Refactoring Animals Routes

6 min

Okay, training wheels off! Now refactor all your /animals routes to **animals.js**.

Instructions

1. Checkpoint 1 Passed

1.

Now, do the same refactoring for /animals routes into an animalsRouter Router that you create in **animals.js**.

app.js

```
const express = require('express');
const app = express();

const { getElementById, getIndexById, updateElement,
  seedElements, createElement } = require('./utils');

const PORT = process.env.PORT || 4001;

app.use(express.static('public'));

const expressionsRouter = require('./expressions.js');
app.use('/expressions', expressionsRouter);

const animalsRouter = require('./animals.js');
app.use('/animals', animalsRouter);

app.listen(PORT, () => {
  console.log(`Server is listening on ${PORT}`);
});
```

expressions.js

```
const express = require('express');

const { getElementById, getIndexById, updateElement,
  seedElements, createElement } = require('./utils');

let expressions = [];
seedElements(expressions, 'expressions');

expressionsRouter = express.Router();

// Get all expressions
expressionsRouter.get('/', (req, res, next) => {
  res.send(expressions);
});

// Get a single expression
expressionsRouter.get('/:id', (req, res, next) => {
  const foundExpression = getElementById(req.params.id, expressions);
  if (foundExpression) {
    res.send(foundExpression);
  } else {
    res.status(404).send();
  }
});

// Update an expression
expressionsRouter.put('/:id', (req, res, next) => {
  const expressionIndex = getIndexById(req.params.id, expressions);
  if (expressionIndex !== -1) {
    updateElement(req.params.id, req.query, expressions);
    res.send(expressions[expressionIndex]);
  } else {
    res.status(404).send();
  }
});

// Create an expression
expressionsRouter.post('/', (req, res, next) => {
  const receivedExpression = createElement('expressions', req.query);
  if (receivedExpression) {
    expressions.push(receivedExpression);
    res.status(201).send(receivedExpression);
  } else {
    res.status(400).send();
  }
});

// Delete an expression
expressionsRouter.delete('/:id', (req, res, next) => {
  const expressionIndex = getIndexById(req.params.id, expressions);
  if (expressionIndex !== -1) {
    expressions.splice(expressionIndex, 1);
    res.status(204).send();
  } else {
    res.status(404).send();
  }
});

module.exports = expressionsRouter;
```

animals.js

```
const express = require('express');

const { getElementById, getIndexById, updateElement,
  seedElements, createElement } = require('./utils');

let animals = [];
seedElements(animals, 'animals');

animalsRouter = express.Router();

// Get all animals
animalsRouter.get('/', (req, res, next) => {
  res.send(animals);
});

// Get a single animal
animalsRouter.get('/:id', (req, res, next) => {
  const animal = getElementById(req.params.id, animals);
  if (animal) {
    res.send(animal);
  } else {
    res.status(404).send();
  }
});

// Create an animal
animalsRouter.post('/', (req, res, next) => {
  const receivedAnimal = createElement('animals', req.query);
  if (receivedAnimal) {
    animals.push(receivedAnimal);
    res.status(201).send(receivedAnimal);
  } else {
    res.status(400).send();
  }
});

// Update an animal
animalsRouter.put('/:id', (req, res, next) => {
  const animalIndex = getIndexById(req.params.id, animals);
  if (animalIndex !== -1) {
    updateElement(req.params.id, req.query, animals);
    res.send(animals[animalIndex]);
  } else {
    res.status(404).send();
  }
});

// Delete a single animal
animalsRouter.delete('/:id', (req, res, next) => {
  const animalIndex = getIndexById(req.params.id, animals);
  if (animalIndex !== -1) {
    animals.splice(animalIndex, 1);
    res.status(204).send();
  } else {
    res.status(404).send();
  }
});

module.exports = animalsRouter;
```