Dataforståelse

Building an API

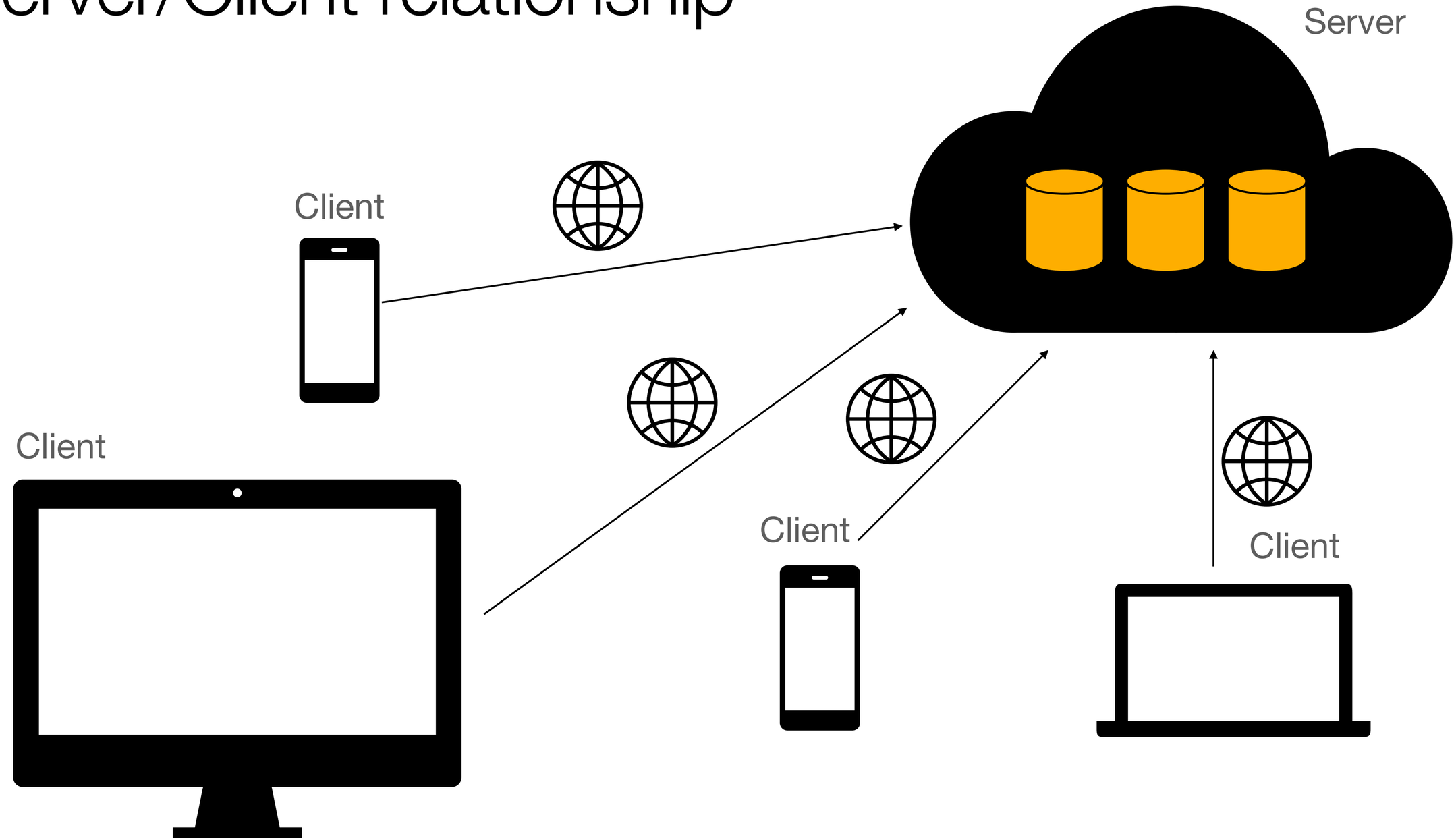
NIFR 2024 - https://github.com/nicklasdean/nodejs-examples

Agenda

Introduction

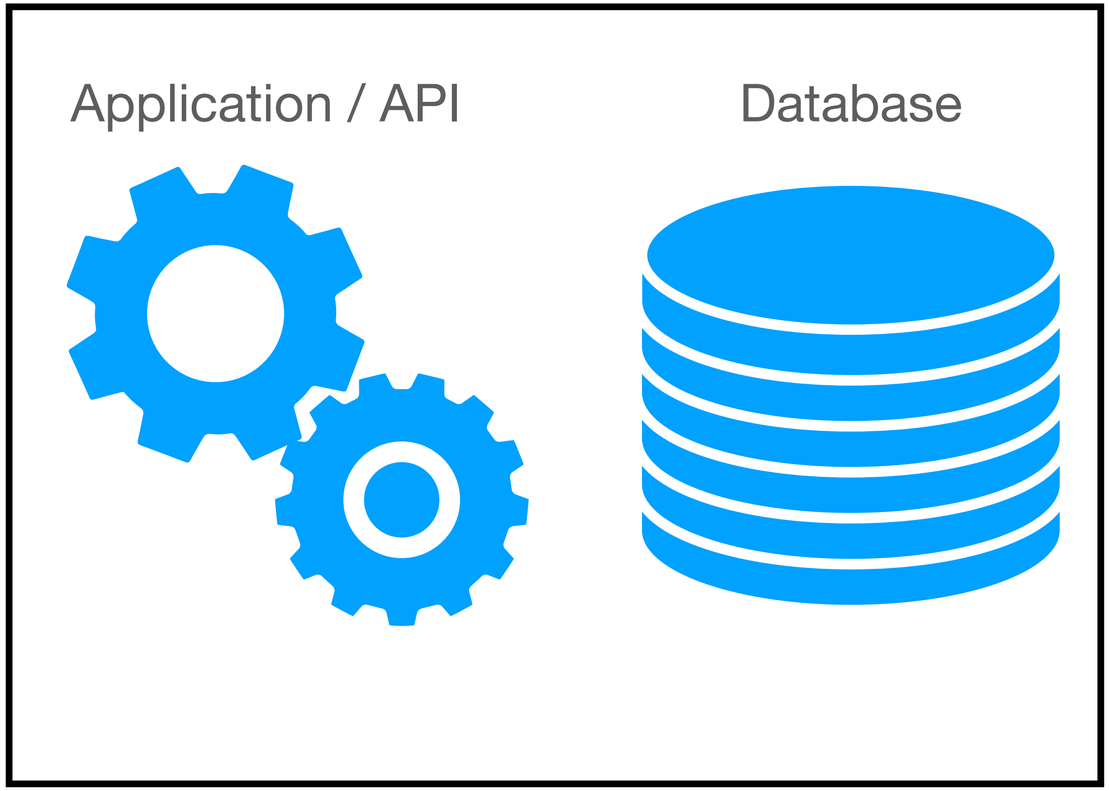
- Definitions & concepts
  - Server / Client
  - HTTP
  - API
  - node.js runtime environment
- Building an API with express.js

