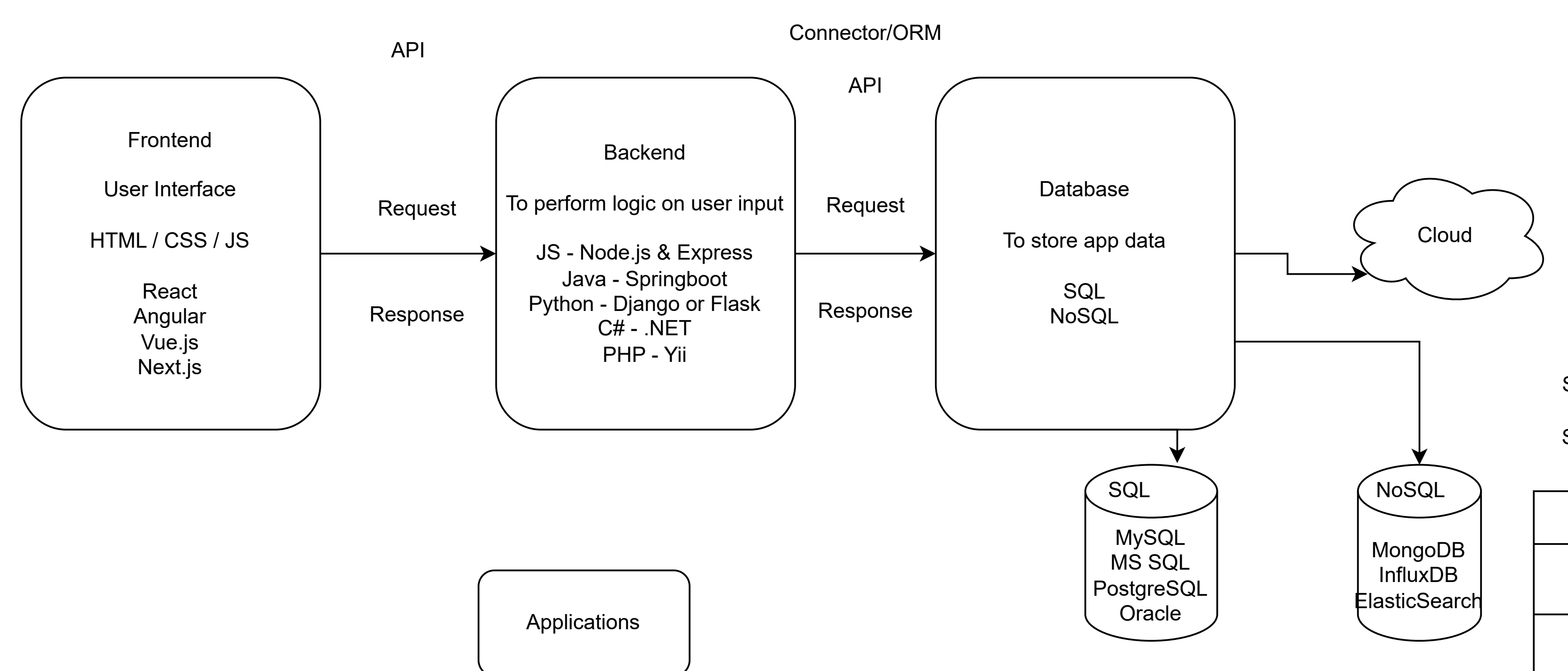


SQL -> Table like structured

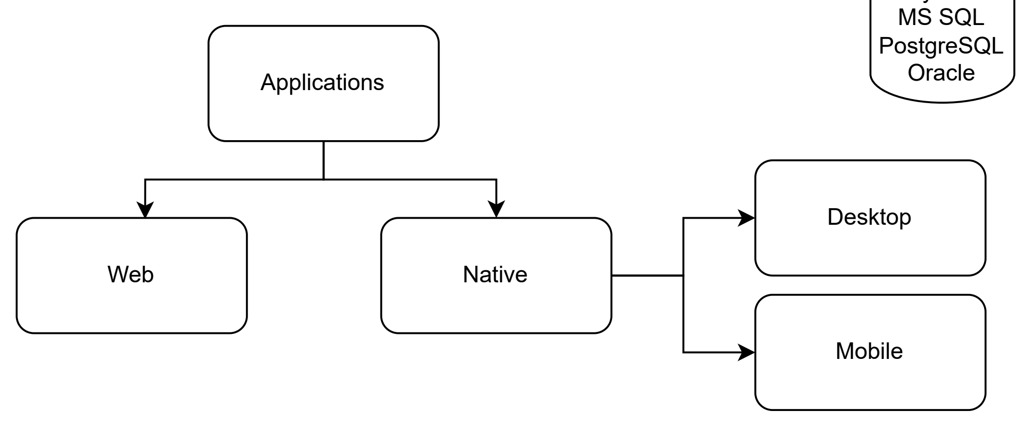
Structured Query Language

User		
id	name	email
1	John	john@gı
2	Priya	priya@g

Key-Value paired data

NoSQL

```
{
id: 1,
name: 'John',
email: 'john@gmail.com
},
{
id:2,
name: 'Priya',
email: 'priya@gmail.com'
}
```

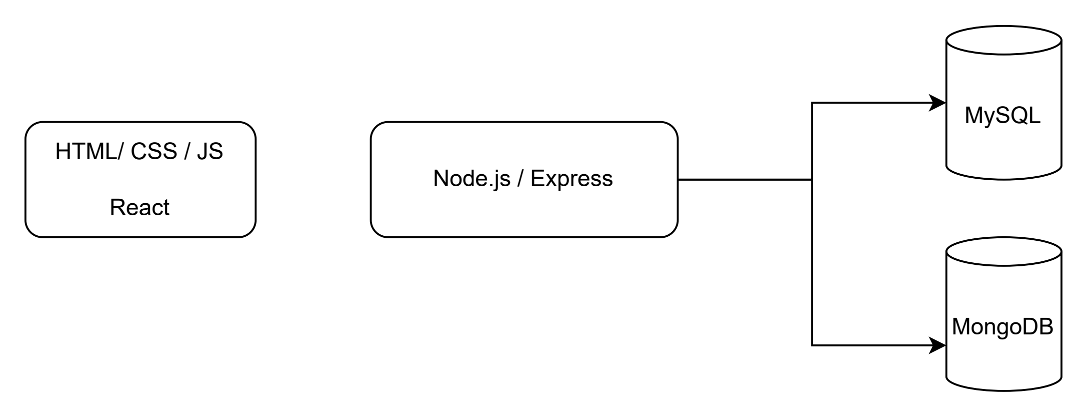


web applications dont need installation

native apps require installing.

web apps use browser resource to run

native apps use OS



JS
(Language)

Node.js
(Environment)

Express
(Library)

Introduction to web Browser Wars

DOM tree CSSOM tree,
Browser internals - HTML parser,
CSS parser, JavaScript V8 engine,
internals

IP – MAC address – Ports &

Evolution of HTTP,
HTTP Methods

How the Server looks at the URL Request & Response cycle

Browsers
(platforms to run web based applications
sends request to the server, gets back the response
from the server)

