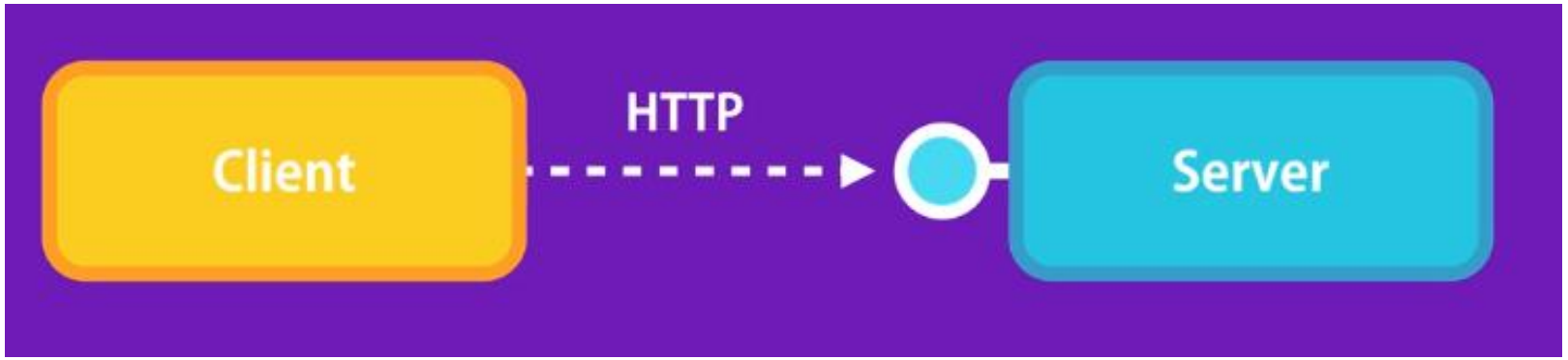
NODE

BUILDING RESTFUL API'S USING EXPRESS

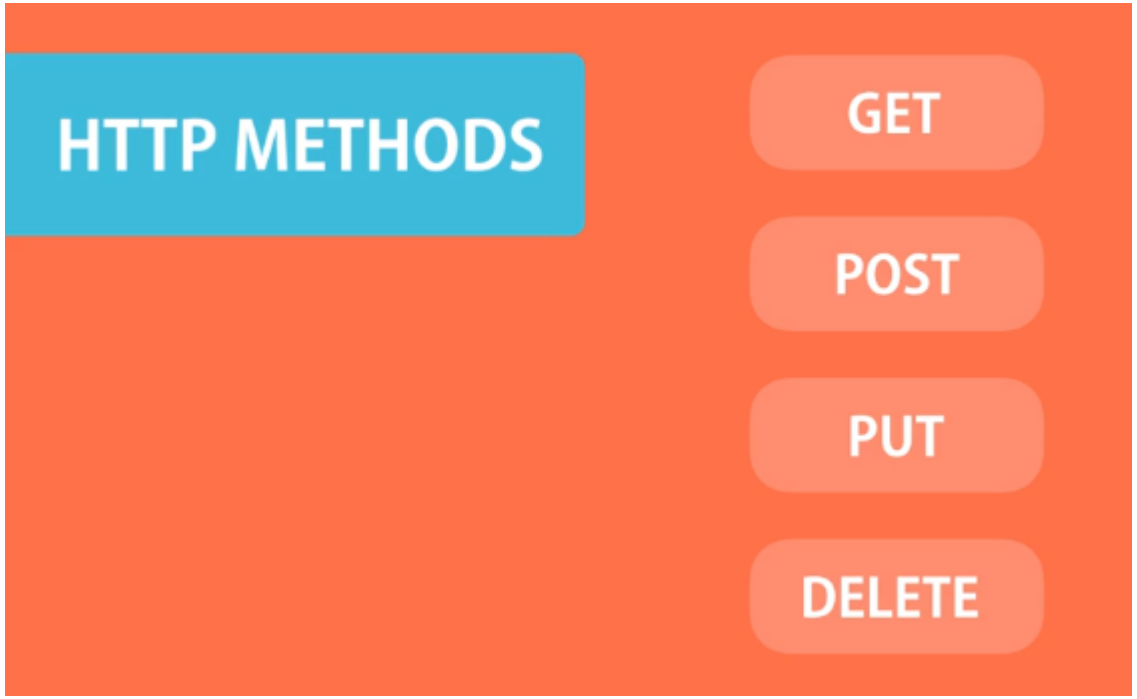
Client/Server Architecture

Client/Server Architecture



REpresentational State Transfer (REST)

