Practice Exercises

You can solve practice these exercises on your own setup and learn how to make your custom web server accept data from a form or JSON. In addition to this, learn how to add a route and fetch data from it. Helper codes are provided below but try to solve it on your own first.

We'll cover the following

1. T-shirt Color

Add a `"/tshirt"` route to your server for handling the submission of form data containing a `size` and a `color` field, like in the chapter 23 example. In the route callback, send back a confirmation message to the client.


```
app.post("/tshirt", upload.array(), (request, response) => {
  const size = request.body.size;
  const color = request.body.color;
  response.send(`Command received! Size: ${size}, color: ${color}`);
});
```

2. Visited Countries

Add a `"/api/countries"` route to your server to manager traveler information received as JSON data, like in the chapter 23 exercise. In the route callback, send back a confirmation message to the client.

```
app.post("/api/countries", jsonParser, (request, response) => {
  const traveler = request.body;
  response.send(
    `Your name is ${traveler.name} and you visited ${traveler.countries
      .length} countries. Keep traveling!`
  );
});


app.get("/api/articles", (request, response) => {
  response.json(countries);
});
```

3. New Article

Add a `"/articles"` route to your server. This route should accept a new blog article as form data and add it to the server's article list, like in the chapter 23 exercise. The new article ID must be equal to the maximum ID among existing articles plus one.


```
// Define an article list
const articles = [
  { id: 1, title: "First article", content: "Hello World!" },
  {
    id: 2,
    title: "Lorem ipsum",
    content:
      "Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut hendrerit mauris ac portti
  },
  {
    id: 3,
    title: "Lorem ipsum in French",
    content:
      "J'en dis autant de ceux qui, par mollesse d'esprit, c'est-à-dire par la crainte de la
  }
];
```

```
// Web route
app.post("/articles", upload.array(), (request, response) => {
  const title = request.body.title;
  const content = request.body.content;
  // Create a new array containing only ids
  const idList = articles.map(article => article.id);
  // Reducing the array to the maximum id value
  const maxId = idList.reduce((acc, value) => {
    if (value > acc) return value;
    return acc;
    // Or: (value > acc) ? value : acc;
  });
  const id = maxId + 1;
  // Add new article to the list
  articles.push({ id, title, content });
  response.send(`New article added successfully with ID ${id}!`);
});


// JSON API

app.get("/api/articles", (request, response) => {
  response.json(articles);
});


// Start listening to incoming requests
// If process.env.PORT is not defined, port number 3000 is used
const listener = app.listen(process.env.PORT || 3000, () => {
  console.log(`Your app is listening on port ${listener.address().port}`);
});
```