Google Chrome
Mozilla Firefox
Microsoft Edge
Safari

Chrome

Google's V8 Engine

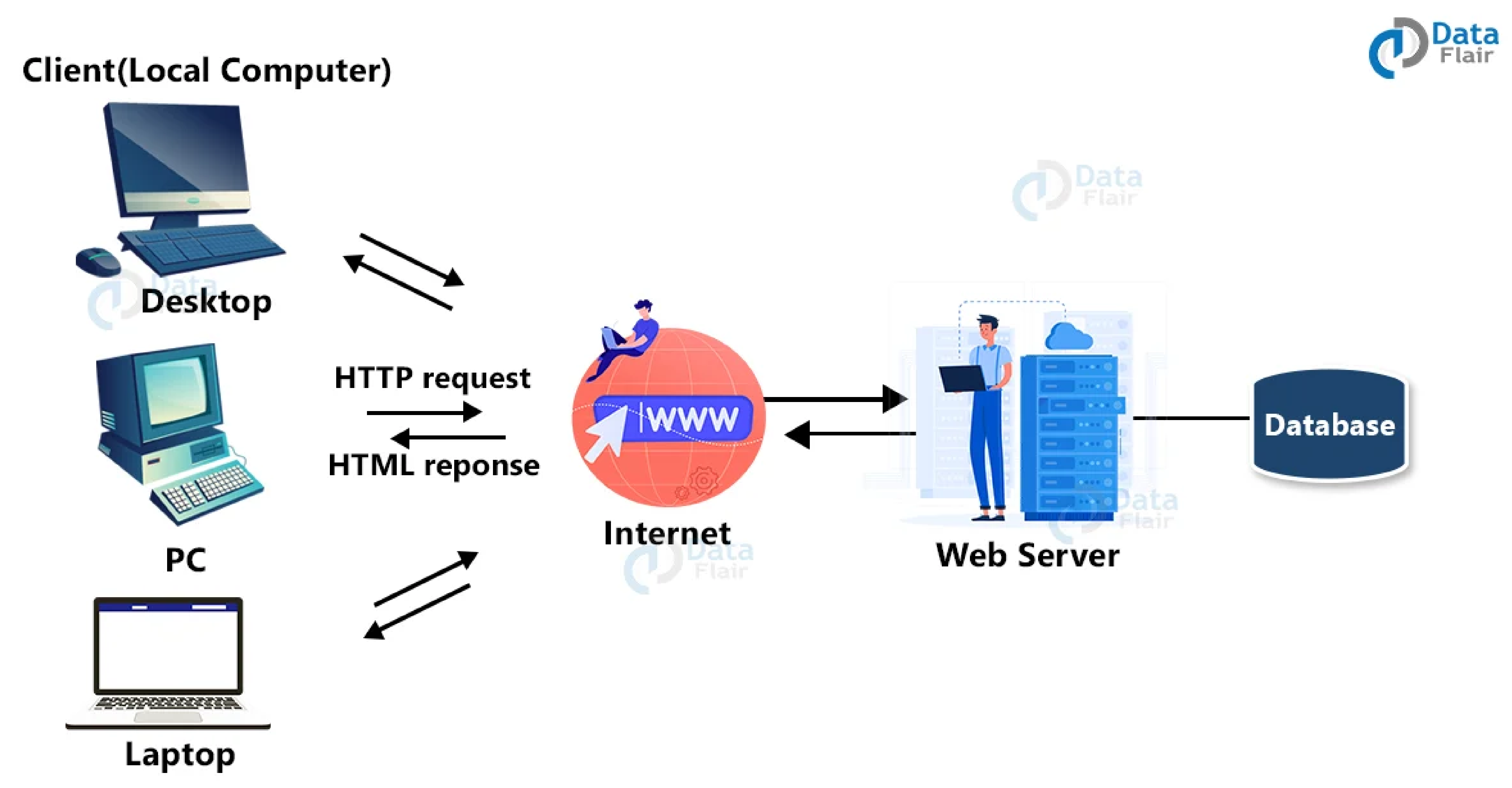
Chromium

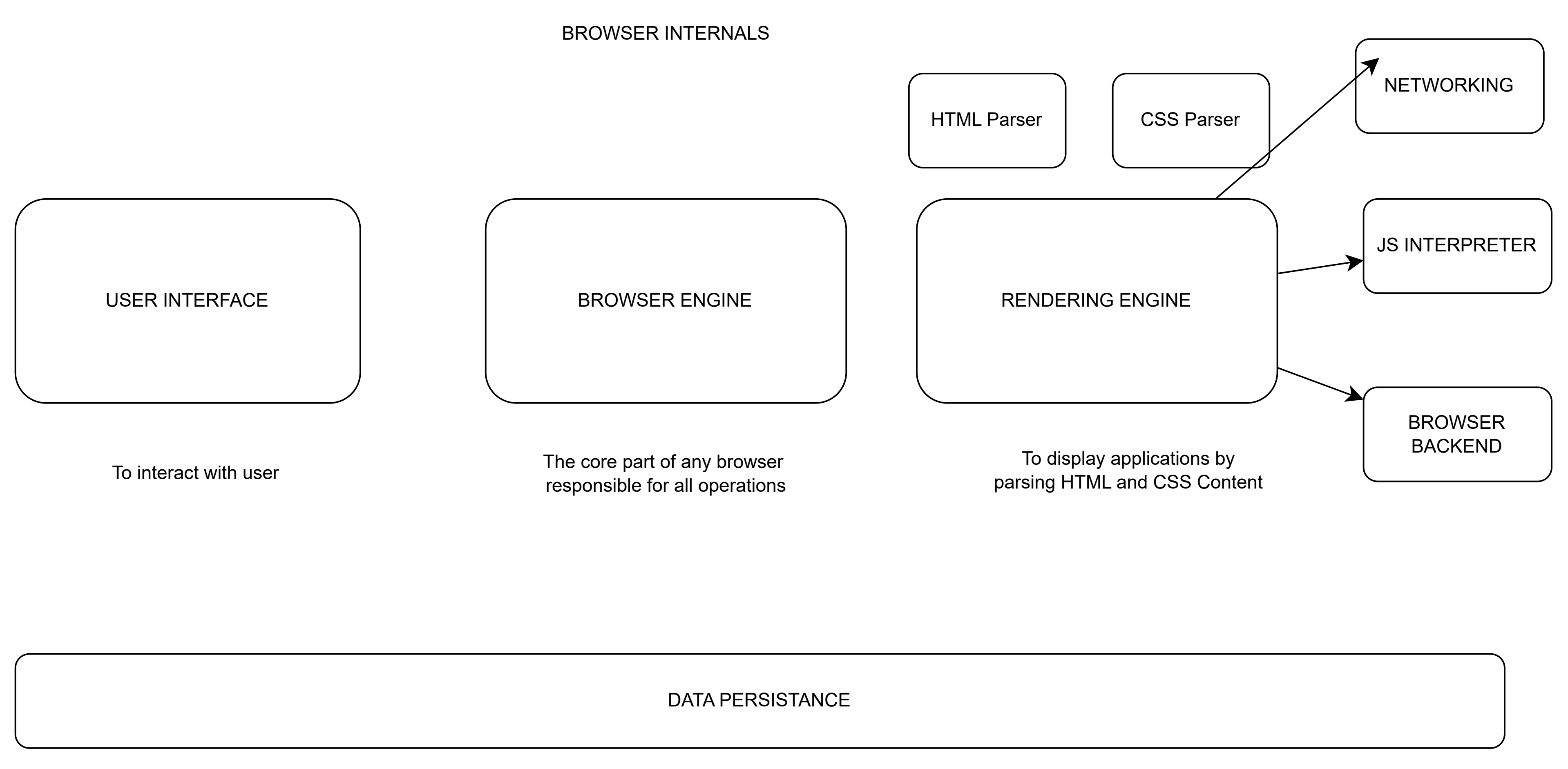
Firefox

SpiderMonkey

Safari

WebKit





IP ADDRESS

IPv4/ IPv6

DNS

OSI Layers

NETWORK PROTOCOLS

2^32 2^128

12.30.45.50 -> IP Address

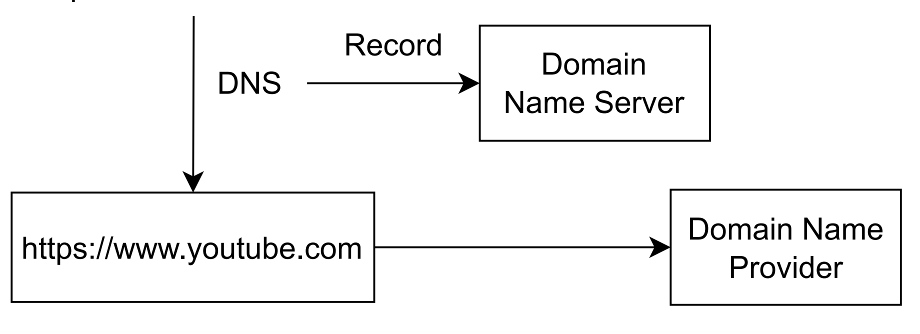
127.0.0.1
localhost

2^16 ports - 0 - 65535

1000 -> App A
1001 -> App B
1002 -> App C

http://localhost:3000

http://12.30.45.50:3000

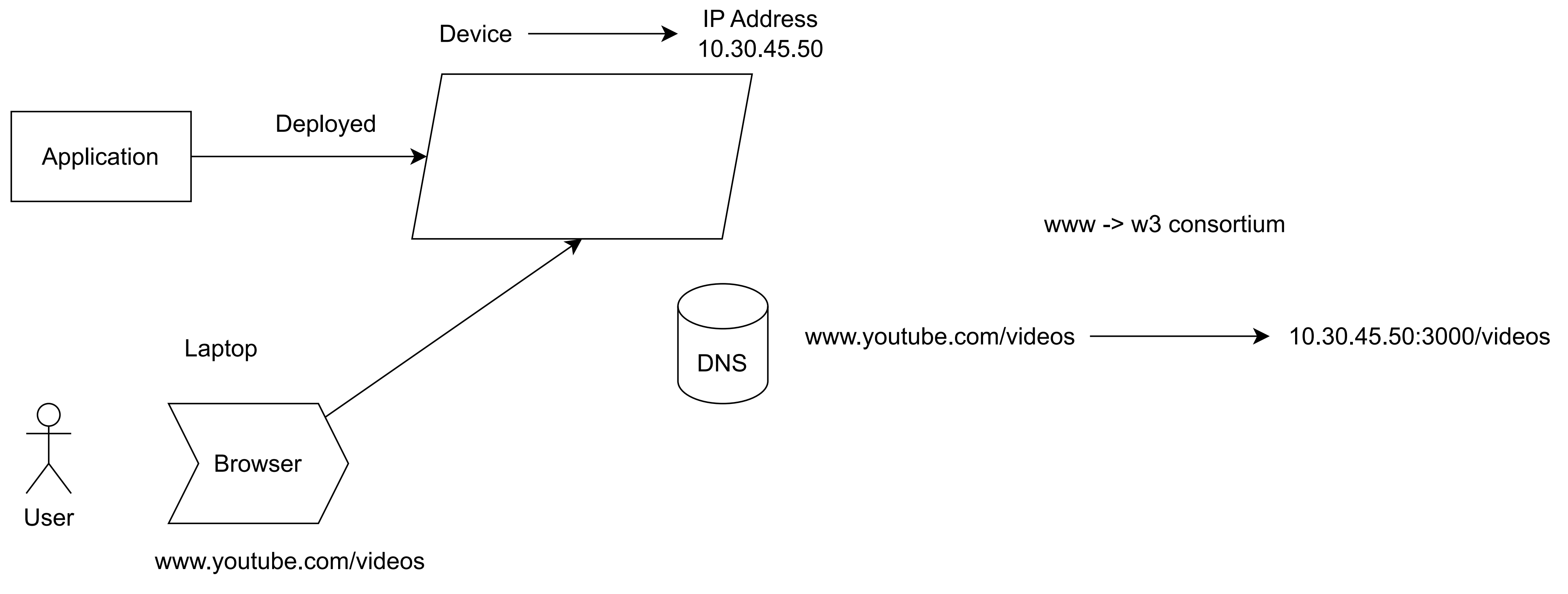


How server looks at the url

Uniform Resource Locator

https://www.example.com/products

www.hotstar.com/sports/cricket



CODEKATA

```
// JS - Javascript - Programming Language

// Scripting Language -> Javascript
// Compiled Language -> C++, Java etc

// Blocking - Synchronous
// Non-Blocking- Asynchronous

// Javascript
//  - JS is single threaded. (it will execute one execution at a time.)
//  - Non-Blocking I/O (Asynchronous) operations, because of a mechanism; (Event Loop)
//  - Used for Client and Server side programming.


// Threads - is used to carry out the process/execution one at a time.


//  - Datatype
//  - Variable
//  - Operators

//  - Looping Statements
//  - Conditional Statements
//  - Functions
//  - Objects and Arrays
//  - Class (OOPS)

//  Object Oriented programming Language

//  Typescript(OOPS) -> parent of Javascript

// - Datatype:

// data -> values or information

// Datatype ->