Server/Client relationship



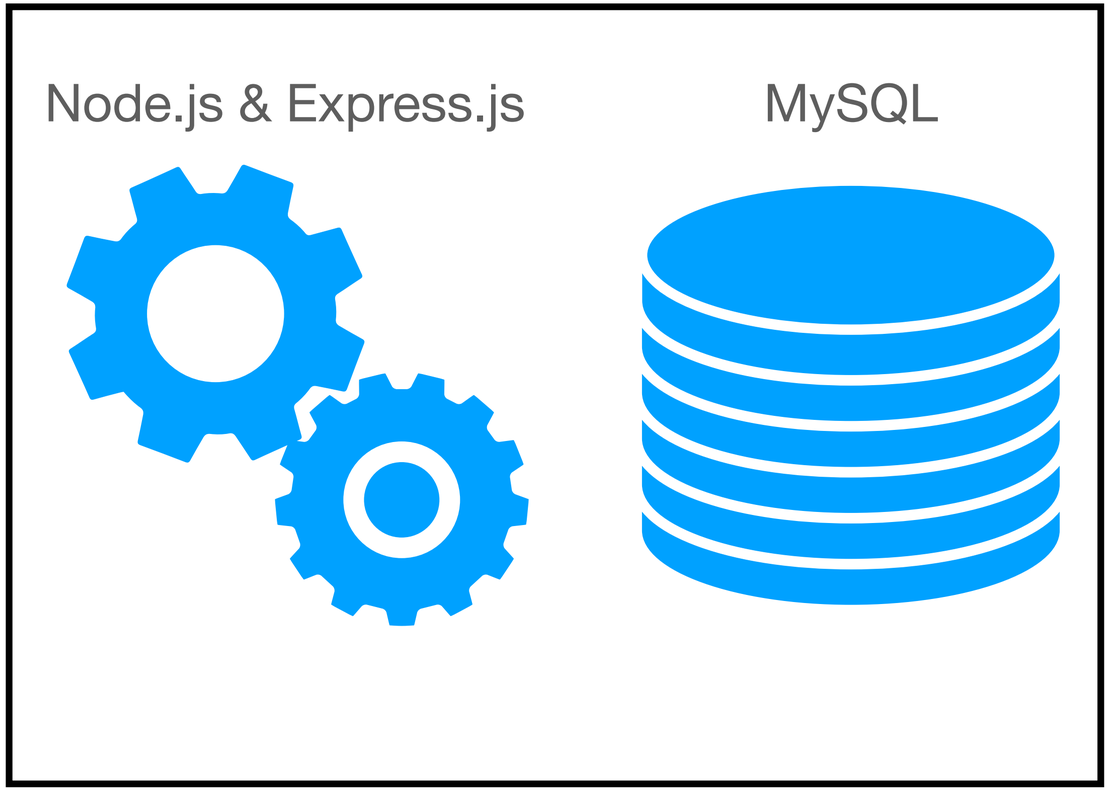
MySQL Database server

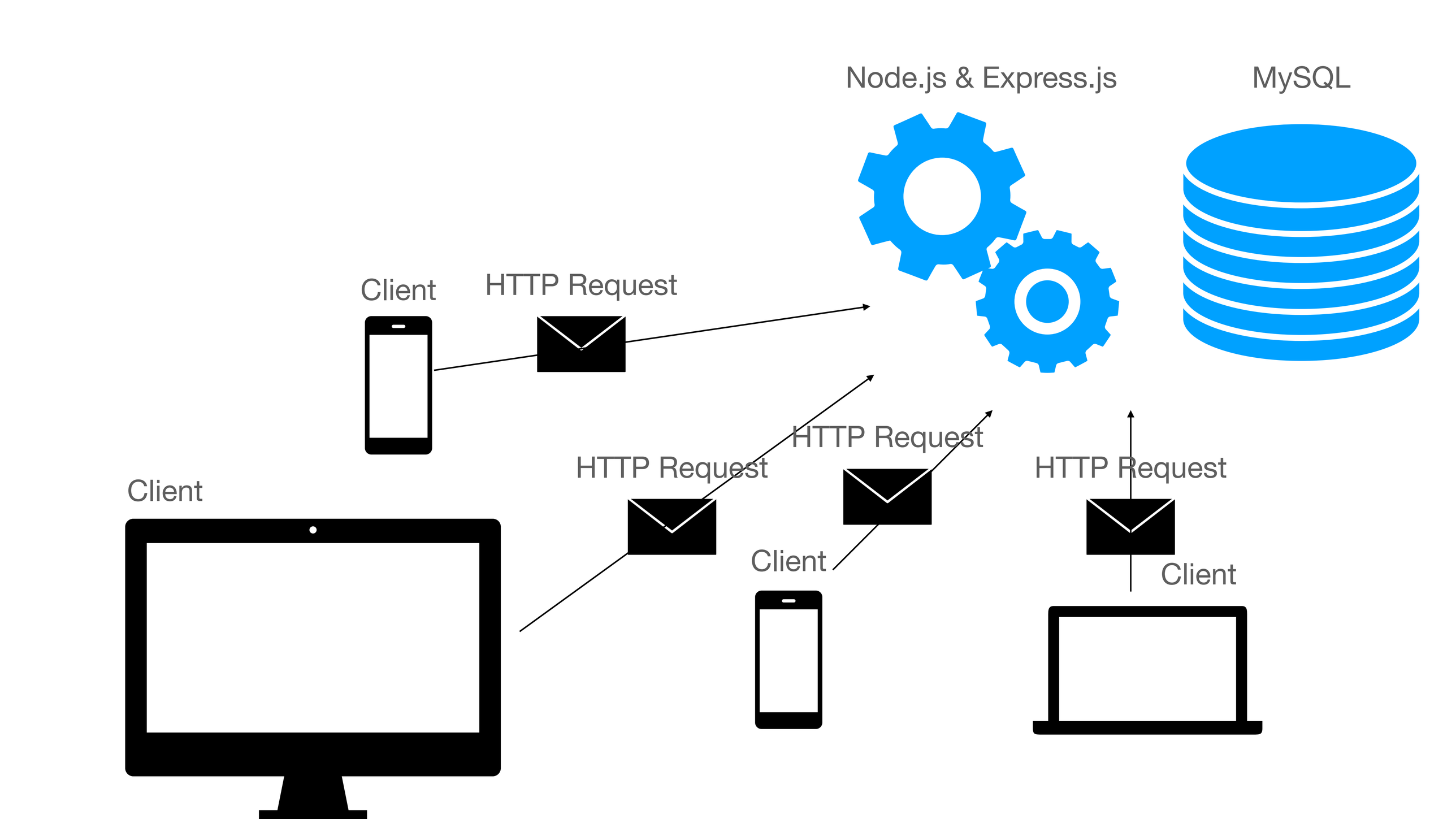
Server

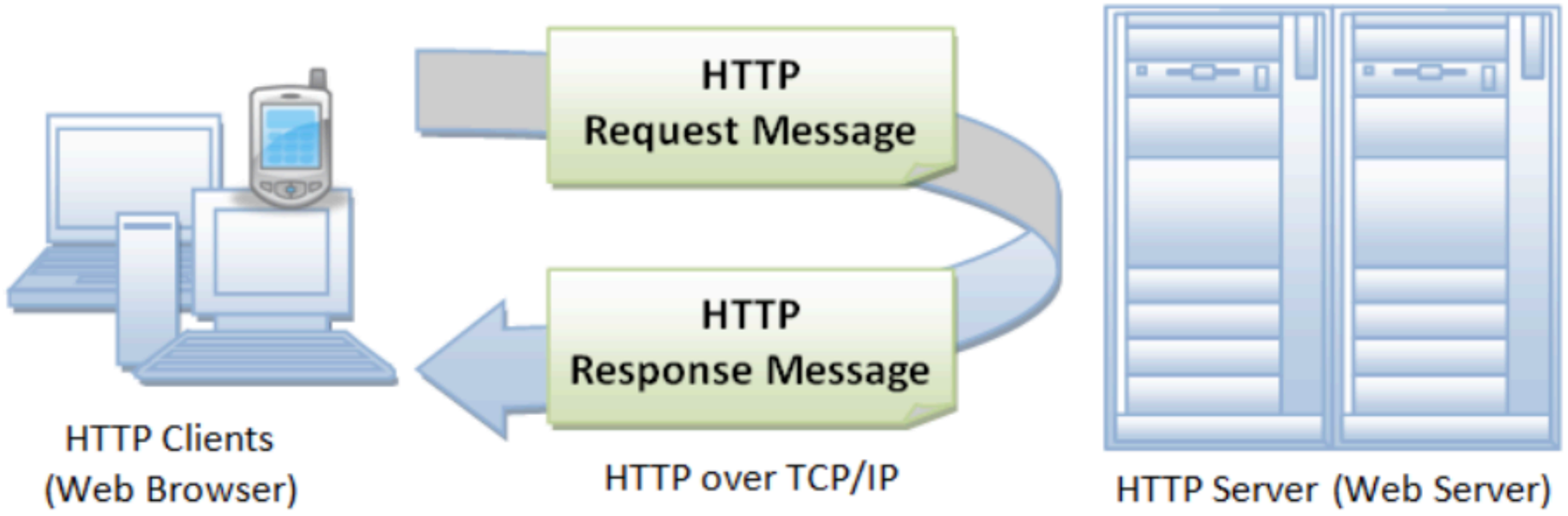


MySQL Database server with Node.js application

Server




Node.js & Express.js
MySQL
Client
HTTP Request
Client
HTTP Request
HTTP Request
Client
HTTP Request
Client

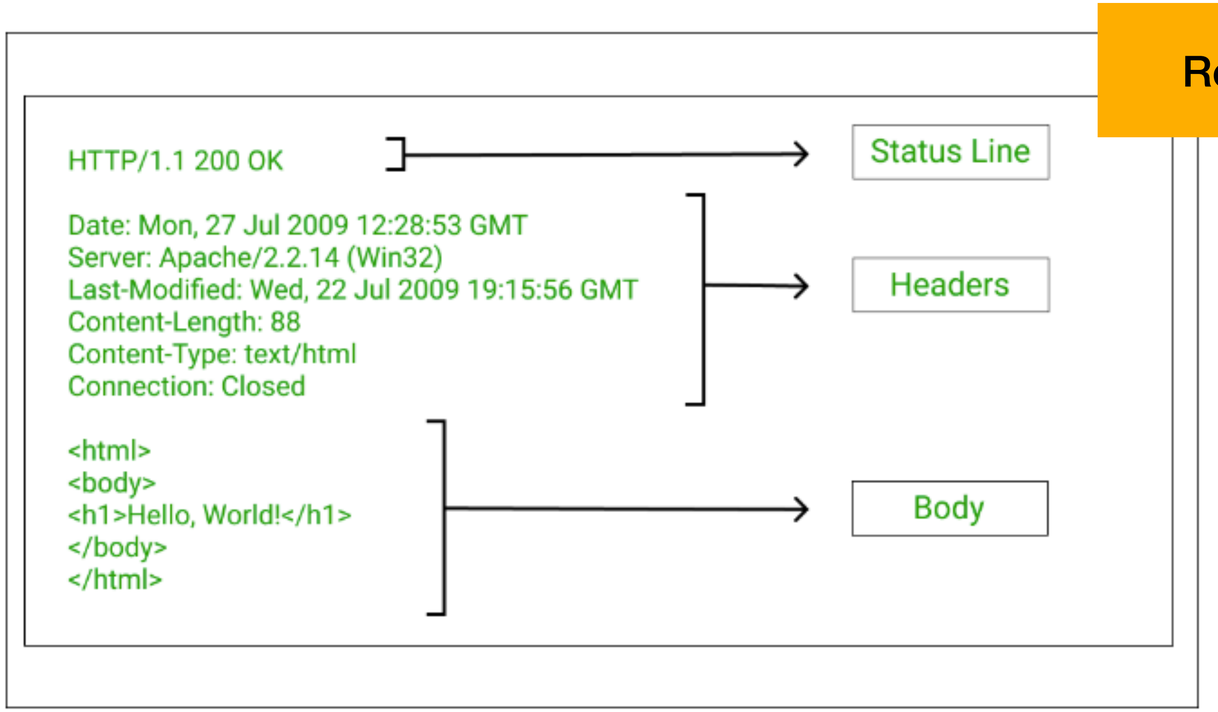



```
GET /docs/index.html HTTP/1.1
Host: www.nowhere123.com
Accept: image/gif, image/jpeg, */*
