JS

WEB SERVER

Welcome to the BACK

WHY DO WE NEED A BACK-END ?

(FIND 2 REASONS)

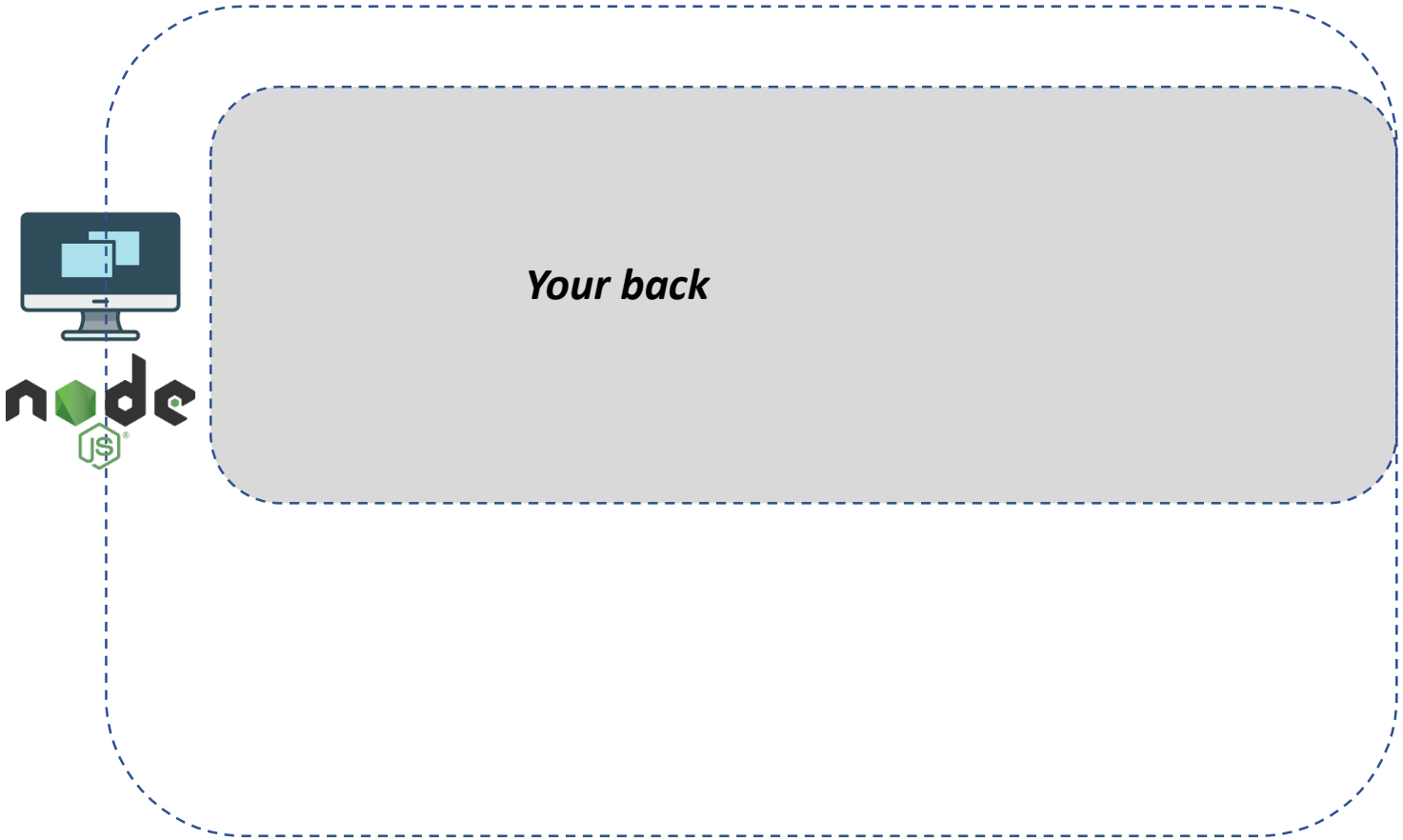
WHY DO WE NEED A BACK-END ?

KEEP DATA ON FILE OR DATABASE

MANY USERS ON THE APP

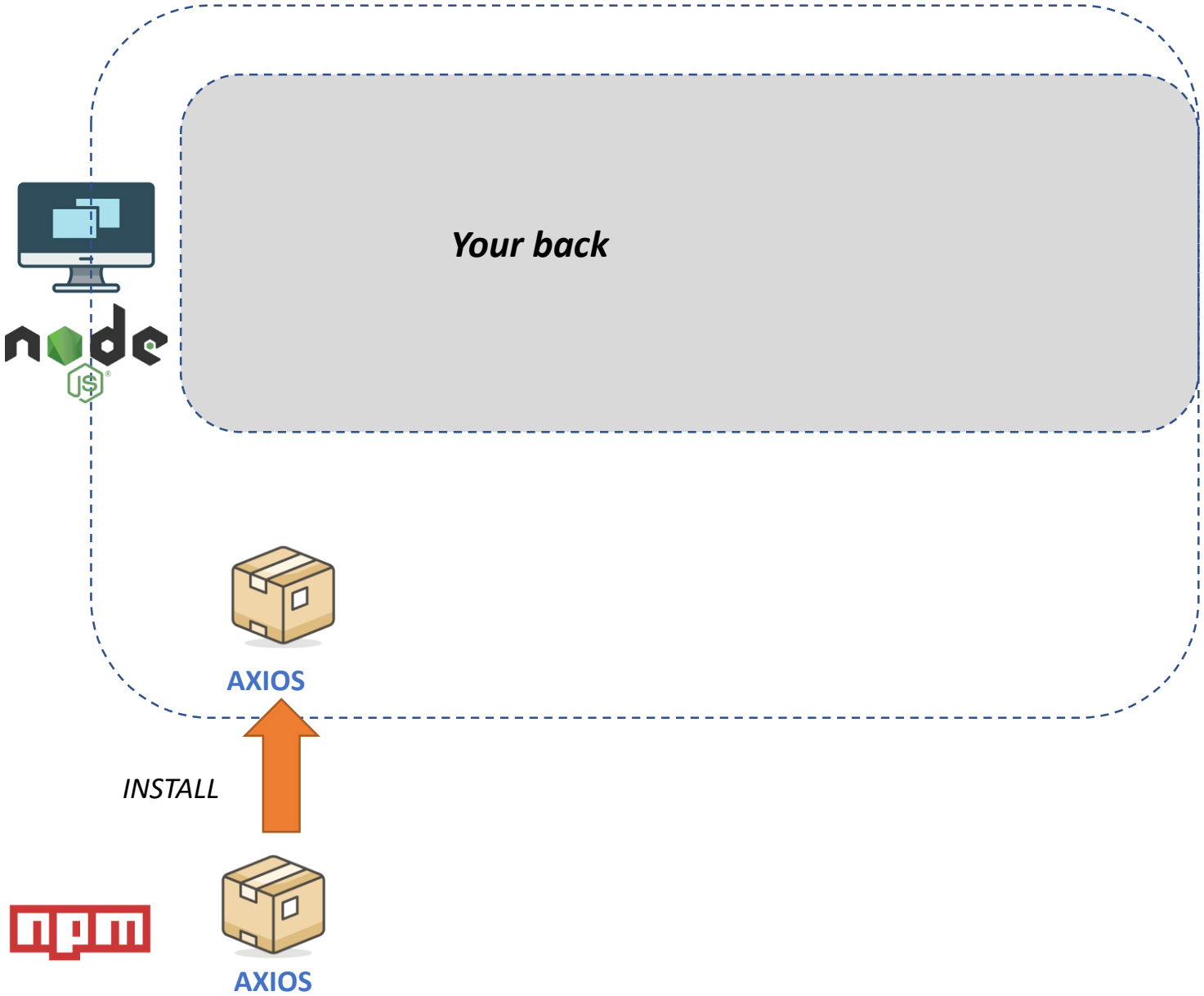
(FIND 2 REASONS)