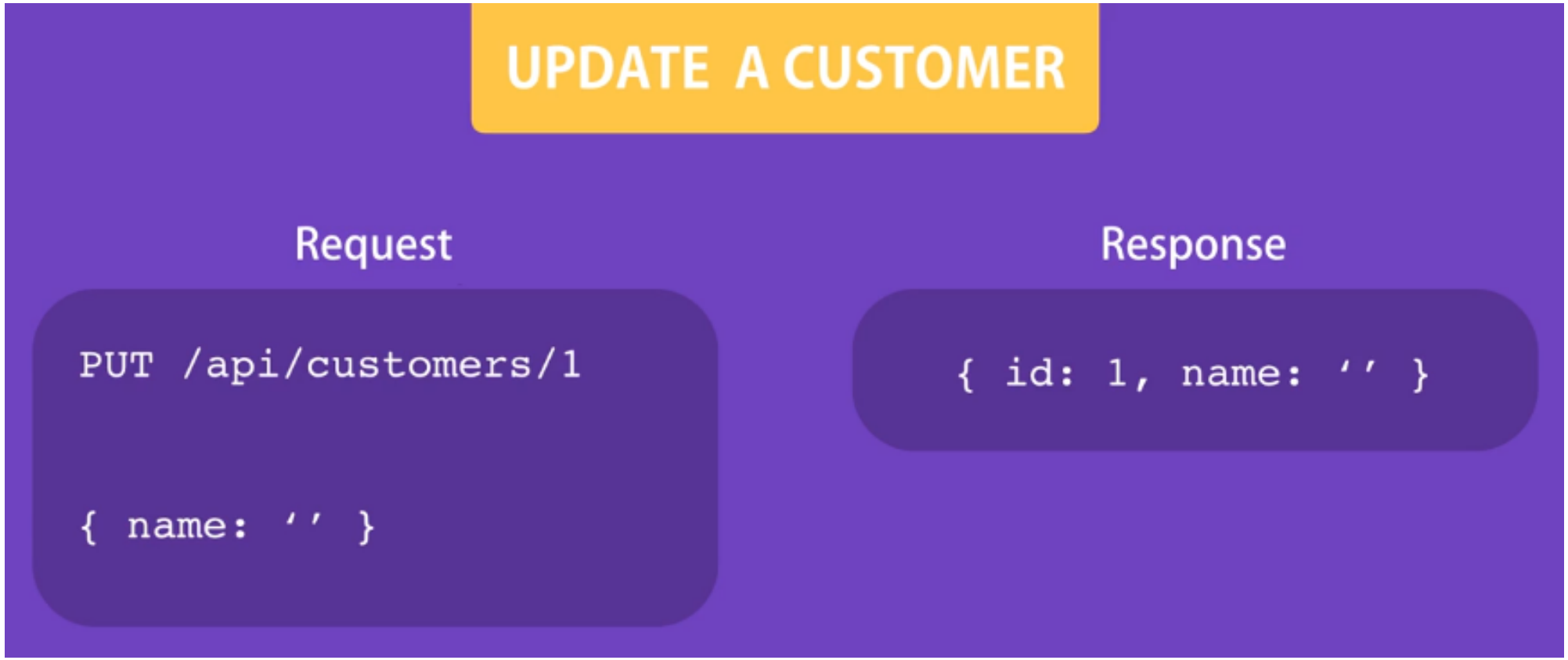
http://usman.com/api/customers



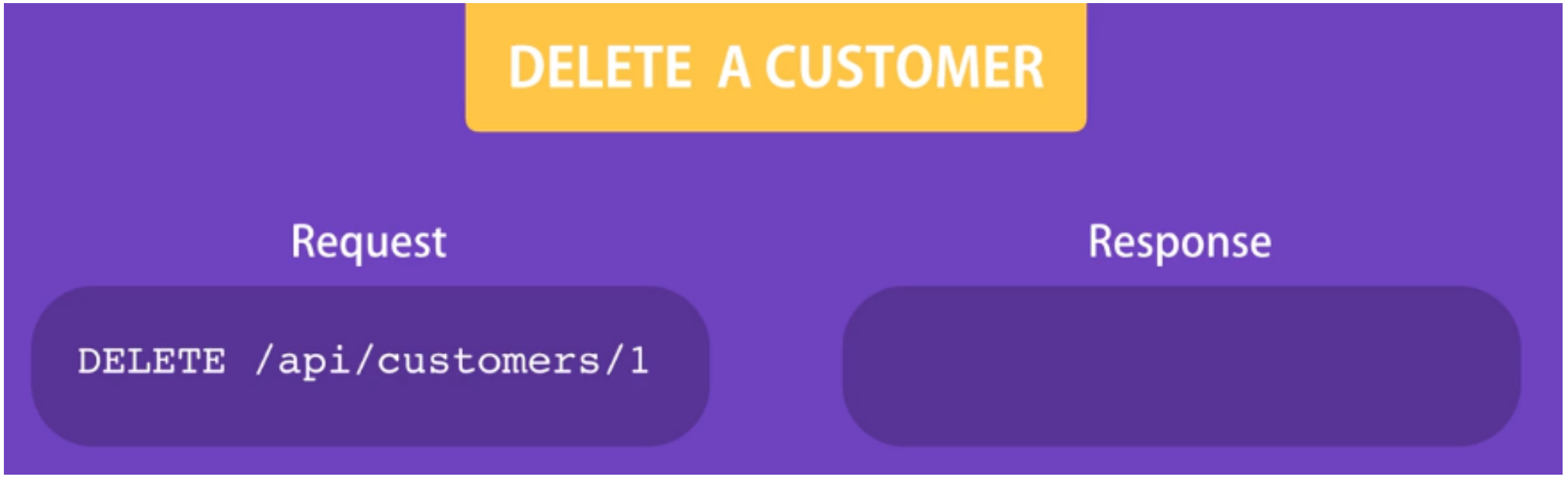
http://usman.com/api/customers

http://usman.com/api/customers/1

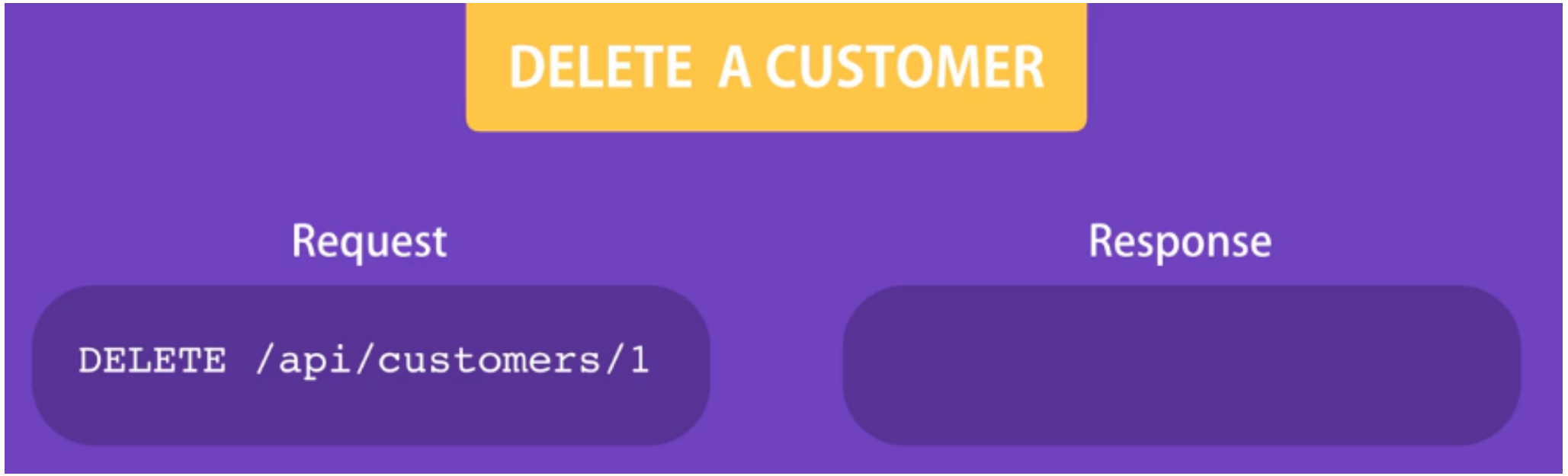
http://usman.com/api/customers/1



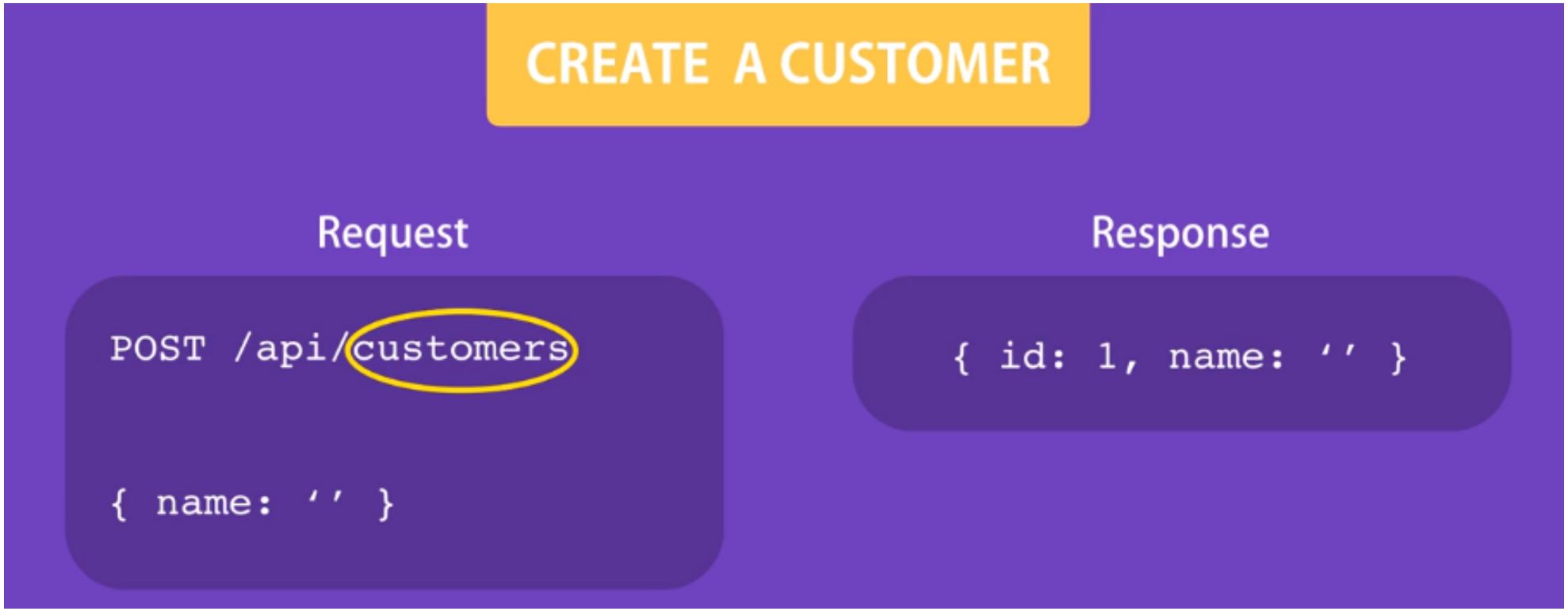
http://usman.com/api/customers/1



