Notes for the interactive lecture

1. Create a index.js file inside the server folder

2. create a basic express server with cors

const express = require('express')

const cors = require('cors')

//middlewares

app.use(express.json())

app.use(cors())

const port = 4004

app.listen(port, () = console.log(`running on port ${port}`)

3. create a controllers folder and a controllers.js file in it

4. inside of the controller:

const movies = require('../db.json')

module.exports = {

getMovies: (req, res) => {

res.status(200).send(movies)

}

}

5. inside the server index.js, add the controller to the top

const ctrl = require('./controller')

6. create the movie GET endpoint

//movie endpoints

app.get('/api/movies', ctrl.getMovies)

7. back to the controller.js, create a delete method

deleteMovie: (req, res) => {

console.log(req.params)

}

8. in the server index.js, create the delete endpoint

app.delete('/api/movies/:id', ctrl.deleteMovie)

9 in the controller, we can destructer id off req.params. add the following, to the delete method:

const { id } = req.params

console.log(id)

console.log(type of)

10 lets find the index for the movie we are trying to delete

const index = movies.findIndex((movie) => {

return movie.id === id

})

11. demonstrate the delete method on the website, you should see a -1 in the console.log()

12. add the + to id in the findIndex method

13. now that we know the index of the movie we want to delete, what method do you think we can use to remove it from the array?

movies.splice(index, 1)

res.status(200).send(movies)

14. because we are passing a parameter than is visible to the user in the url, there is a chance for errors if the user were to directly go to the url and try to delete a movie. so lets handle that error

if(+id > 10 || +id === -1){

// res.sendStatus(400)

res.status(400).send({error: 'you tried to remove a movie id that doesnt exist'})

} else {

movies.splice(index, 1)

res.status(200).send(movies)

}

15. lets work on adding a new movie to our website. inside the controller (show the console.log(req.body) first)

createMovie: (req, res) => {

const { title, rating, imageURL } = req.body

console.log(title, rating, imageURL)

}

16. create the endpoint

app.post('/api/movies', ctrl.createMovie)

17. we need to have our server automatically create our id's for us because we arent using a database with serial primary key

create the global id outside module.exports

at the top of the page

let id = 11

then inside the post method

const newMovie = {

id, title, rating, imageURL

}

movies.push(newMovie)

res.status(200).send(movies)

id++

18. finally lets create the PUT method in the controller

updateMovie: (req, res) => {

const { id } = req.params

const index = movies.findIndex( movie => {

return movie.id === +id

})

console.log(req.body) //**(console.log this to show whether + or - was pressed and show the students)**

}

19. create the endpoint

app.put('/api/movies/:id', ctrl.updateMovie)

20. now go test out the +/- button on the website

21. add this to the updateMovie method:

const currRating = movies[index].rating

console.log(currRating)

22. error handling because we dont want ratings to go above 5 or below 1

const { type } = req.body

if (currRating === 5 && type === 'plus') {

res.status(400).send({error: 'no can do'})

} else if (currRating === 0 && type === 'minus') {

res.status(400).send({error: 'no can do'})

} else if (type === 'plus') {

movies[index].rating++

res.status(200).send(movies)

} else if (type === 'minus'){

movies[index].rating--

res.status(200).send(movies)

} else {

res.status(400).send({error: 'failed to update'})

}

23. if there is extra time, try to create a global function for your findIndex method that you can reuse in any of the endpoint handler methods