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)
(blank line)
```

Request

```
GET /docs/index.html HTTP/1.1
Host: www.nowhere123.com
Accept: image/gif, image/jpeg, */*
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)
(blank line)
```

Response



Anatomy of HTTP Response

Browser example

https://javascript.info/

Insomnia example

HTTP STATUS CODES

2xx Success

- 200 Success / OK

3xx Redirection

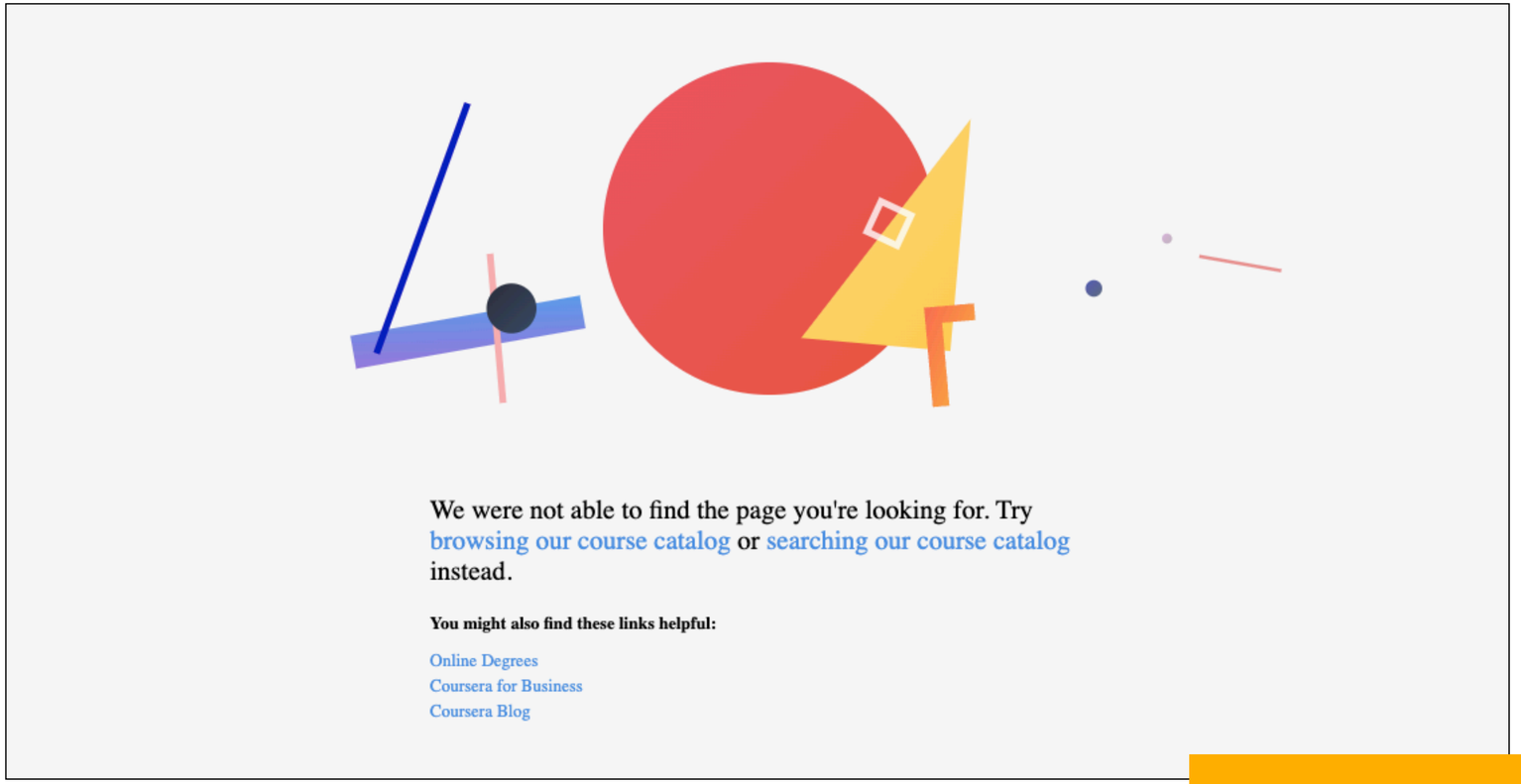
- 301 Permanent Redirect
- 302 Temporary Redirect
- 304 Not Modified