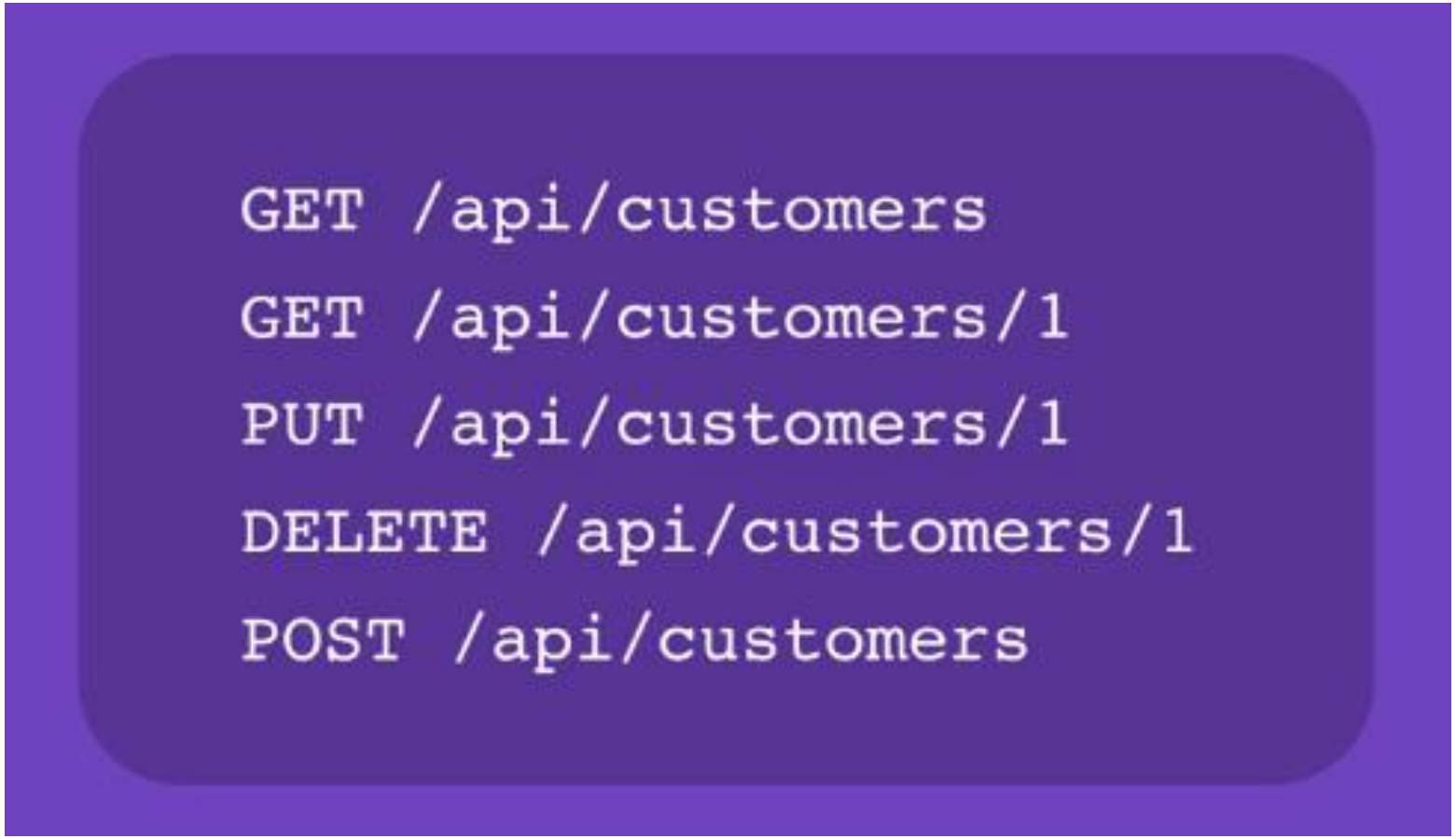
http://usman.com/api/customers/1



http://usman.com/api/customers



All RESTFUL calls



Pain to code like this

```
 const server =
http.createServer((req,res)=>{
  if(req.url==='/'){
  res.write(JSON.stringify([1,4,5,6]));
  res.end();
  }
 });
```

Express
https://www.npmjs.com/package/express

15 Million Downloads a Month

npm i express

Express

```
const express = require('express');
const app = express();
app.use(express.json()); //middleware
//handle api calls here
```

Express

```
const port = 3000;
app.listen(port,function(){
console.log(`Listening on Port 3000....`);
})
```

Express API Calls

```
let courses = [
{id:1,name:'Operating Systems'},
{id:2,name:'Math'}
];
```

Get call

```
app.get('/',function (request,response) {
 response.send("Hello World");
});
```

Send array back to client

```
app.get('/api/courses',function (request,response)
{
response.send([1,3,4,5,5,8]);
});
```