// Numbers - 1, 100, 50, -5, -10, 0.001, 0.6 (including integers, float, double)
// String - "priya", "john", "chennai", "India", "Cricket", "How are you?", "a", "b"
// Boolean - true,  false

// console.log("Hello")
// console.log(12345)
// console.log(true)

// // typeof - keyword used to find the type of the data

// console.log(typeof "hello");
// console.log(typeof 100);
// console.log(typeof false);

// console.log(typeof -123);
// console.log(typeof "1000");
// console.log(typeof "true")

// console.log(typeof 'Hey');



// Variable - is used to hold some values or data
// var, let, const

// Declaration and Assignment
// var username = 'John';
// let city = 'Bangalore';
// const gender = 'Male';


// console.log(username)
// console.log(city)
// console.log(gender)
// // // Declaration
// var myValue1;

// //Assignment:
// myValue1 = 1000;

// var myValue1 = 1000;



// var - can be redeclared and can be reassigned
// let - cannot be redeclared but can be reassigned
// const - cannot be redeclared and cannot be reassigned.

// var value1 = 'ABC';
// var value3 = 'PQR';
// var value1 = 'XYZ';

// value1 = 'Hello';
// console.log(value1);


// let value2 = '123';
// // let value2 = '456';
// value2 = 'How are you?';
// console.log(value2)

// const bloodGroup = 'O+ve'
// const bloodGroup = 'A-ve'
// bloodGroup = 'B +ve'

// OPERATORS:

// Arithmetic = +, -, *, /, %

// console.log(5 + 2)
// console.log(5 - 2);
// console.log(5 * 2);
// console.log(5 / 2);
// console.log(5 % 2);
// Logical