4xx Client Error

- 401 Unauthorized Error
- 403 Forbidden
- 404 Not Found
- 405 Method Not Allowed

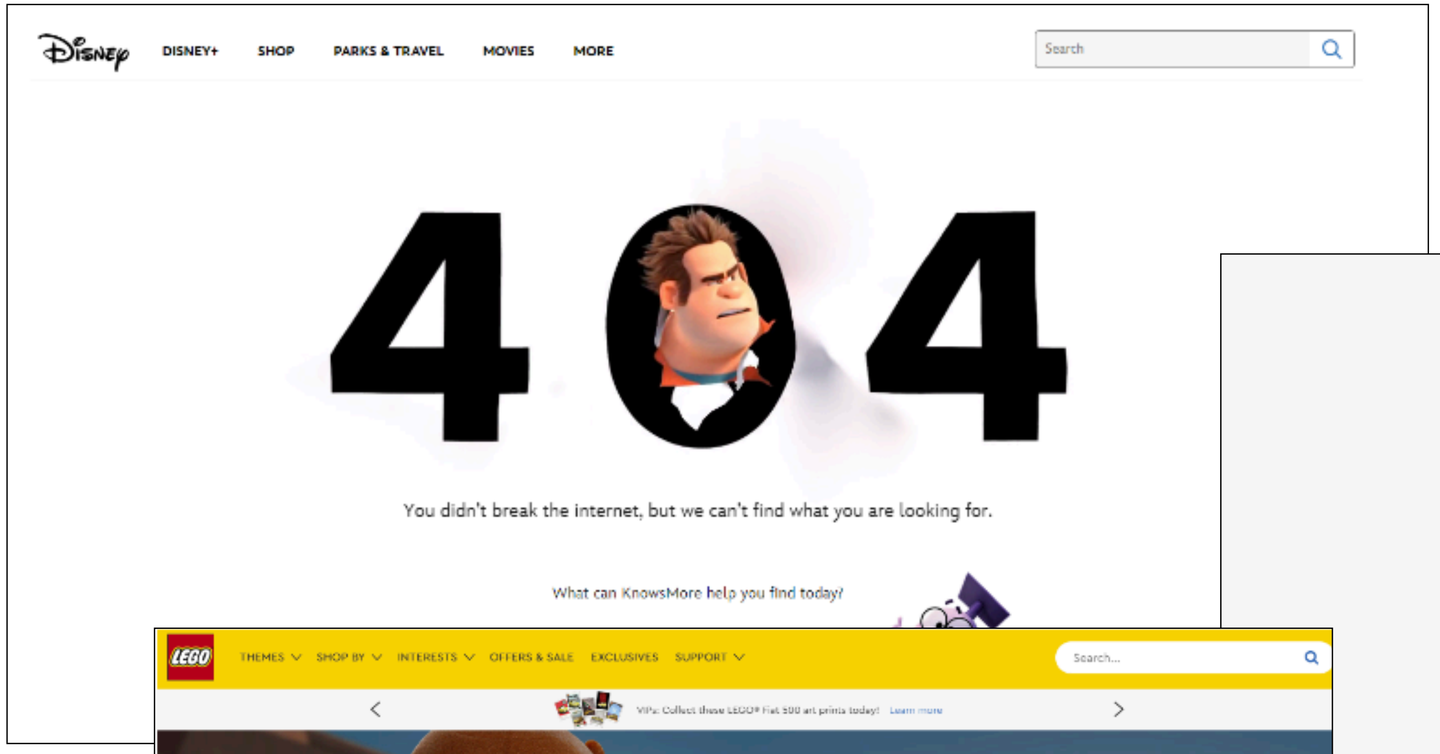
5xx Server Error

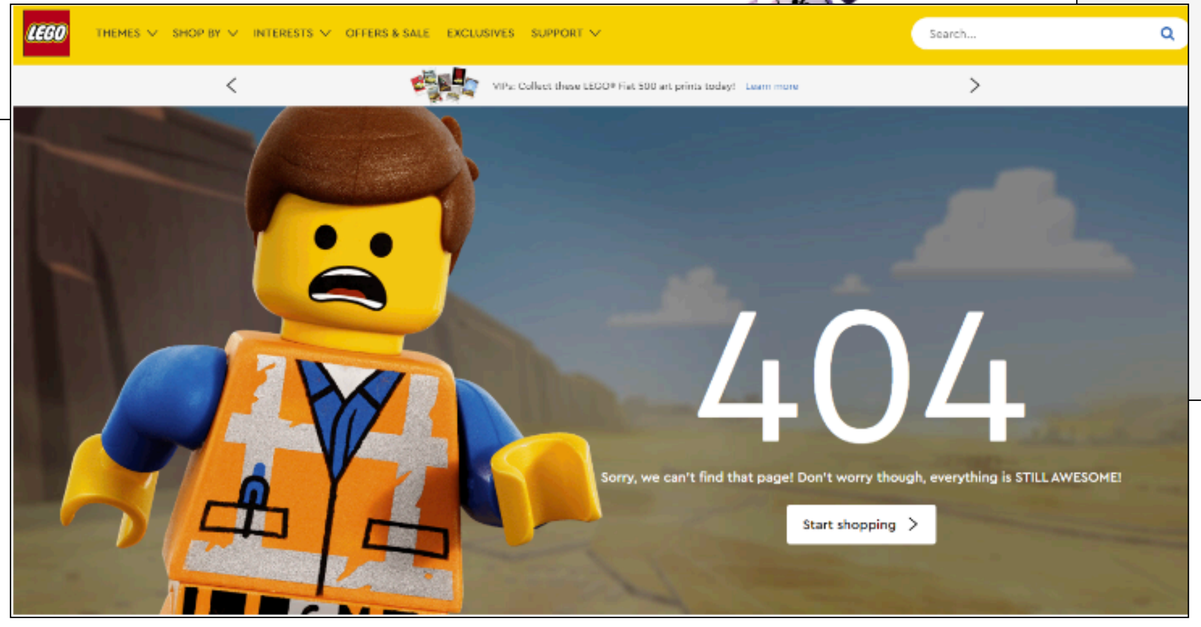
- 501 Not Implemented
- 502 Bad Gateway
- 503 Service Unavailable
- 504 Gateway Timeout

