

Universidad Distrital Francisco José de Caldas

Facultad de Ingeniería

Ingeniería de sistemas



**UNIVERSIDAD DISTRICTAL
FRANCISCO JOSÉ DE CALDAS**
Acreditación Institucional de Alta Calidad

Workshop 2

Juan Sebastian Colorado Caro - 20202673001

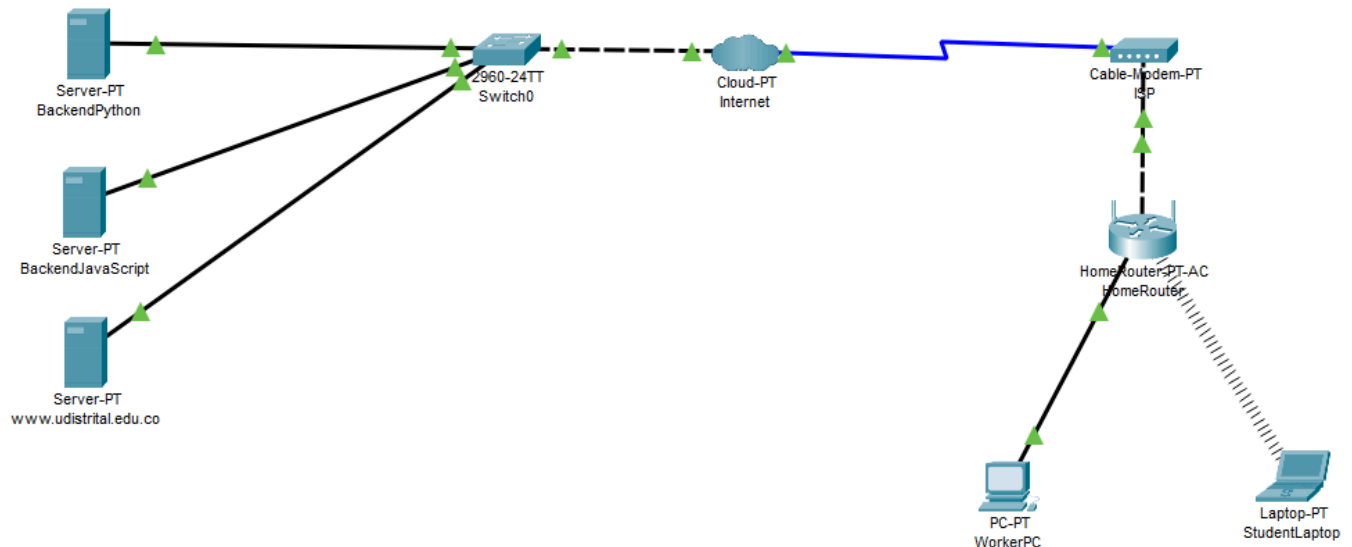
Profesor: Carlos Andrés Sierra Virgüez

Bogotá – Colombia

03 de febrero de 2025

Topology and modifications

During the development of this workshop, the network was modified two times. Firstly by adding a switch to the LAN of the servers, this was done in order of getting the traffic coming from the internet to the correct destiny.



In this case the switch device does not need to be configured internally in the CLI. This ensures that workerPC and StudentLaptop can use the backend services available in this LAN, getting direct connection to them.