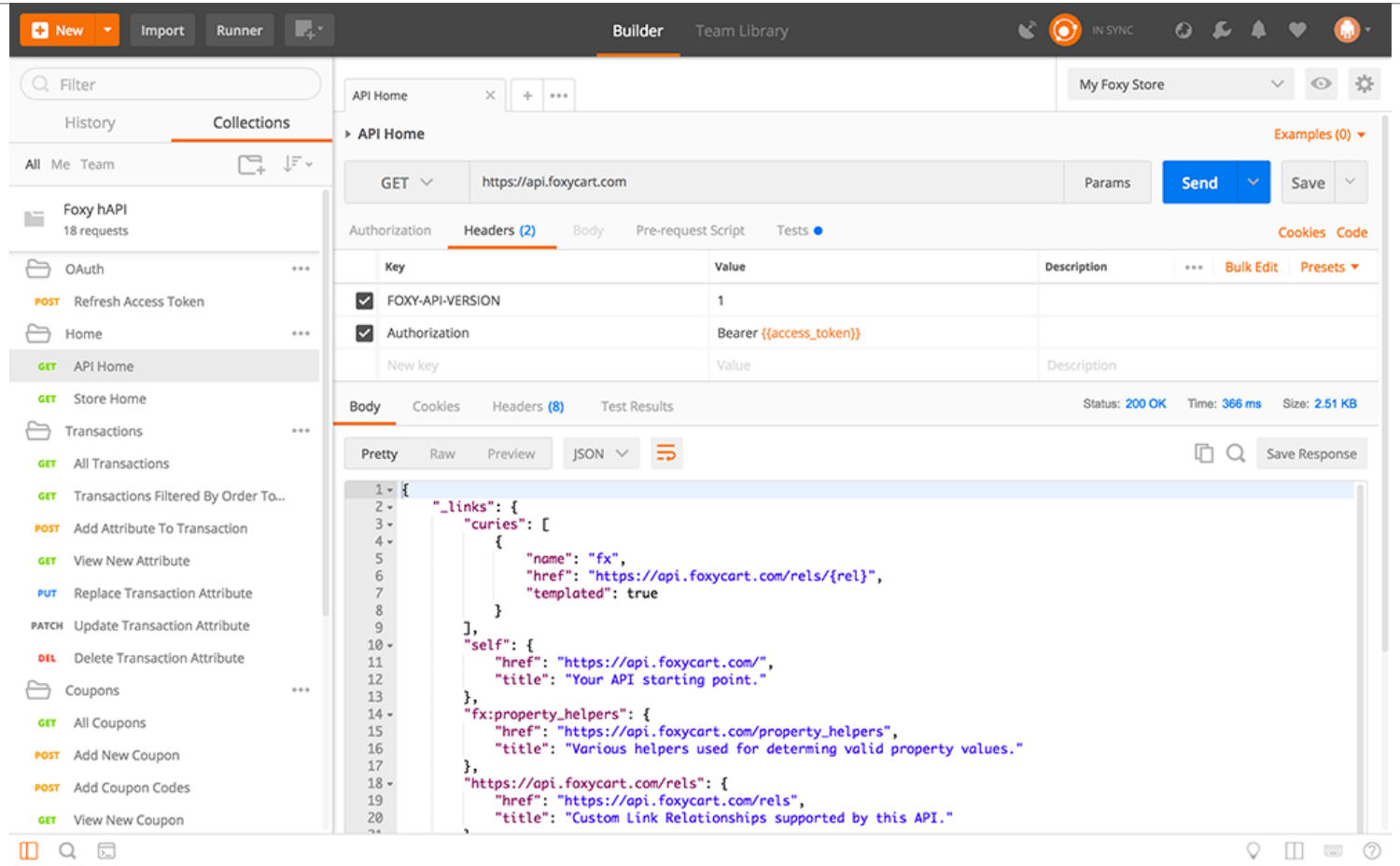
Route parameters

```
app.get('/api/courses/:id',function (request,response) {
//request.query to get query parameters
const course = courses.find(c=>c.id==request.params.id);
if(!course)
return response.status(404).send("Course not found for given ID");
response.send(course);
});
```