What have we learnt so far ?



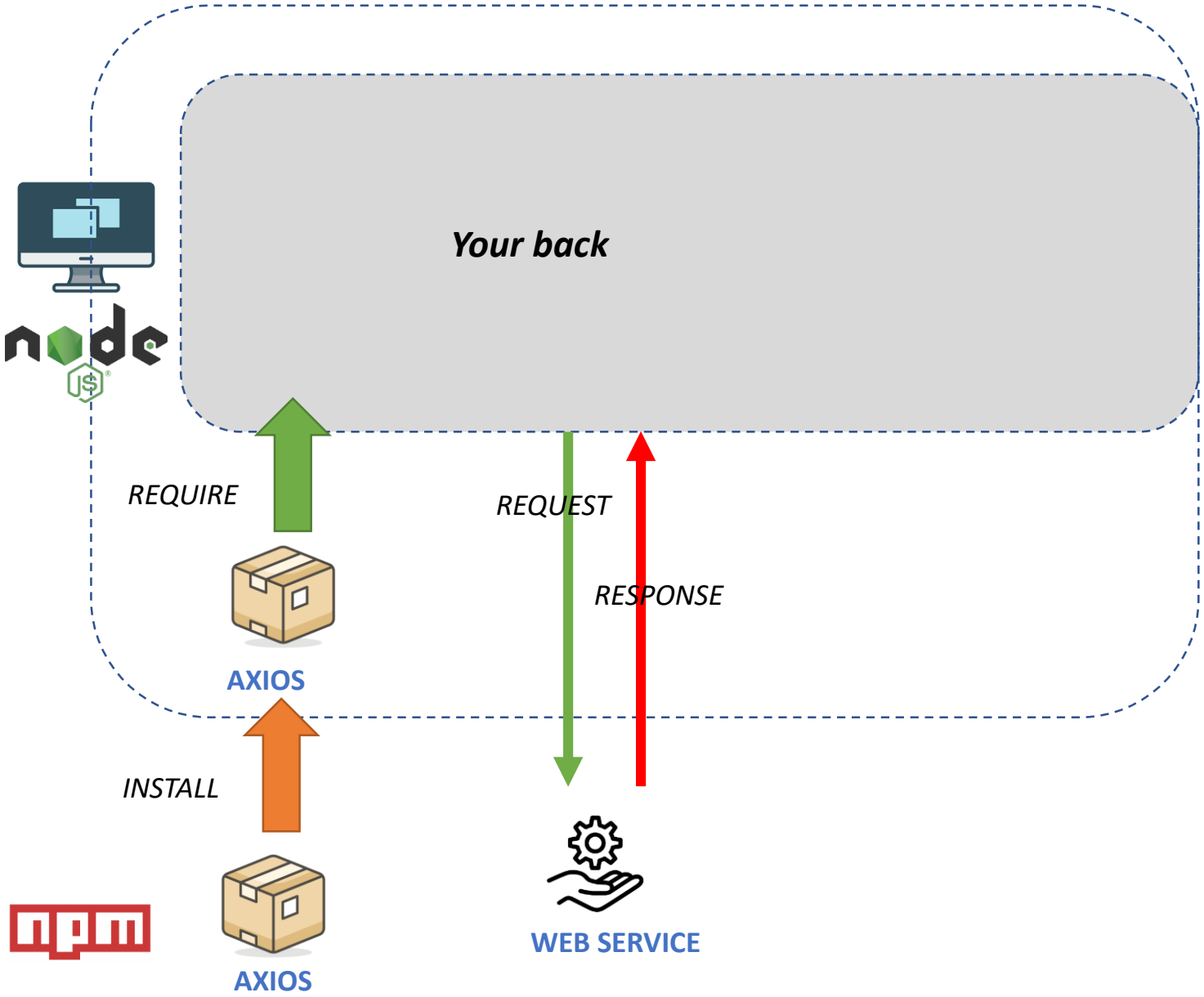
Run node on console

What have we learnt so far ?