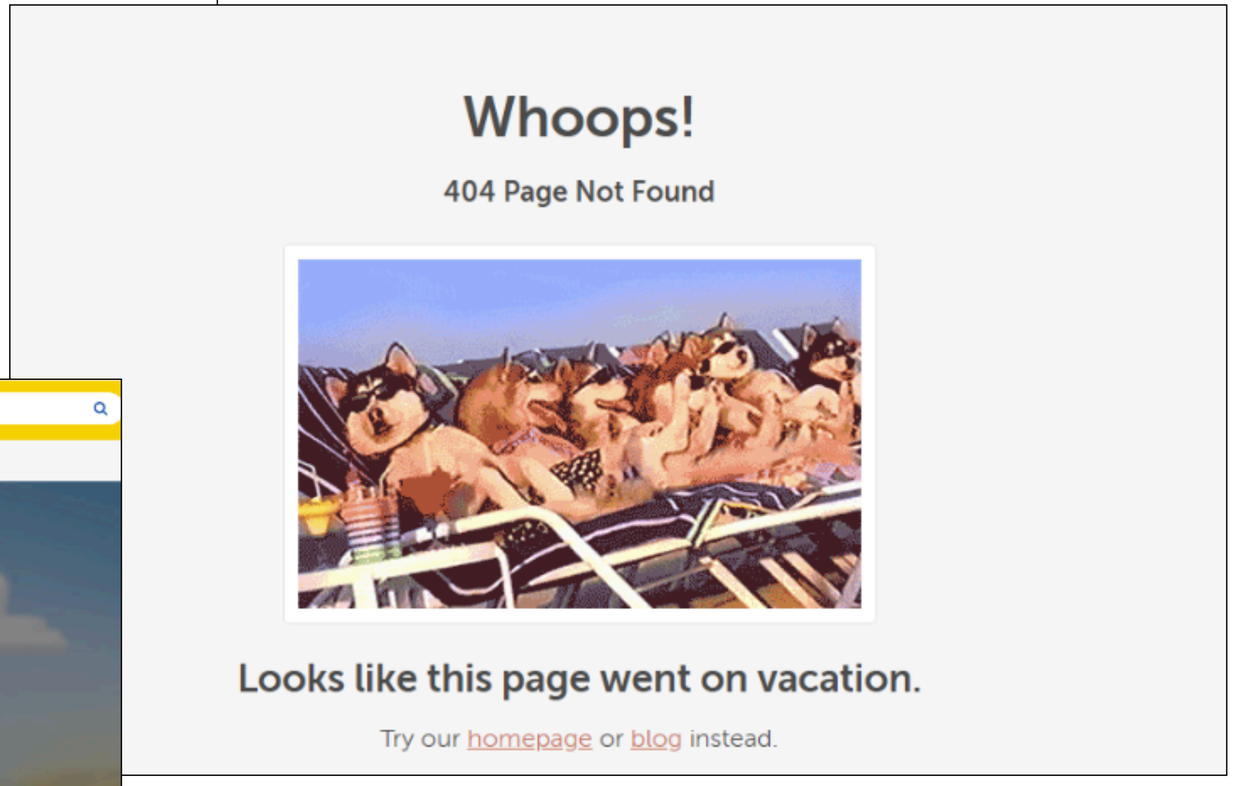


404 Response Body

https://www.coursera.org/


DISNEY+
SHOP
PARKS & TRAVEL
MOVIES
MORE
Search
404
You didn't break the internet, but we can't find what you are looking for.
What can KnowsMore help you find today?


THEMES
SHOP BY
INTERESTS
OFFERS & SALE
EXCLUSIVES
SUPPORT
Search...
404
Sorry, we can't find that page! Don't worry though, everything is STILL AWESOME!
Start shopping

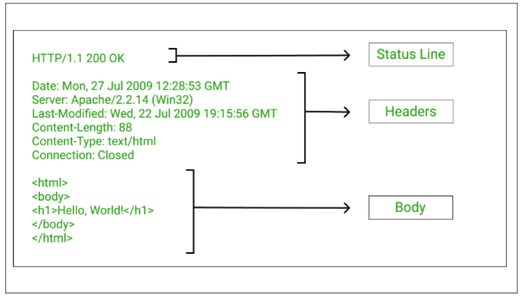

Whoops!
404 Page Not Found
Looks like this page went on vacation.
Try our homepage or blog instead.

```
GET /docs/index.html HTTP/1.1
Host: www.nowhere123.com
Accept: image/gif, image/jpeg, */*
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)
(blank line)
```

HTTP Clients
(Web Browser)

HTTP over TCP/IP

HTTP Server (Web Server)



Anatomy of HTTP Response

```
<!DOCTYPE html>
<html>
<body>

<h1>My First Heading</h1>

<p>My first paragraph.</p>

</body>
</html>
```

Responding with data / API

API

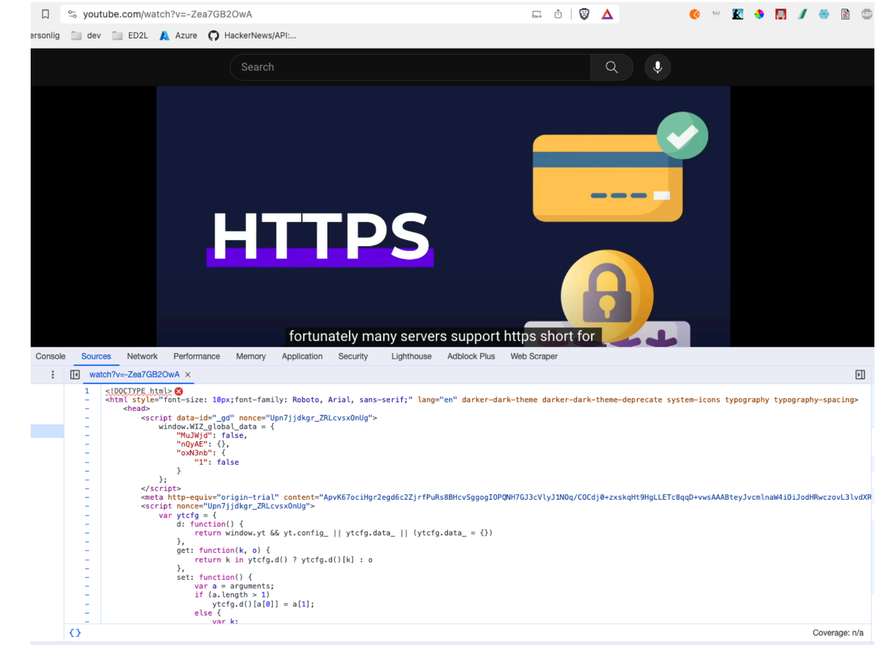
Design pattern

- Abstract Programming Interface
- De facto standard for sharing/distributing data online
- Multiple applications/clients can interact with the same data
- Usually connected to a datasource
- Will typically follow a design pattern known as REST / RESTful