// AND (&&) - true and true = true, if any one input is false, the output will be false
// OR (||) - if anyone is true , the output will be true.if  both are false, then the output will be false.
// NOT (!) - if it is true, output will false. if it is false, output will be true.
// NAND - true and true = false, if any one input is false, the output will be true
// NOR - if anyone is true , the output will be false.if  both are false, then the output will be true.

// console.log(true && true); // true
// console.log(true || true) // true
// console.log(true || false); // true
// console.log(false && true); // false
// console.log(!true); // false

// console.log(!(true || true))  // false
// console.log(!(true && false)); // true


// Comparison
// >
// <
// >=
// <=
// ==
// !=

// console.log(5 > 2) // true
// console.log(5 == 2); // false
// console.log(5 <= 5); // true
// console.log(5 != 4);  // true
// console.log(5 < 5); // false

// == -> checks only for the value
// === -> checks for the value and its datatype

// console.log(5 == '5'); // true
// console.log(5 === '5'); // false
// console.log(5 === 5); // true

// console.log( 5 != '5'); // false
// console.log(5 !== '5'); // true

// String
// console.log(5 + 5); //10
// console.log('5' + '5'); //55 -> concatenation -> appending;


// Conditonal or Ternary OPERATORS

// ? :
// condition ? statement1 : statement2

// 5 > 20 ? console.log('5 is greater') : console.log('5 is not greater');


let value1 = '1000';