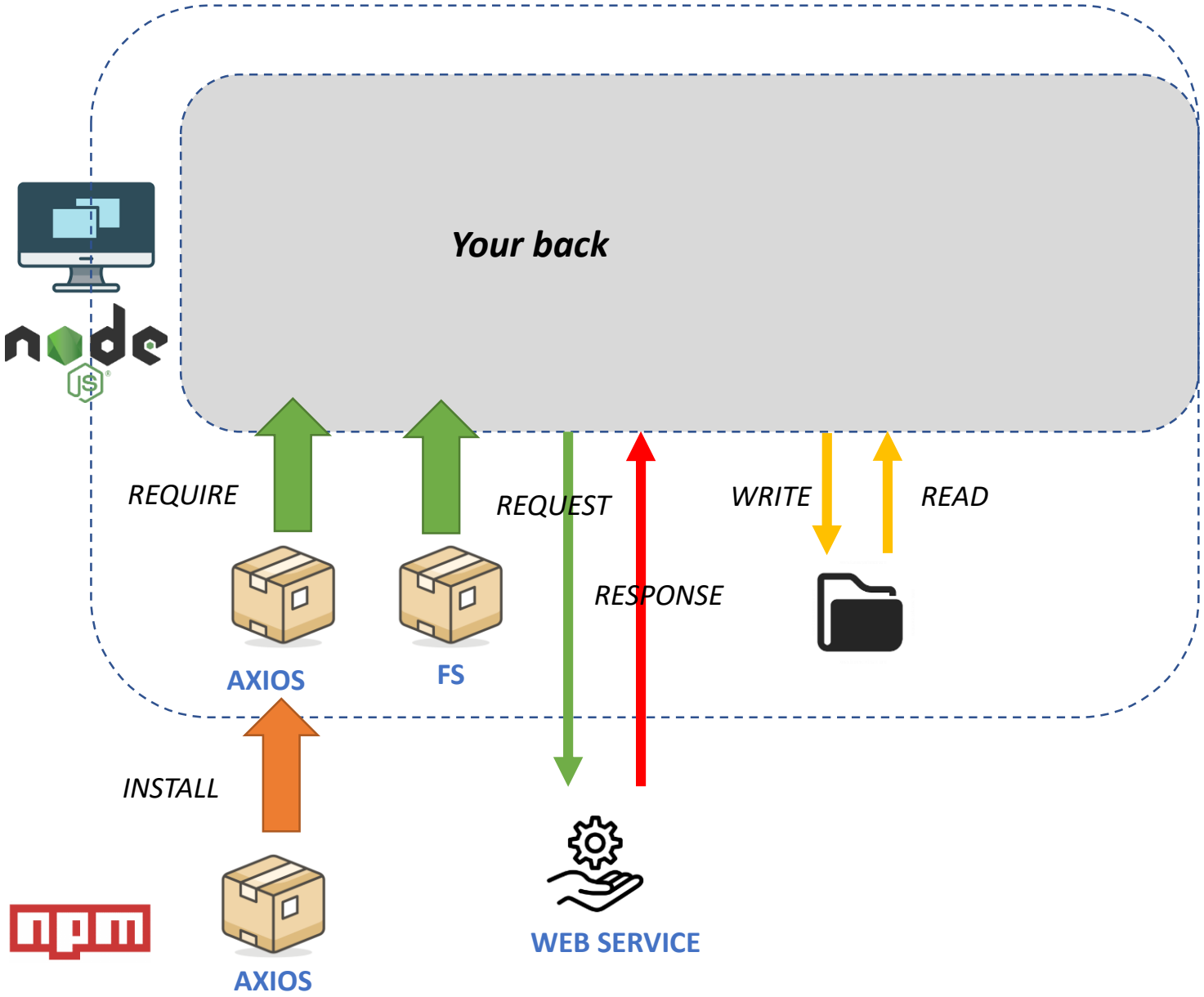
Install package from NPM repository

What have we learnt so far ?



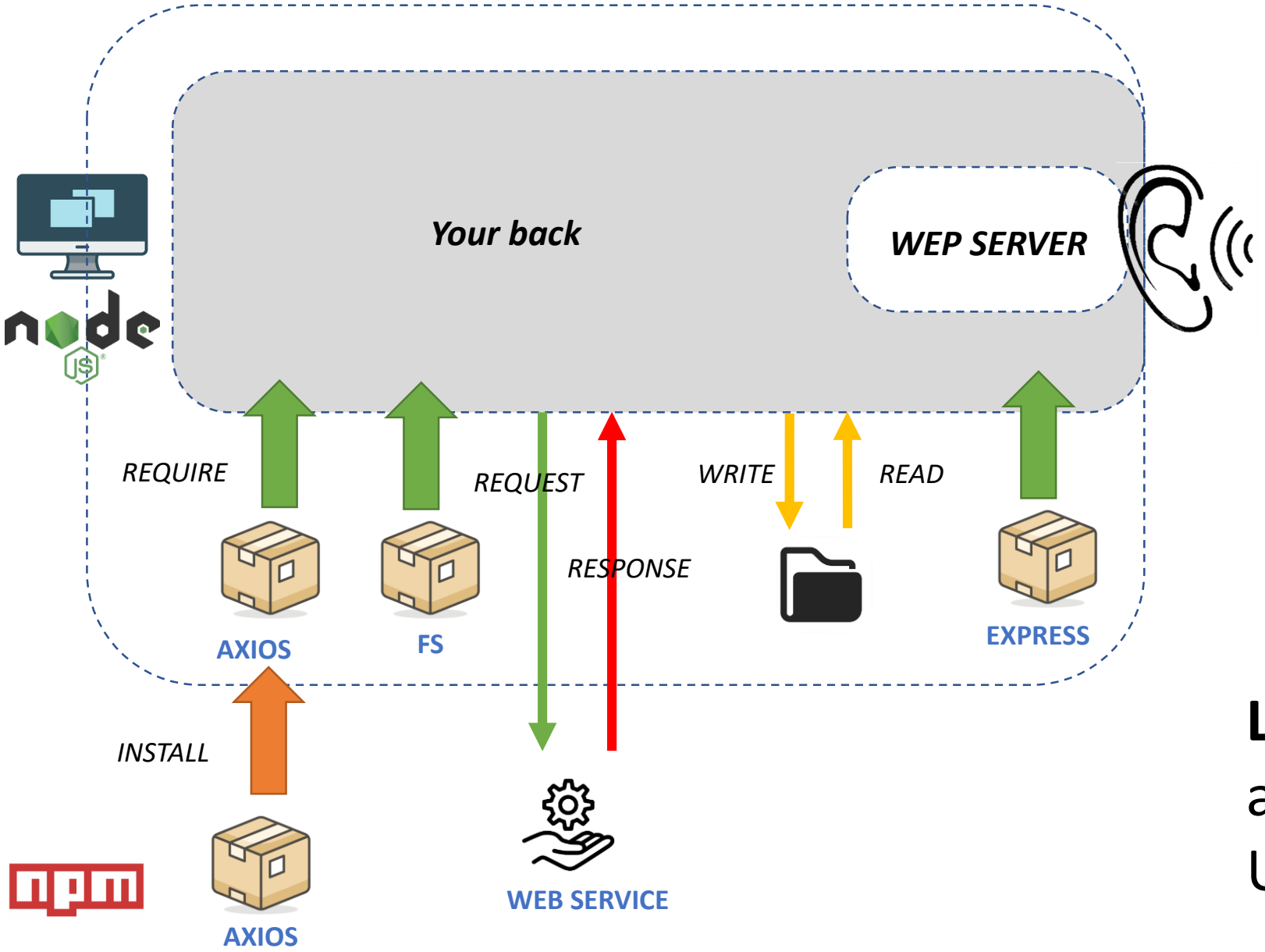
Send requests to
Web services
Using axios

What have we learnt so far ?



