by: @bruno_kiafuka

> Representational State Transfer: is what an API that uses http requests to manipulate data. REST technology is generally preferred to the more robust Simple Object Access Protocol (SOAP) technology because REST leverages less bandwidth, making it more suitable for internet usage.

Creating a new project

- Install yarn
- open your terminal
- run `mkdir <Name of your project>`
- cd `cd <Name of your project>`
- run `yarn init -y` to create the package.json file
- open the project using your favorite folder

> For this project we will use express as framework.

Adding a script

Create a folder called src which will house your files, then create an `index.js` file. Open the file you created and add the code bellow

```
console.log("hello world")
```

Then open your package.json and add the lines below.

```
"scripts": {
    "dev": "node src/index.js"
  },
```

Lastely on your terminal run `yarn dev`. Then you should get a result similar to the one below:

```
yarn run v1.15.2
$ node src/index.js
hello world
```

Run express server

> You will want to download an http client such as insomnia or postman to test your requests.

Firstly we need to add express as a dependency into our project. To install express we will need to run the following command `yarn add express`

Open your index.js and add the following code:

```
const express = require("express")
const app = express()

app.listen(3333, () => console.log("Server running @ port 3333"))
```

Adding first method to our express server

We will add a `get` path that returns a json object.

```
const express = require("express")
const app = express()

app.get("/", (req, res) => {
  res.json({ message: "hello world" })
})

app.listen(3333, () => console.log("Server running @ port 3333"))
```

- `"/"` - endpoint path
- `req` - our http request
- `res`- our http response

The code above allows user to send a get request and and we get a json response.

What if we want to send data to our RESTFull server?

To send a request method we need to send a `post` request to our server. So we can add after the `get` our code so it can look like this:

```
app.post("/", (req, res) => {
  const { name } = req.body

  res.json({ message: "success", name })
})
```

- `req.body` - contains a json object with our http request body.

When trying this you might get an error like the one bellow:

```
TypeError: Cannot destructure property `name` of 'undefined' or 'null'.
```

> To fix this you will need to install a middleware called body-parser. Body-parser is a middleware that allows data sent to your server on the fomarts such as json or xml, etc...

After adding the `body-parser` your code should look like this:

```
const express = require("express")
const app = express()
const bodyParser = require("body-parser")

// parse application/json
app.use(bodyParser.json())

app.get("/", (req, res) => {
  res.json({ message: "hello world" })
})

app.post("/", (req, res) => {
  const { name } = req.body

  res.json({ message: "user", name })
})

app.listen(3333, () => console.log("Server running @ port 3333"))
```

> On the next post we will be making improvements on the our server.

👉 Github Repo