So far…

How we have been building software

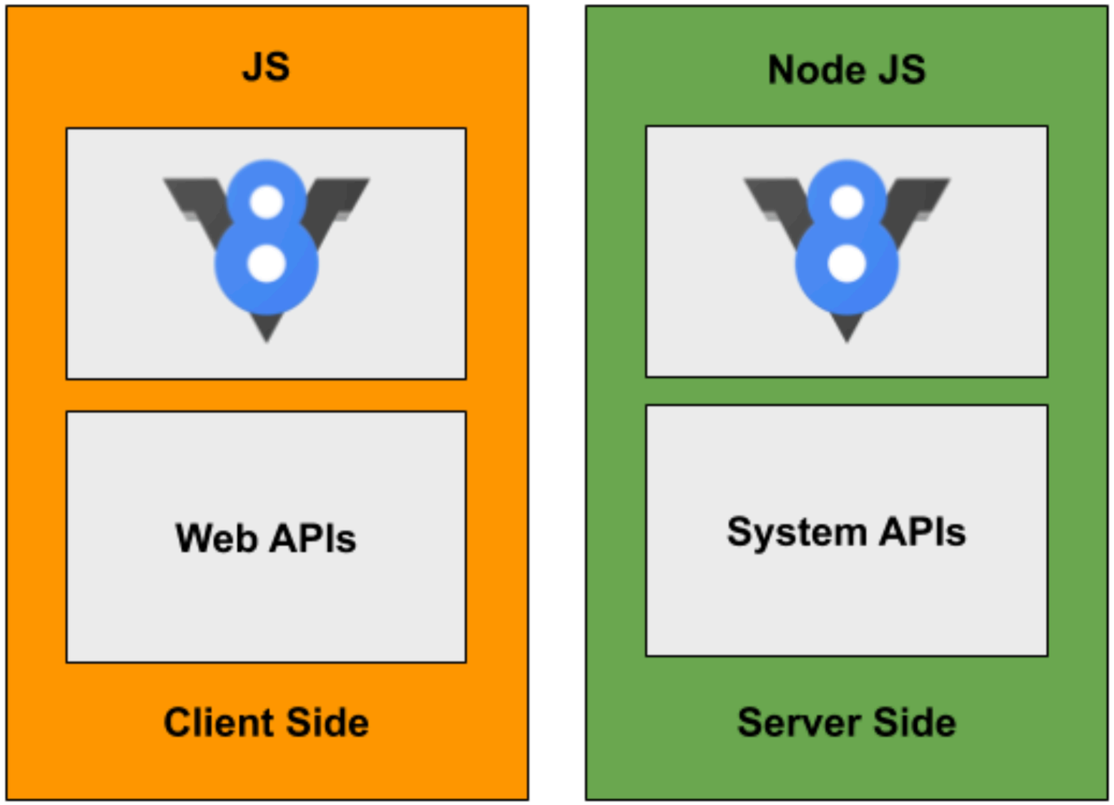
- Using the browser as runtime system
- Javascript has been running “inside” the browser
- The browsers runtime system has been executing javascript connected to a HTML document
- Client side code -> Executed in the browser and by the browser
- Client side javascript can be used to connect to a dom e.g. document.querySelector(“#main”);