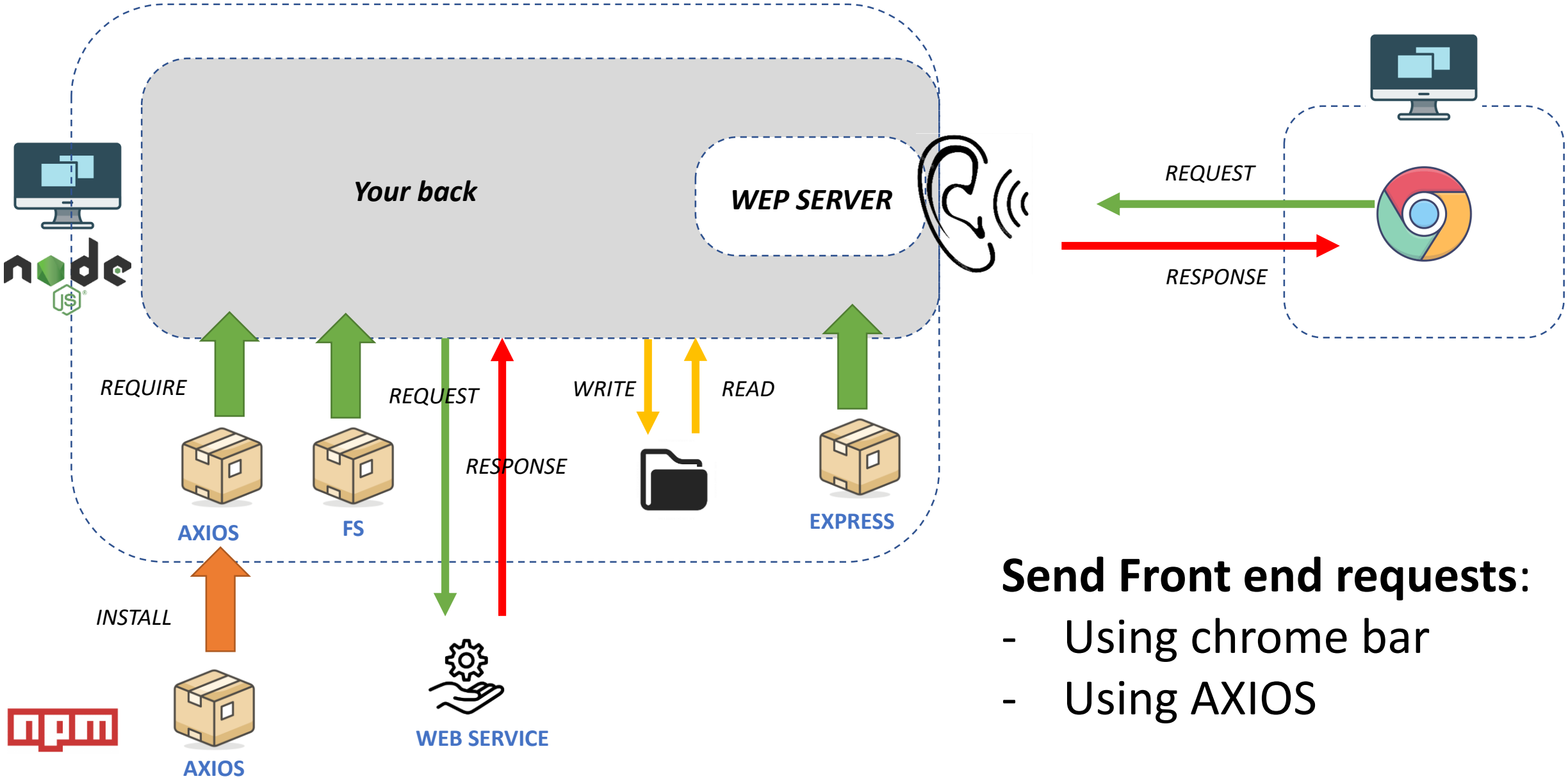
Read/Write file
Using FS module

What we will learn this week



Listen to requests on a PORT
Using Express module

Let’s communicate beetwen FRONT & BACK



Send Front end requests:

- Using chrome bar
- Using AXIOS

OBJECTIVES FOR TODAY

- ✓ What are **PORT** and **IP ADDRESSES**
- ✓ How to handle **requests** **to** your **server**
- ✓ How to send **responses** **from** your **server**
- ✓ Different **kind** of **Reponses** (text, JSON, resource ..)