console.log(typeof value1); // String


let value2 = parseInt(value1);

console.log(typeof value2)

let value3 = 'abc';

let value4 = parseInt(value3)

console.log(value4)

let value5 = 1234.44;
let value6 = parseInt(value5);

console.log(value5)

console.log(value5.toFixed(3)) // number of digits from decimal point

console.log(value5.toPrecision(9)) // number of digits from the start.



// Object / Arrays

// arrays -> []
// object -> {}

// arrays

let fruits = ['apple', 'mango', 'orange', 'watermelon', 'pineapple']; // collection of strings
let marks = [80, 90, 95, 92, 78]; // collection of number
var myArray = ['Blue', 1200, true, -10]

console.log(fruits);
console.log(marks);

// array index starts with 0 and goes upto (length - 1)

console.log(fruits[3]); //watermelon
console.log(marks[0]); // 80

console.log(marks[5]) //undefined

console.log(fruits.length)

fruits[3] = 'lemon';

fruits[8] = 'jackfruit'

console.log(fruits)
console.log(fruits[7]); //undefined

console.log(fruits.length)


// Object

// {
//     key: value
// }

let userDetails = {
```

```
name: 'John',
city: 'Bangalore',
email: 'john@gmail.com',
mobileNumber: '+91 1234567890'
}

console.log(userDetails)

console.log(userDetails['email'])

console.log(userDetails.mobileNumber)

userDetails['city'] = 'Mumbai';
userDetails.mobileNumber = '+91 0987654321'

console.log(userDetails)

// let key = 'email';

// console.log(userDetails[key]); // userDetails['email']

// console.log(userDetails.key);
```