

//find the book - is it available?

exports.show = function (req, res, next) {

    Book.findOne({\_id: ObjectId(req.params.id)})

    .then ((bookitem) => {

        if(!bookitem) {

            return (next(createError(404, "No book with that ID")))

        }

        res.send(bookitem)

    })

}

    // const bookitem = booklist.find((book) => book.id == req.params.id)

    // if(!bookitem) {

    //     return(next(createError(404, "no book with that id")))

    // }

    // res.send(bookitem)

//DELETE BY ID

//add the delete functionality to the API

exports.delete = function (req, res, next) {

    Book.deleteOne({\_id: ObjectId(req.params.id)})

    .then ((r) => {

        if(!r.deletedCount) {

            return res.send ({results: true})

        }

        return (next(createError(404, "No book with that ID")))

    })

}

     //using Mongo Database connection

    //  client.connect ((error) => {

    //     client.db('myFirstDatabase')

    //     .collection('books')

    //     .deleteOne({\_id: ObjectId(req.params.id)},)

    //     .then((result) => {

    //         if (result.deletedCount){

    //         return res.send({result:true})

    //     }

    //     return (next(createError(404, "No book with that ID")))

    //     })

    // })

    // const bookitem = booklist.find((book) => book.id == req.params.id)

    // if(!bookitem) {

    //     return(next(createError(404, "no book with that id")))

    // }

    // booklist = booklist.filter((book) => book.id != req.params.id)

    // res.send({result: true})

//EDIT / UPDATE

//add the edit functionality to the API

exports.update = function (req, res, next) {

    // const bookitem = booklist.find((book) => book.id == req.params.id)

    if(!req.body.title) {

        return(next(createError(400, "name is required")))

    }

    Book.findOne({\_id: ObjectId(req.params.id)})

    .then ((result) => {

        if(!result) {

            return (next(createError(404, "No book with that ID")))

        }

         result.title = req.body.title

         result.available = req.body.available

         result.save()

         .then(() => res.send({result: true}))

    })

}

     //using Mongo Database connection

    //  client.connect ((error) => {

    //     client.db('myFirstDatabase').collection('books').updateOne(

    //         {\_id: ObjectId(req.params.id)},

    //         {$set: {

    //             title: req.body.title,

    //             author: req.body.author,

    //             available: req.body.available

    //         }}

    //     )

    //     .then((result) => {

    //         if (result.matchedCount){

    //         return res.send({result:true})

    //     }

    //     return (next(createError(404, "No book with that ID")))

    // })

    // })

    // booklist = booklist.map((book) => {

    //     if (book.id == req.params.id) {

    //         book.title = req.body.title

    //         book.author = req.body.author

    //         book.available = req.body.available

    //       }

    //     return book

    // })

    // res.send({result: true})