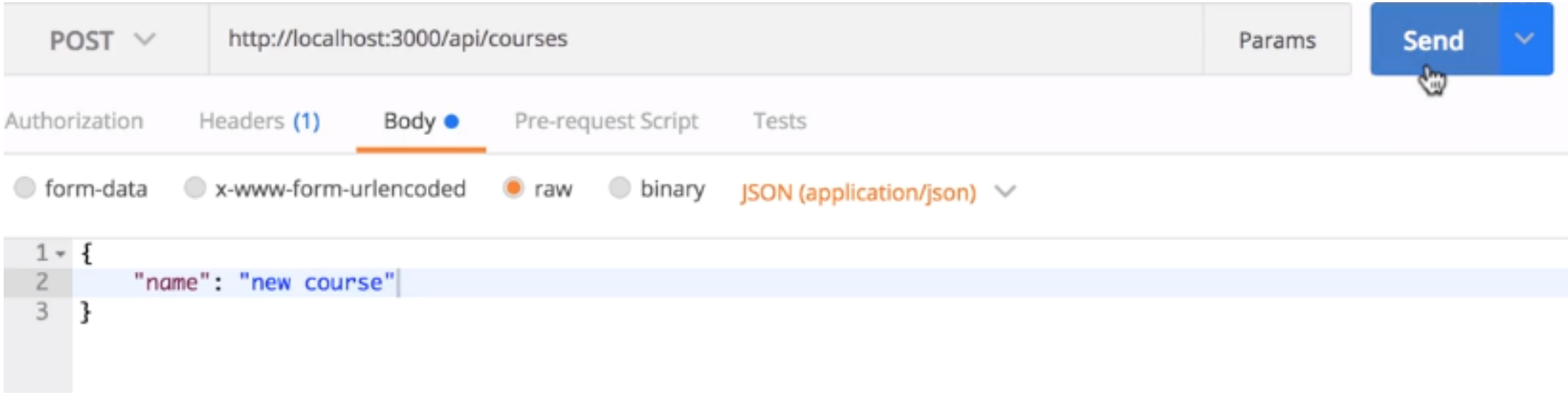
Post request

```
app.post('/api/courses',(request,response)=>{
  if(!request.body.name) response.status(400).send("Body Not provided");
  else {const course = {
  id:courses.length+1,
  name:request.body.name
}
courses.push(course);
response.send(course);
}});
```

Postman


New
Import
Runner
Builder
Team Library
IN SYNC
Filter
History
Collections
All Me Team
Foxy hAPI
18 requests
OAuth
POST Refresh Access Token
Home
GET API Home
GET Store Home
Transactions
GET All Transactions
GET Transactions Filtered By Order To...
POST Add Attribute To Transaction
GET View New Attribute
PUT Replace Transaction Attribute
PATCH Update Transaction Attribute
DEL Delete Transaction Attribute
Coupons
GET All Coupons
POST Add New Coupon
POST Add Coupon Codes
GET View New Coupon
API Home
My Foxy Store
API Home
Examples (0)
GET
https://api.foxycart.com
Params
Send
Save
Authorization
Headers (2)
Body
Pre-request Script
Tests
Cookies
Code
Key
Value
Description
Bulk Edit
Presets
FOXY-API-VERSION
1
Authorization
Bearer {{access_token}}
New key
Value
Description
Body
Cookies
Headers (8)
Test Results
Status: 200 OK
Time: 366 ms
Size: 2.51 KB
Pretty
Raw
Preview
JSON
Save Response
{
"_links": {
"curies": [
{
"name": "fx",
"href": "https://api.foxycart.com/rels/{rel}",
"templated": true
}
],
"self": {
"href": "https://api.foxycart.com/",
"title": "Your API starting point."
},
"fx:property_helpers": {
"href": "https://api.foxycart.com/property_helpers",
"title": "Various helpers used for determing valid property values."
},
"https://api.foxycart.com/rels": {
"href": "https://api.foxycart.com/rels",
"title": "Custom Link Relationships supported by this API."

Postman


POST
http://localhost:3000/api/courses
Params
Send
Authorization
Headers (1)
Body
Pre-request Script
Tests
form-data
x-www-form-urlencoded
raw
binary
JSON (application/json)
1 {
2 "name": "new course"
3 }

Validation with joi

npm I joi

```
const Joi = require('joi');
const schema = {
name:Joi.string().min(3).required()
}
```

Validation with joi

```
 const result = Joi.validate(request.body,schema);

 //console.log(result);

 if(result.error) response.status(400).send(result.error.details[0].message);
```

Clean joi

```
function validateCourse(course) {
const schema = {
name: Joi.string().min(3).required()
};
const result = Joi.validate(course, schema);
return result;
} //require joi at top preferably extract a
//module
```

Put Request

```
app.put('/api/courses/:id',(request,response)=>{
const course = courses.find(c=>c.id==request.params.id);
if(!course) {
response.status(404).send("Course not found for given ID"); return;
}
course.name = request.body.name;
const {error} = validateCourse({name:course.name});
```