An IP address is the address of your computer

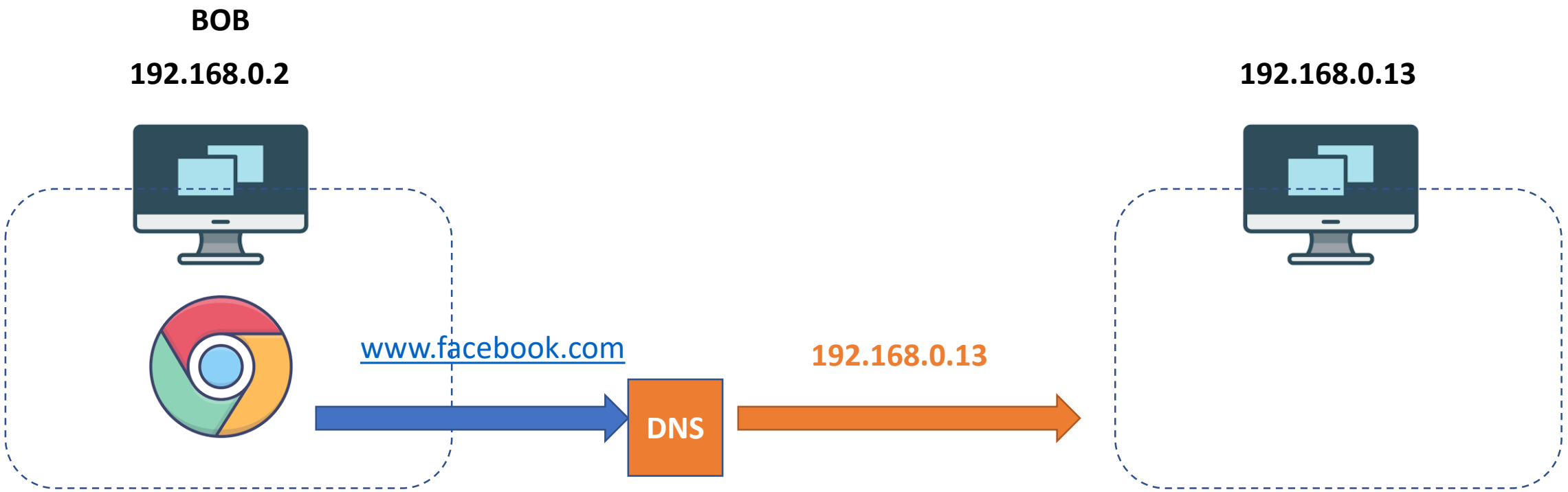
BOB

192.168.0.2

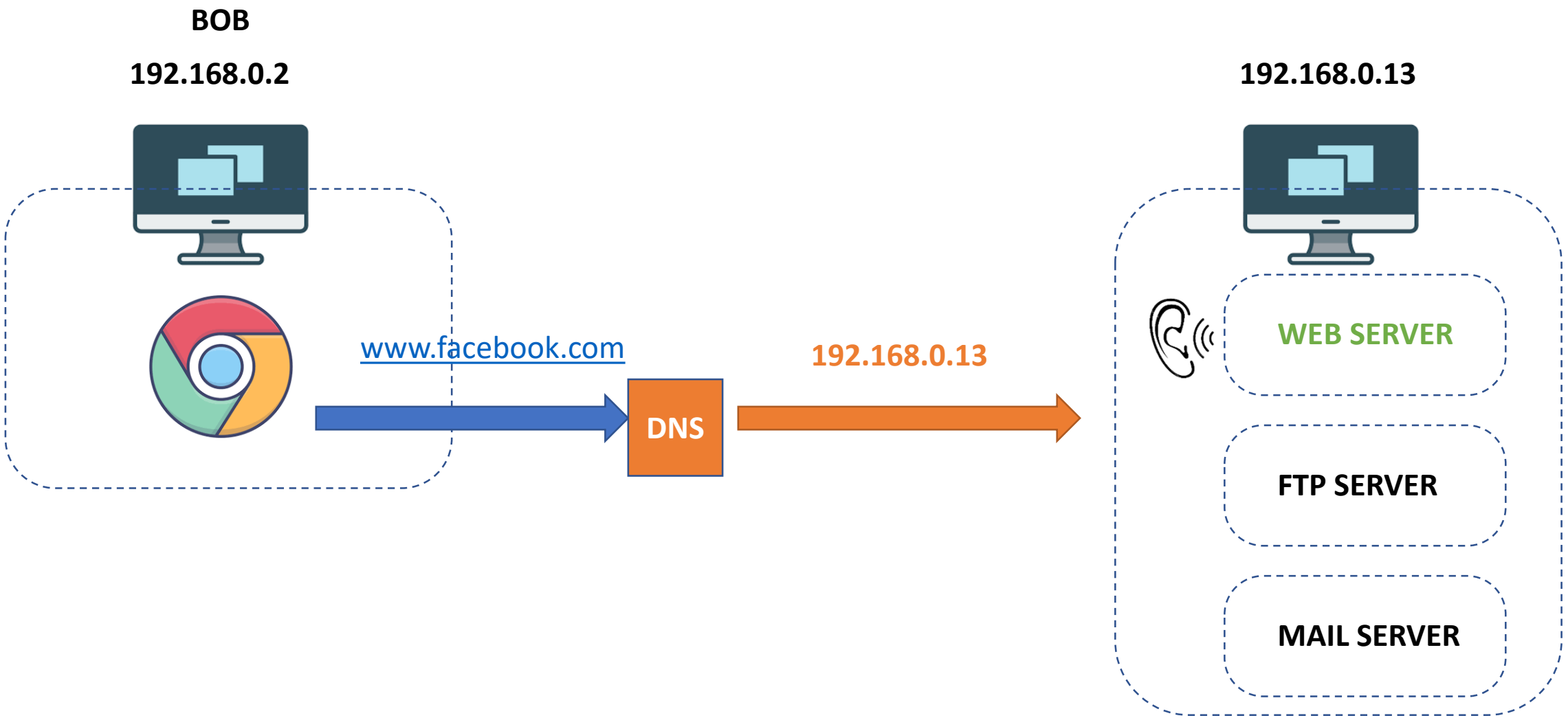
192.168.0.13

Bob wants to visit to facebok.com