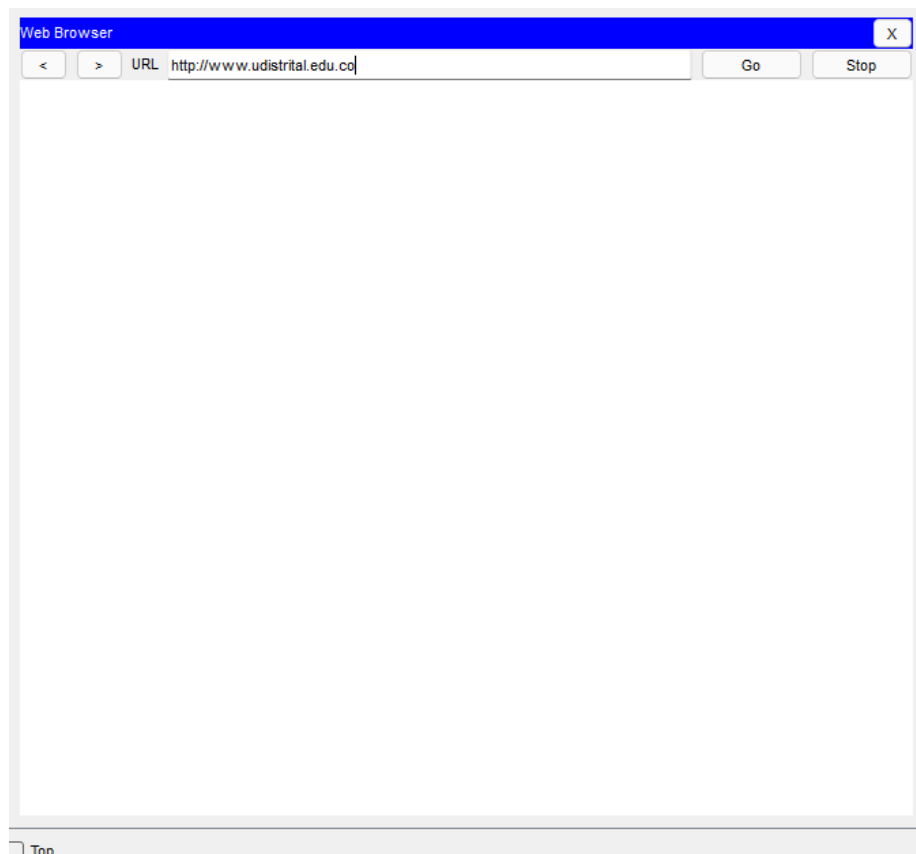
Also, the HTML and styles.css files were modified, the result is shown below:



The second modification was done taking in mind the bonus section which is gonna be explained later in this document.

Analysis using simulation

Worker PC requesting the web page



First: DNS resolution

When the WorkerPC requests the webpage, a DNS query is sent to resolve the domain name. The DNS request follows these steps:

- **Application Layer (Layer 7):** The request is generated by the browser.
- **Transport Layer (Layer 4):** The request is encapsulated in a UDP segment (port 53).
- **Network Layer (Layer 3):** The UDP segment is placed inside an IP packet.
- **Data Link Layer (Layer 2):** The packet is encapsulated in an Ethernet frame and sent to the switch.
- **Physical Layer (Layer 1):** The frame is transmitted over the network.

The DNS request travels from the WorkerPC to the HomeRouter, then through the Cable Modem to the Cloud. The DNS server responds with the IP address of the requested domain, following the same OSI process in reverse.

Second: TCP Connection (Handshake)

Once the IP is resolved, the WorkerPC initiates a TCP connection with the frontend server using the three-way handshake:

1. **SYN:** WorkerPC sends a TCP SYN packet to port 80 of the frontend server.
2. **SYN-ACK:** The server responds with a SYN-ACK packet.
3. **ACK:** WorkerPC sends an ACK, establishing the connection.

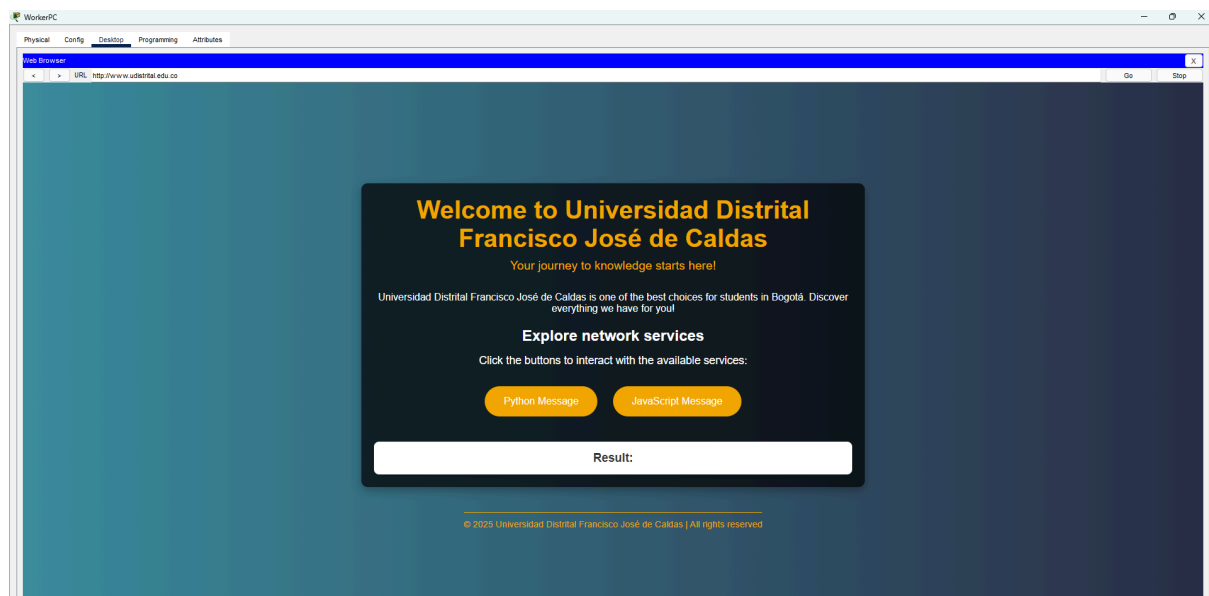
This occurs at the **Transport Layer (Layer 4)** and is encapsulated in IP packets at the **Network Layer (Layer 3)**

