LESSON 16

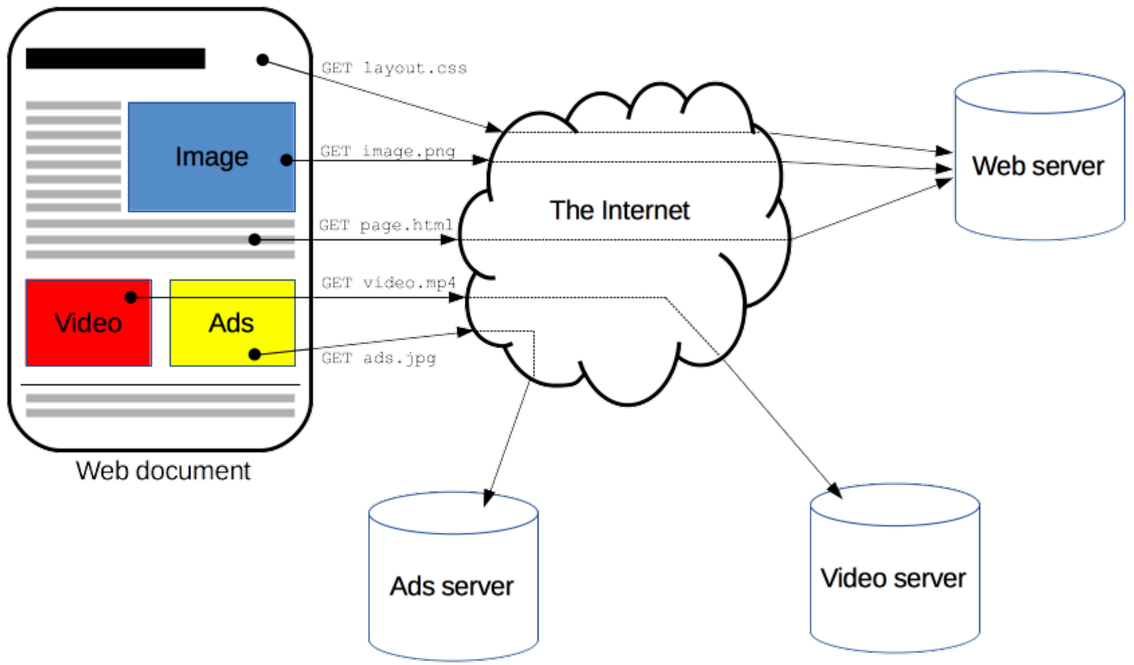
HTTP, AJAX

CLIENT – SERVER ARCHITECTURE

- Client make a request
- DNS - server redirects domain to IP address
- Server listen to requests and delegate it to corresponding port listener
- Server app process request
- Get data from the DB
- Make response to client
- Client’s browser receives data

HTTP VERSION 2.0

- Hypertext Transfer Protocol



https://medium.com/platform-engineer/evolution-of-http-69cfe6531ba0#:~:text=HTTP%20has%20four%20versions%20%E2%80%94%20HTTP,1.1%2C%20and%20HTTP%2F2.0.

HTTP VS HTTPS

- HTTP URL in your browser's address bar is `http://` and the HTTPS URL is `https://`.
- HTTP is unsecured while HTTPS is secured.
- HTTP sends data over port 80 while HTTPS uses port 443.
- HTTP operates at application layer, while HTTPS operates at transport layer.
- No SSL certificates are required for HTTP, with HTTPS it is required that you have an SSL certificate and it is signed by a CA.
- HTTP doesn't require domain validation, where as HTTPS requires at least domain validation and certain certificates even require legal document validation.
- No encryption in HTTP, with HTTPS the data is encrypted before sending.

METHODS

- Request
- Response
- Method
- Status code

HTTP REQUEST

```
GET /en-US/docs/Glossary/Simple_header HTTP/1.1
Host: developer.mozilla.org
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.9; rv:50.0) Gecko/20100101 Firefox/50.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer: https://developer.mozilla.org/en-US/docs/Glossary/Simple_header
```

RESPONSE

```
HTTP/1.1 200 OK
Connection: Keep-Alive
Content-Encoding: gzip
Content-Type: text/html; charset=utf-8
Date: Wed, 20 Jul 2016 10:55:30 GMT
Etag: "547fa7e369ef56031dd3bff2ace9fc0832eb251a"
Keep-Alive: timeout=5, max=1000
Last-Modified: Tue, 19 Jul 2016 00:59:33 GMT
Server: Apache
Transfer-Encoding: chunked
Vary: Cookie, Accept-Encoding
```

METHODS

HTTP Verb	CRUD	Entire Collection (e.g. /customers)	Specific Item (e.g. /customers/{id})
POST	Create	201 (Created), 'Location' header with link to /customers/{id} containing new ID.	404 (Not Found), 409 (Conflict) if resource already exists..
GET	Read	200 (OK), list of customers. Use pagination, sorting and filtering to navigate big lists.	200 (OK), single customer. 404 (Not Found), if ID not found or invalid.
PUT	Update/Replace	405 (Method Not Allowed), unless you want to update/replace every resource in the entire collection.	200 (OK) or 204 (No Content). 404 (Not Found), if ID not found or invalid.
PATCH	Update/Modify	405 (Method Not Allowed), unless you want to modify the collection itself.	200 (OK) or 204 (No Content). 404 (Not Found), if ID not found or invalid.
DELETE	Delete	405 (Method Not Allowed), unless you want to delete the whole collection—not often desirable.	200 (OK). 404 (Not Found), if ID not found or invalid.

STATUS CODES

- 1xx - информационные
- 2xx - успех
- 3xx - переадресация
- 4xx - ошибка на стороне клиента
- 5xx - ошибка на стороне сервера

STATUS CODES

- HTTP Status Code 200 - OK. ...
- HTTP Status Code 301 - Permanent Redirect. ...
- HTTP Status Code 302 - Temporary Redirect. ...
- HTTP Status Code 404 - Not Found. ...
- HTTP Status Code 500 - Internal Server Error. ...
- HTTP Status Code 503 - Service Unavailable.

- wiki

RESTFUL API (REST API)

- A RESTful API is an architectural style for an application program interface (API) that uses HTTP requests to access and use data. That data can be used to GET, PUT, POST and DELETE data types, which refers to the reading, updating, creating and deleting of operations concerning resources.

SERVER DATA PUSH

- Ajax Polling
- Long Polling
- WebSockets
- Server Sent Events