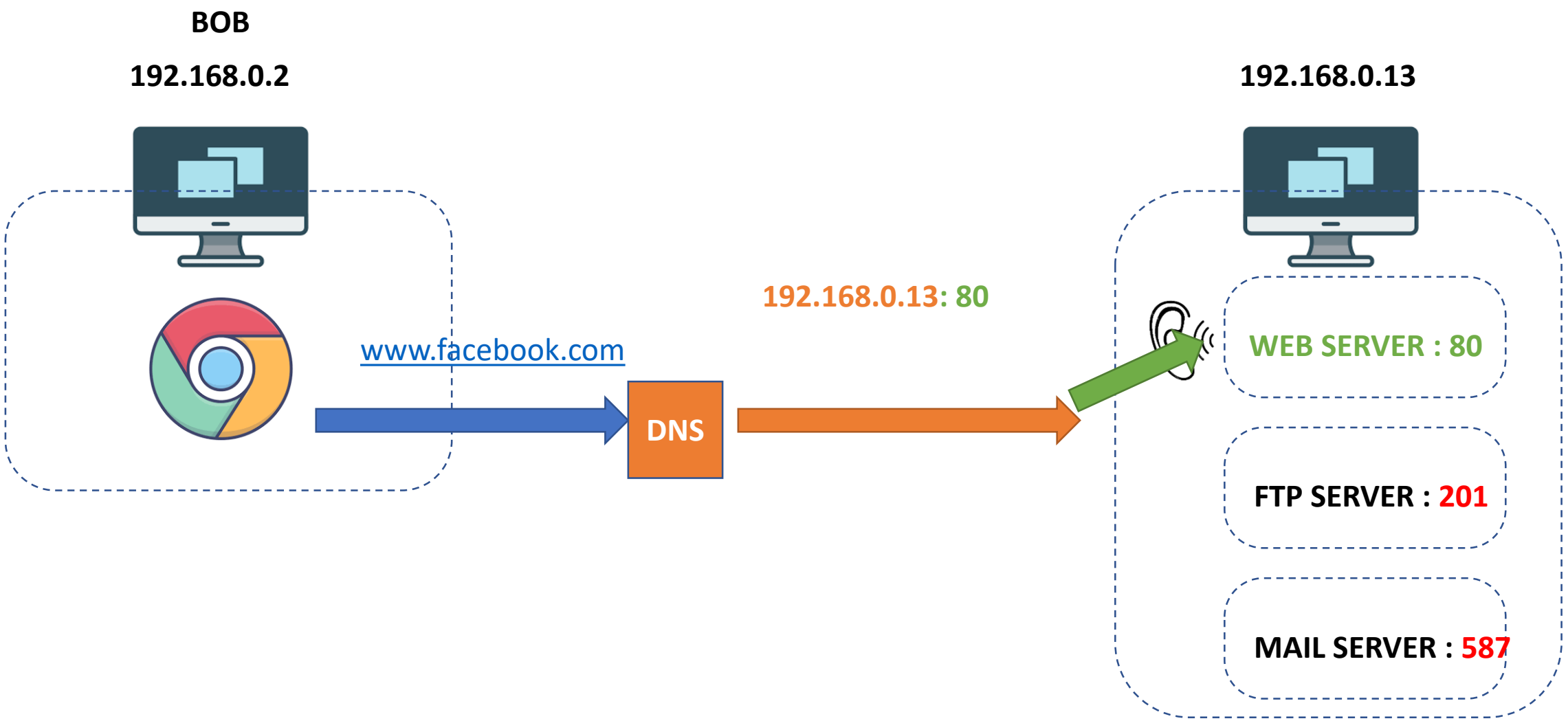
The URL is converted to an IP address



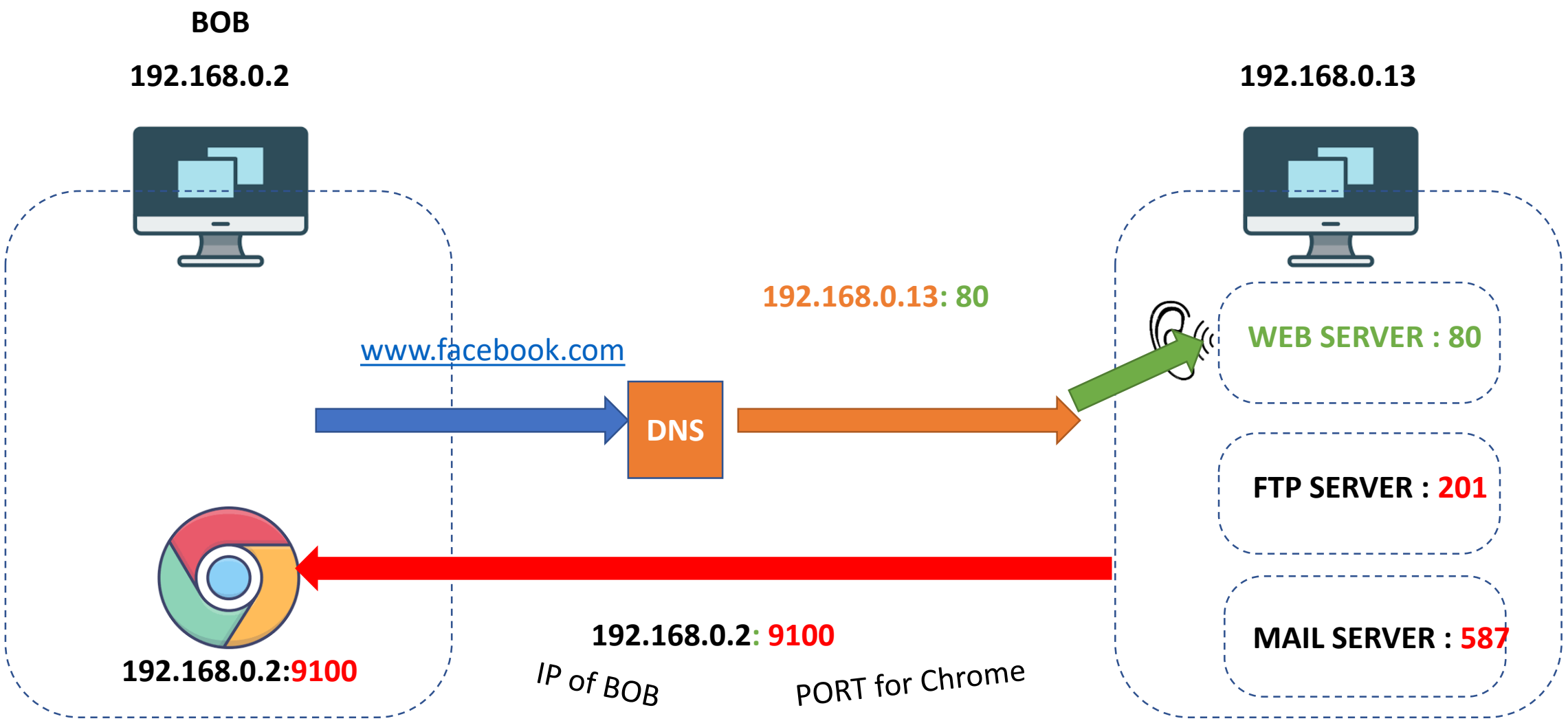
But which service to call in the server ?