Third: HTTP requests and responses

With the TCP connection established, WorkerPC sends an HTTP GET request:

- The request is encapsulated in a TCP segment (Layer 4) and an IP packet (Layer 3).
- The frontend server processes the request and responds with multiple HTTP packets containing the requested webpage and its resources.
- These packets include HTML, CSS, and JavaScript files, each sent in separate TCP segments due to size limitations.
- The WorkerPC acknowledges receipt of these packets with TCP ACK responses.

Once all required HTTP packets arrive, the browser processes and renders the page. At this point, the webpage is fully loaded.

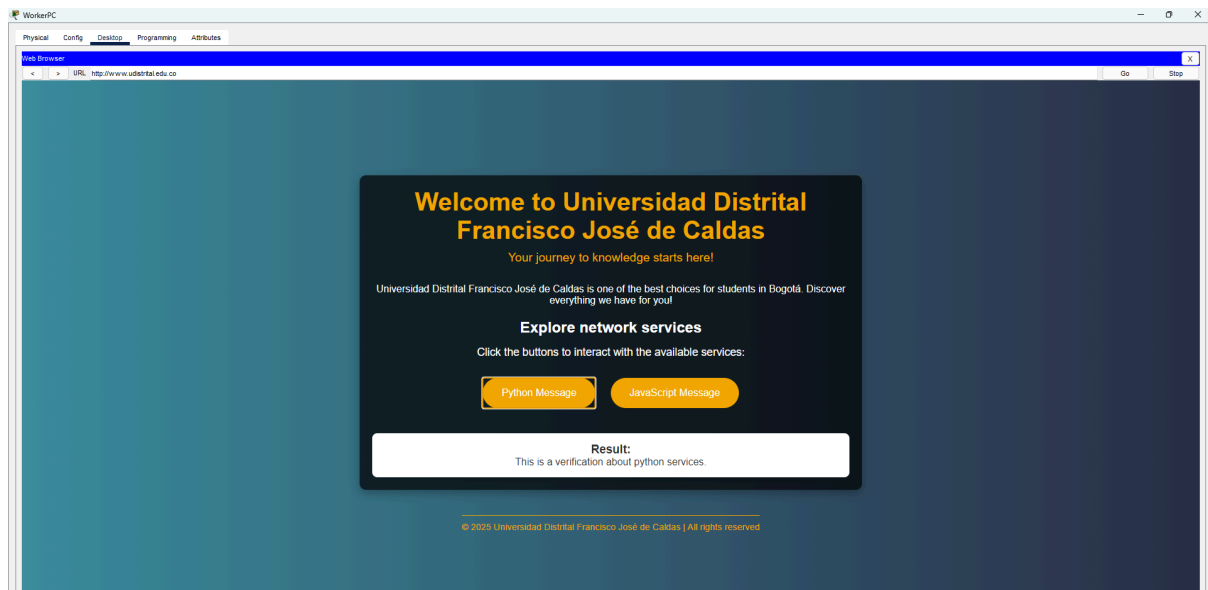


Worker PC clicking the button for one of the backend services

Clicking one of the two buttons initiates a new network operation to communicate with the backend:

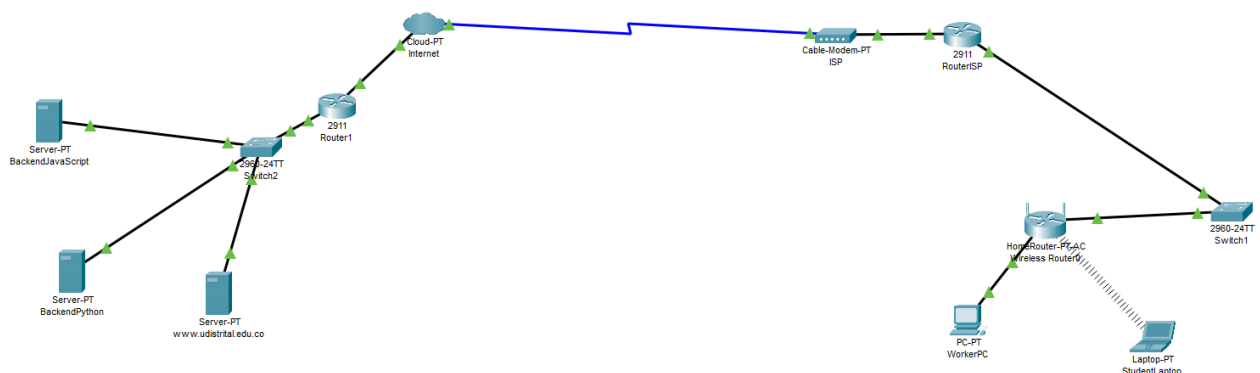
1. WorkerPC sends a **TCP SYN packet** to 193.168.100.201 (BackendPython).
2. The switch broadcasts an **ARP request** to resolve the MAC address.
3. BackendPython responds with an **ARP reply** containing its MAC address.
4. WorkerPC sends an **HTTP request** to 193.168.100.201/healthcheck.
5. BackendPython responds with a TCP acknowledgment followed by an HTTP response.
6. The frontend updates the webpage with the received data.

In this case the worker PC is communicating directly with the BackendPython server.



VLANS: Modifying the network to make this work

In the first place the network topology was modified in order of making two different VLANS where the frontend server could communicate with backend servers but worker PC and StudentLapton cannot.



From the previous topology to this one the changes are:

1. Two routers were added to the network, this was done because the VLANS needed this type of device to route the traffic from one VLAN to another. For this static and OSPF routing was used. The router on the right simulates an ISP router.
2. A switch was added to the homerouter LAN, this was done because if we connect the homerouter directly to the ISP router the LAN doesn't work.

DHCP

ROUTER ISP: It was needed to assign an IP address to the homerouter.

```
ip dhcp excluded-address 172.12.0.1
!
ip dhcp pool POOL-HOMEROUTER
network 172.12.0.0 255.255.255.0
default-router 172.12.0.1
```

Routing

ROUTER ISP: Static routing was used in this router to send the requests to the servers LAN. Any traffic going to the IP address "193.168.100.0" is gonna be sent out the GigabitEthernet 0/1 interface.

```
ip classless
ip route 193.168.100.0 255.255.255.0 GigabitEthernet0/1
```

Also dynamic routing for the LAN:

```
router ospf 1
 log-adjacency-changes
 network 10.0.0.0 0.0.0.3 area 0
 network 172.12.0.0 0.0.0.255 area 0
!
```

ROUTER 1: Dynamic routing was used for this router. OSPF

```
router ospf 1
 log-adjacency-changes
 network 193.168.0.0 0.0.0.255 area 0
 network 10.0.0.0 0.0.0.3 area 0
!
```

VLANS CREATION

Two vlans were created:

- VLAN 10: Where the backend servers will be.
 - Address: 193.168.200.0 255.255.255.0
- VLAN 20: Where the frontend server will be.
 - Address: 193.168.100.0 255.255.255.0

The commands used for this were (CLI of switch 2):

```
enable
configure terminal
interface fastEthernet 0/3
switchport mode access
switchport access vlan 10
```

```
enable
configure terminal
interface fastEthernet 0/4
switchport mode access
switchport access vlan 10
```

```
enable
configure terminal
interface fastEthernet 0/2
```

```
switchport mode access
switchport access vlan 20
```

```
enable
configure terminal
interface fastEthernet 0/1
switchport mode trunk
```

Now we need to configure the router to communicate both VLANs:

We do this by configuring the interface gigabitEthernet 0/0 which is connected to the switch, with two sub interfaces.



GigabitEthernet0/0.10 for vlan 10
GigabitEthernet0/0.20 for vlan 20

```
interface GigabitEthernet0/0.10
 encapsulation dot1Q 10
 ip address 193.168.200.1 255.255.255.0
!
interface GigabitEthernet0/0.20
 encapsulation dot1Q 20
 ip address 193.168.100.1 255.255.255.0
!
```



This configurations ensure communication between both VLANs.

Testing

Packet sent from worker PC to backendPython server:

	Failed	WorkerPC	BackendPython	ICMP		0.000	N	1	(edit)
---	--------	----------	---------------	------	---	-------	---	---	--------

Packet sent from Frontend server (www.udistrital.edu.co) to the backendPython:

	Successful	www.udistrital.edu.co	BackendPython	ICMP		0.000	N	0	(edit)
---	------------	-----------------------	---------------	------	---	-------	---	---	--------