GET and POST

Learning Objectives:

- Learning basic express routing syntax
- Learning how to redirect from one route to another
- Learning how to set our server up to receive data sent from a post request
- Understanding how we can chain most express methods

Basic Routing Syntax

Because we are using Express primarily to build our APIs, it's best practice to start every such route with "/api" which will help us avoid route collisions with React's client-side routing.

GET Data

```
// we can hard code some users like this
// later on we will want to store users in a database
const users = [
    { firstName: "Reimu",  lastName: "Hakurei"    },
    { firstName: "Marisa", lastName: "Kirisame"   },
    { firstName: "Sanae",  lastName: "Kochiya"    },
    { firstName: "Sakuya", lastName: "Izayoi"     },
    { firstName: "Momiji", lastName: "Inubashiri" }
];

app.get("/api/users", (req, res) => {
    res.json( users );
});
```

POST Data

In order to be able to access POST data, we need to be able to pull it out of the request object. To do this, we first have to add a new setting to our server.js file:

```
// make sure these lines are above any app.get or app.post code blocks
app.use( express.json() );
app.use( express.urlencoded({ extended: true }) );
```

both `express.urlencoded()` and `express.json()` are ***Express middleware functions***. They are responsible for providing and parsing the request.body data.

Now that we have included our middleware, here's how we get form data:

```
app.post("/api/users", (req, res) => {
    // req.body will contain the form data from Postman or from React
    console.log(req.body);
    // we can push it into the users array for now...
    // later on this will be inserted into a database
    users.push(req.body);
    // we always need to respond with something
    res.json( { status: "ok" } );
});
```

Route Parameters

Getting Data from a URL

Any data you wish to pass via the URL must be indicated by a ':'. It will then be available in the req.params object:

```
// if we want to get a user with a specific id, we can make the id a part of the url
// be sure to preface the id variable with a `:` colon
app.get("/api/users/:id", (req, res) => {
    // we can get this `id` variable from req.params
    console.log(req.params.id);
    // assuming this id is the index of the users array we could return one user this way
    res.json( users[req.params.id] );
});
```

Update Data

updating data using a put request:

```
app.patch("/api/users/:id", (req, res) => {
    // we can get this `id` variable from req.params
    const id = req.params.id;
    // assuming this id is the index of the users array we can replace the user like so
    users[id] = req.body;
    // we always need to respond with something
    res.json( { status: "ok" } );
});
```

Deleting Data

deleting data using a delete request.

```
app.delete("/api/users/:id", (req, res) => {
    // we can get this `id` variable from req.params
    const id = req.params.id;
    // assuming this id is the index of the users array we can remove the user like so
    users.splice(id, 1);
    // we always need to respond with something
    res.json( { status: "ok" } );
});
```