Introducing node.js runtime system

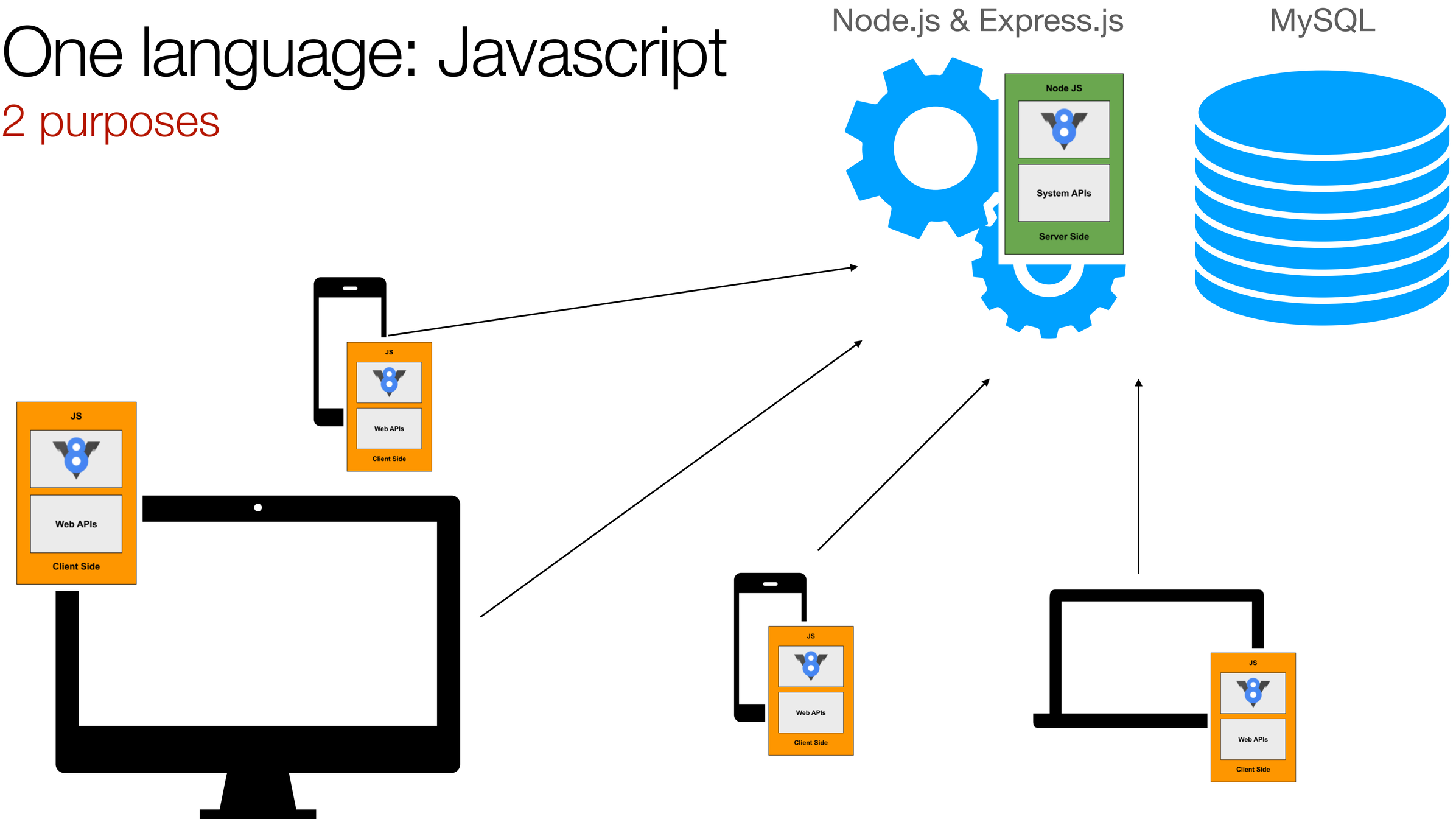
One language: Javascript

2 runtime environments



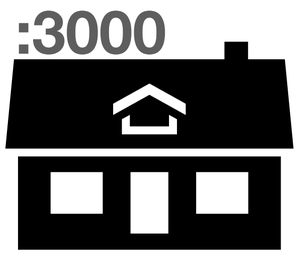
One language: Javascript

2 purposes

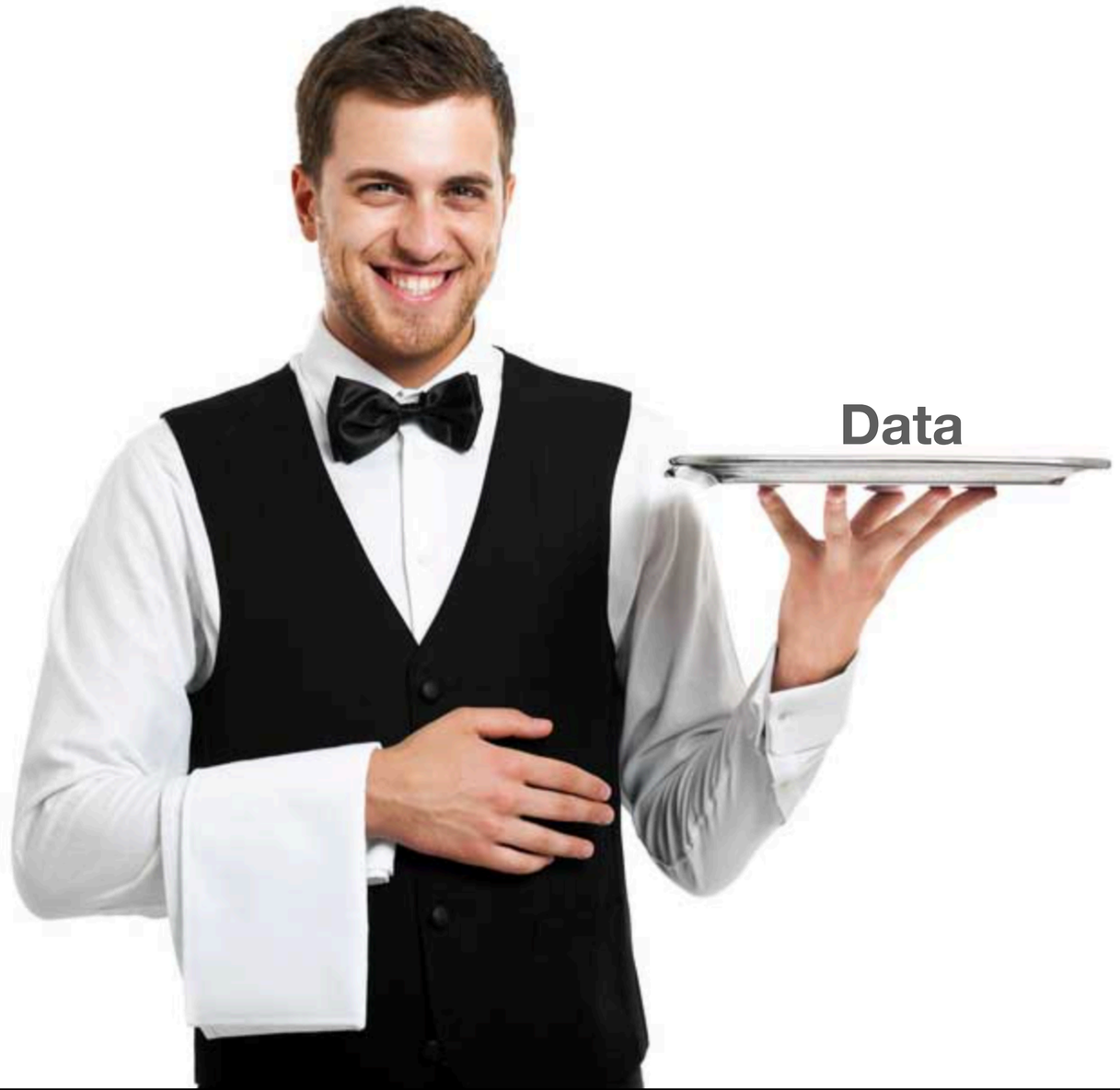


```
GET /docs/index.html HTTP/1.1
Host: www.nowhere123.com
Accept: image/gif, image/jpeg, */*
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)
(blank line)
```

Get *me some data!*



Server



Building a basic web server with node

Node.js examples

No interaction with the DOM

Because the code is server-side
(no HTML document / browser runtime)

Capability: Node.js

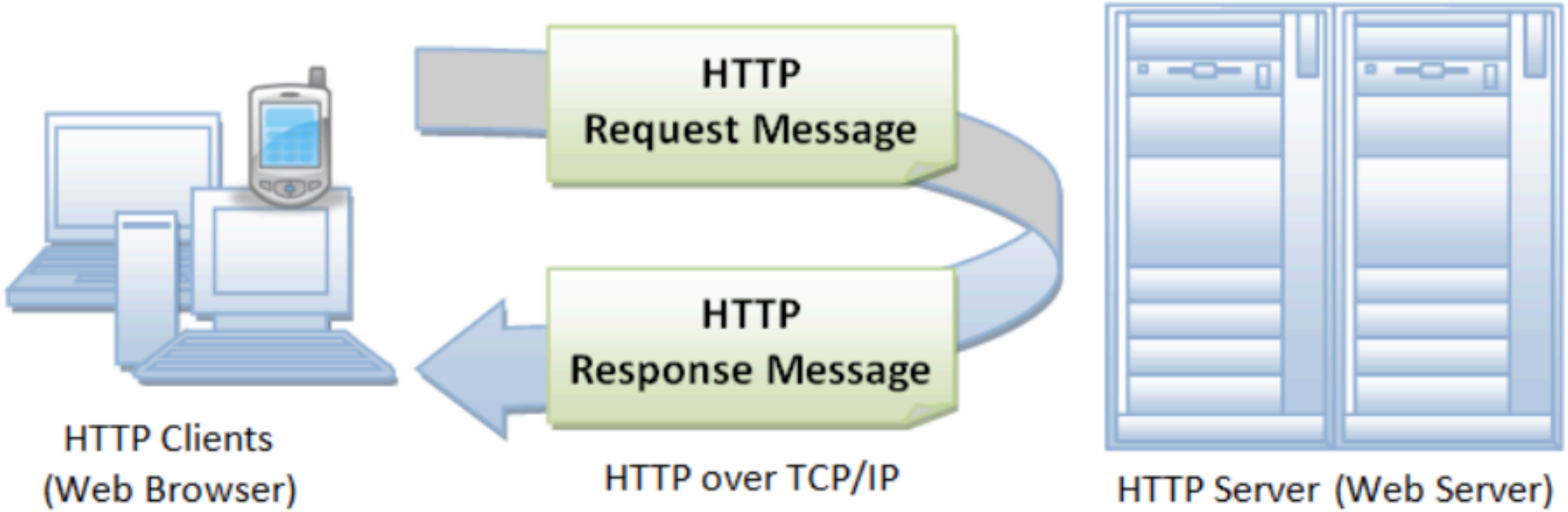
Interaction with a datasource such as a mysql database