All services have a PORT number assigned



After RECEIVED, the server RESPONSES

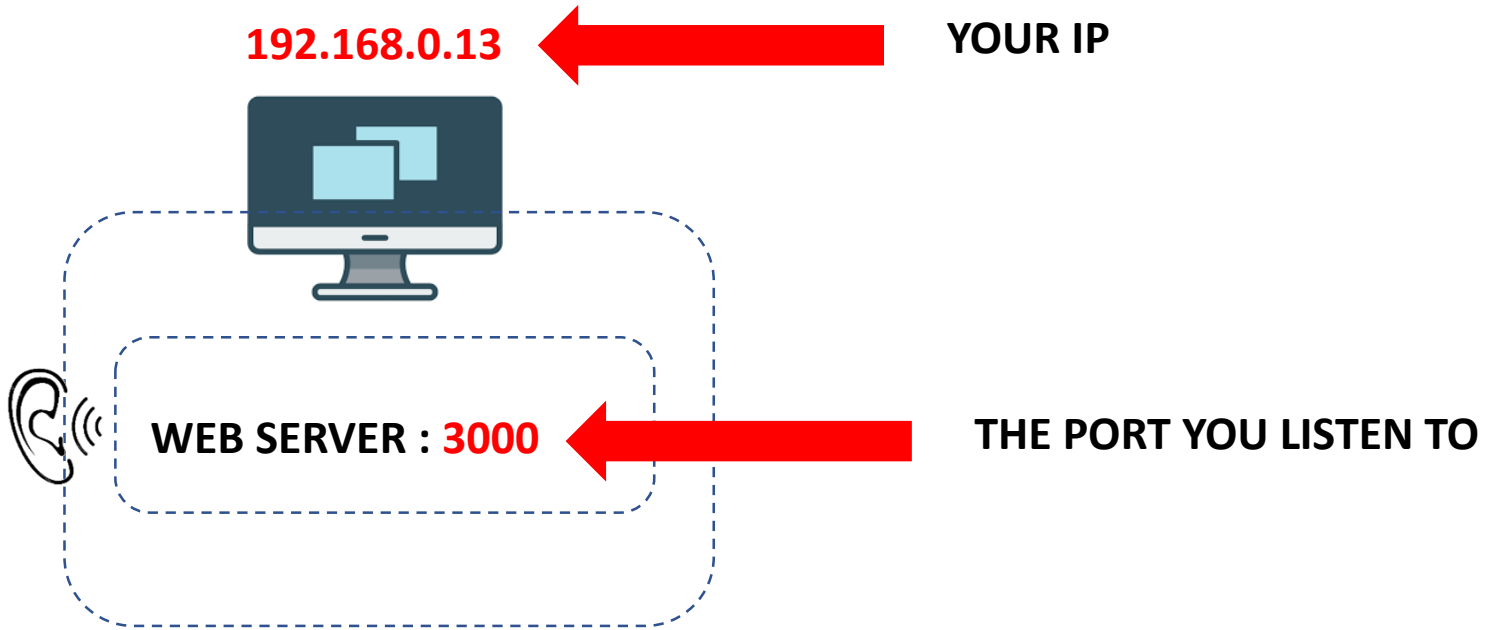


192.168.102:3000

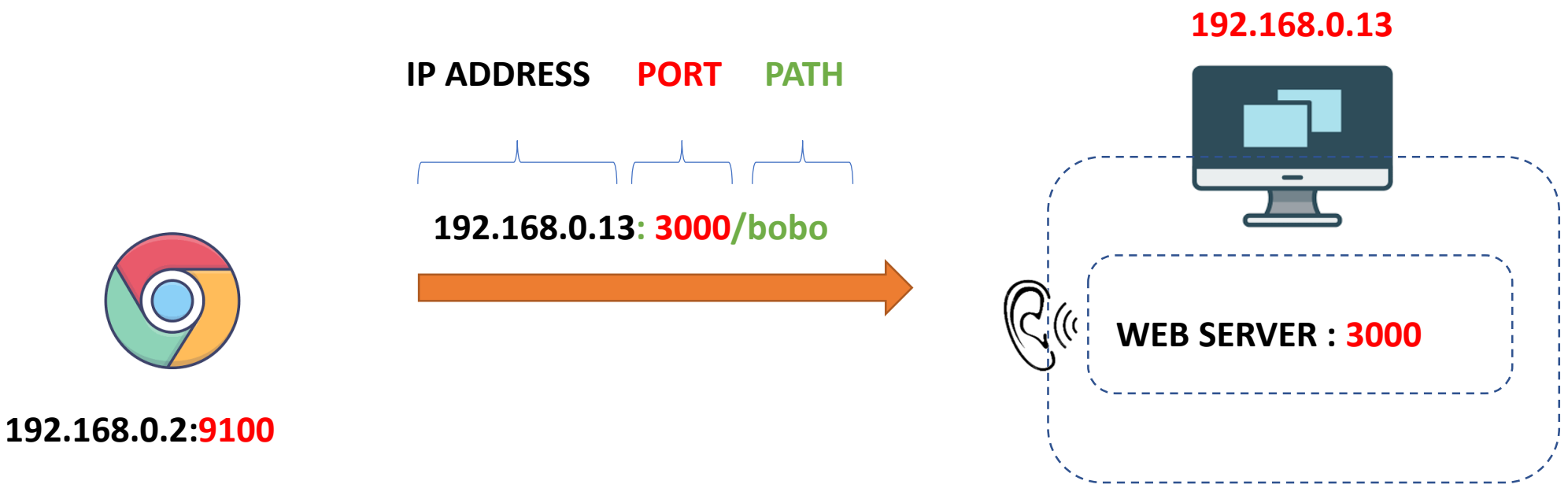
The **IP**
is the address
of the house
(the computer)

The **PORT**
is the address
of the room
(the web service)