Put Request

```
 if(error) {
 response.status(400).send(error.details[0].
message);
 return;
 }
 response.send(course);
 });
```

Delete Request

```
app.delete("/api/courses/:id", (request, response) => {
const course = courses.find(c => c.id == request.params.id);
if (!course) {
response.status(404).send("Course not found for given ID"); return;
}
```

Delete Request

```
const index = courses.indexOf(course);
courses.splice(index, 1);
response.send(course);
});
```

Grab the source code and get started

https://1drv.ms/f/s!AtGKdbMmNBGdhHN4tlJnBBdKqjUm

Run

> npm install

>node index.js

Go through the code

Express- Advanced Topics

GO PRO

Middleware

```
app.get('/',function (request,response) {
response.send("Hello World");
});
```

Is technically a middleware. It breaks the request response cycle

What if we do something here and then don’t send response.

Adding a middleware

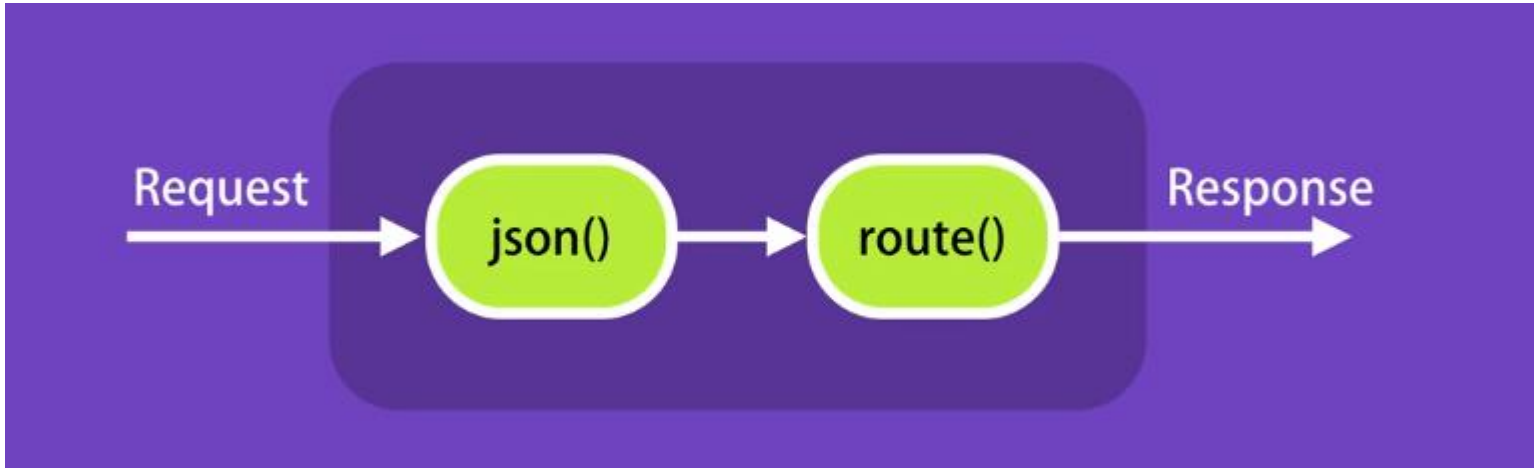
```
const express = require('express');
const app = express();
app.use(express.json());
```

Get request

If body contain json then parse it

Forward it to next middleware.

Request processing pipeline



Custom middleware

```
function log(request,response,next){
console.log("Logging...");
next();
}
module.exports = log;
```

Using your own middleware

```
const logger = require('./logger-middleware');

app.use(logger);
```

First, run npm install helmet

helmet

helmet helps you secure your Express apps by setting various HTTP headers. It's not a silver bullet, but it can help!

First, run npm install helmet

```
const express = require('express')
const helmet = require('helmet')
const app = express()
app.use(helmet())
```

Exercise

Check what morgan is and how it works

Info:

Urlencoded middleware is built in. It parse info from url

HTML ?

PUG

```
app.use(express.static('public'));
app.set('view engine', 'pug');
app.get('/', function (request, response) {
//response.send("Hello World");
response.render('index', { title: "My Paoge Title", message: 'Hello Hareem h1 Tag' });
});
```

/views/index.pug

```
html
  head
    title=title
  body
    h1=message
```