Exercises 1

Introducing express.js

```
$ npm install express --save
```

Basic express example



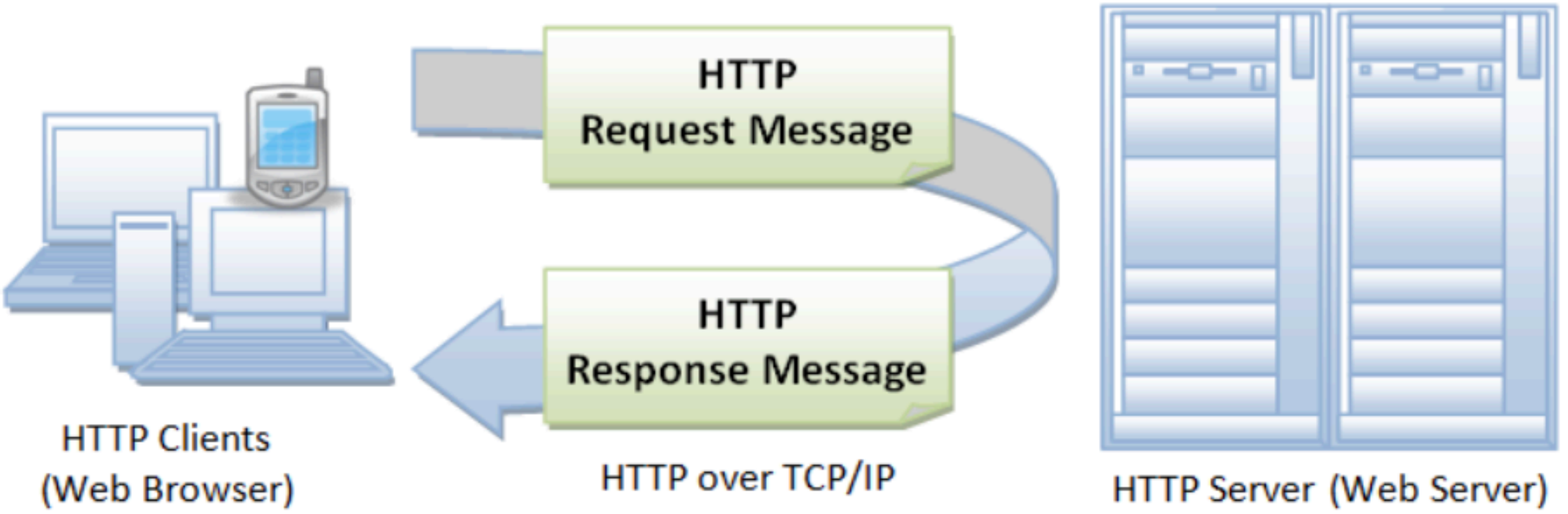

```
app.get("/",(req, res) =>{

});
```

Request parameter

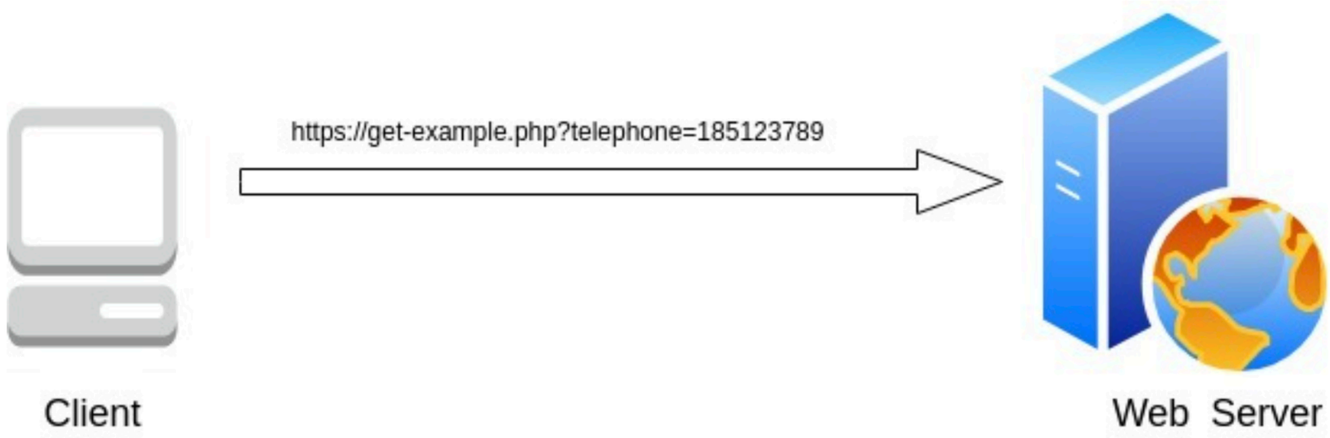
https://politiken.dk/kultur/art9625582/

https://politiken.dk/kultur/art9625582/



2. The GET Method

GET is used to request data from a specified resource. It can retrieve any visible data to a client, such as HTML documents, images, and videos:

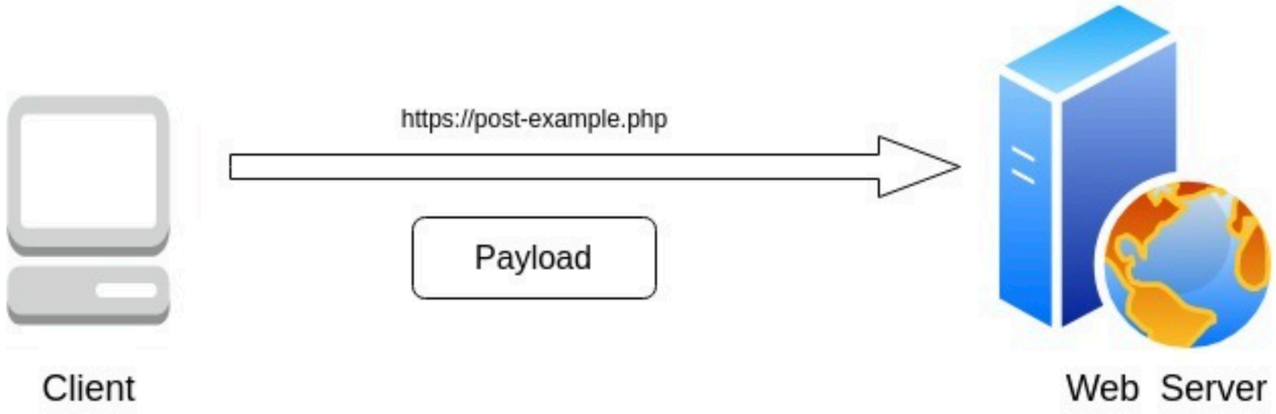


To send a GET request, a client needs to specify the URL of the resource it wants to retrieve. The request is then sent to the server, which processes the request and sends the requested data back to the client.

https://www.baeldung.com/cs/http-get-vs-post

3. The POST Method

The POST sends data to a server to create or update a resource. For example, it is often used to submit an HTML form to a server:



To send a POST request, a client needs to specify the URL of the resource to which it wants to send data and the data itself. The request is then sent to the server, which processes the request and sends a response back to the client.

The POST method is often used to submit forms or upload files to a server.

https://www.baeldung.com/cs/http-get-vs-post

Exercises 2