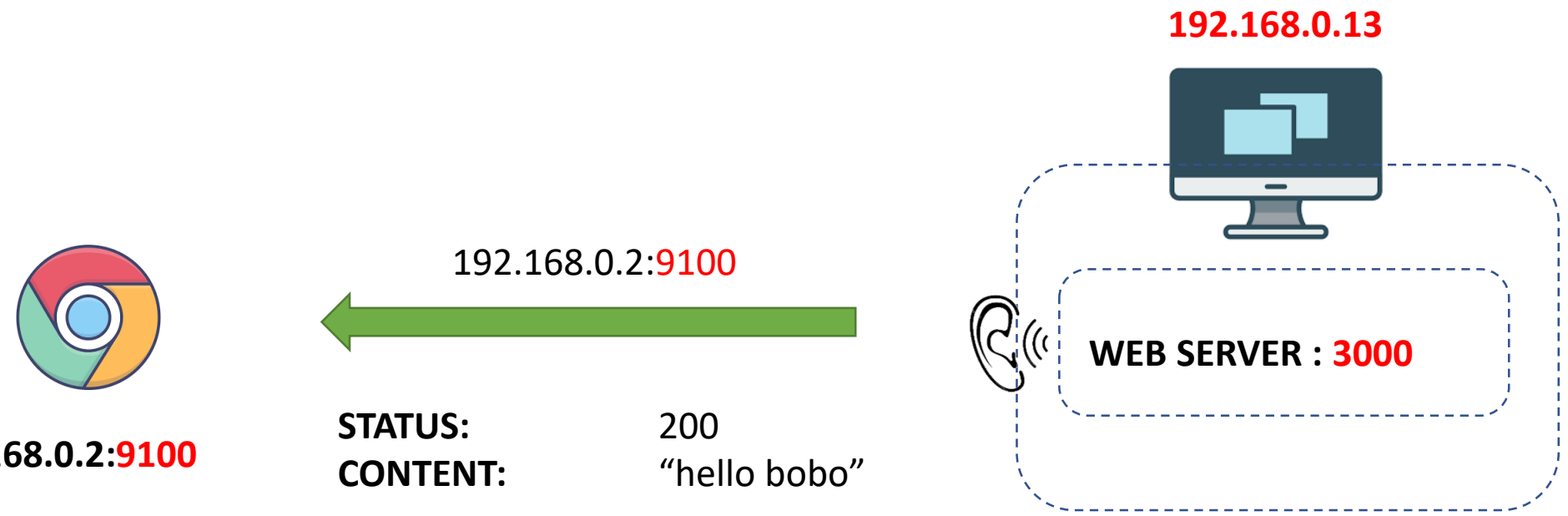
Let’s launch your first WEP server



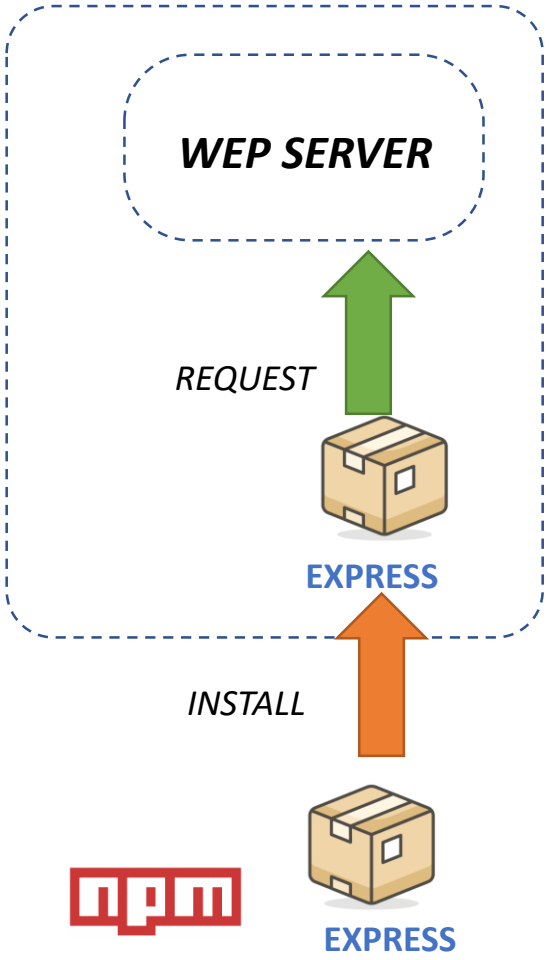
Request


IP ADDRESS
PORT
PATH
192.168.0.13: 3000/bobo
192.168.0.2:9100
192.168.0.13
WEB SERVER : 3000

Response


192.168.0.13
192.168.0.2:9100
WEB SERVER : 3000
192.168.0.2:9100
STATUS: 200
CONTENT: “hello bobo”

Let’s do it with Express package !



First step : install EXPRESS from NPM repository

Let's do it with Express package !

1- Create The server

2- Start listening port 3000

3- Send RESPONSE To client

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



// 1- the app is listening on PORT
app.listen(3000, () => {
  console.log(" listening on port  3000 ");
});



app.get("bobo", (req, res) => {
  res.send("Hello it's ronan");
});
```

THE REQUEST INFO

THE RESULT INFO

PAIR OF STUDENTS

ACTIVITY 1

15 MIN

SERVER SIDE

1 – Open activityy1/server.js

2- launch npm install to install express

3- Set the PORT to listen to 4000

4- Change the answer with your name

5- Run the server.js to start your server

BONUS !

Guess **who sent the request**: to do this, compare:

- The IP of the req : req.ip
- The file of IPS : HERE

CLIENT SIDE

1 – From browser, connect to YOUR sever:

```
localhost:4000
```

2– See what ‘s happen on

- the server console
- The browser

3 – Try to connect first to OTHER server (friends):

```
<friendIp>:4000
```

SAME SAME OR DIFFERENT ?

`localhost:4000`

`<your IP>:4000`

WHEN CLIENT REQUEST THIS…

ACTION	IP ADDRESS	PORT	PATH	PARAMETERS
POST	192.168.0.13	3000	/bobo/baba	
GET	192.168.0.13	3000		
GET	192.168.0.13	3000	/weather	?address=pnc

… YOU CATCH IT ON SERVER LIKE THIS

```
app.post(“bobo/baba", (req, res) => { });

app.get(“", (req, res) => { });

app.get(“weather", (req, res) => {
        let address = req.query.address

});
```

How to use the query parameters?

SERVER SIDE

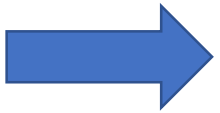
```
const teacherScore = {
  ronan: 45,
  rady: 99,
  him: 50,
};

app.get("/results", (req, res) => {
  let name = req.query.name;
  let score = teacherScore[name];

  res.send("Score for teacher " + name + " is " + score);
});
```

CLIENT SIDE

GET http://localhost:3000/results?name=him



Score for teacher him is 50

PAIR OF STUDENTS **ACTIVITY 2** 15 MIN

CLIENT SIDE

Send following requests to server:

<SERVER IP>: 3000/bobo/baba

<SERVER IP>: 3000

<SERVER IP>: 3000/weather?address=pnc

<SERVER IP>: 3000/teacher/skills?name=rady

<SERVER IP>: 3000/teacher/skills?name=rronan

SERVER SIDE

Catch each request and return something

```
Bobo and baba are happy

Hello

The weather at PNC is 25°

The skill of Rady is: Javascript

The skill of Ronan is: nothing found
```