

## PRACTICAL 7

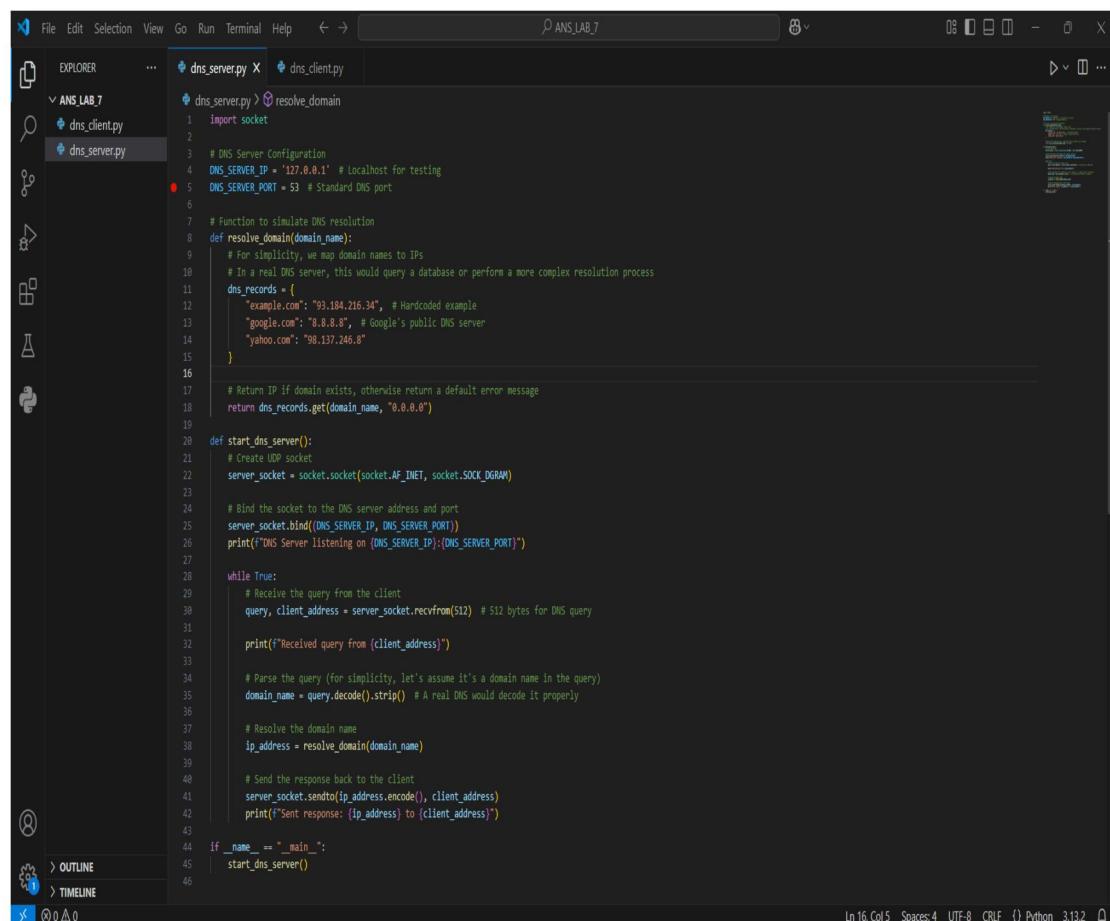
**Aim:** Simulation of DNS using UDP sockets.

**DNS Server:** A server that listens for queries and responds with the appropriate IP address.

The server will listen for DNS queries, process them, and send back an appropriate response. This example will simulate a very basic DNS server that responds with a predefined IP address.

**Code:**

**Dns\_server.py**



```

File Edit Selection View Go Run Terminal Help ← → ⌘ ANS_LAB_7
EXPLORER dns_server.py X dns_client.py
dns_server.py > resolve_domain
1 import socket
2
3 # DNS Server Configuration
4 DNS_SERVER_IP = "127.0.0.1" # Localhost for testing
5 DNS_SERVER_PORT = 53 # Standard DNS port
6
7 # Function to simulate DNS resolution
8 def resolve_domain(domain_name):
9     # For simplicity, we map domain names to IPs
10    # In a real DNS server, this would query a database or perform a more complex resolution process
11    dns_records = {
12        "example.com": "93.184.216.34", # Hardcoded example
13        "google.com": "8.8.8.8", # Google's public DNS server
14        "yahoo.com": "98.137.246.8"
15    }
16
17    # Return IP if domain exists, otherwise return a default error message
18    return dns_records.get(domain_name, "0.0.0.0")
19
20 def start_dns_server():
21    # Create UDP socket
22    server_socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
23
24    # Bind the socket to the DNS server address and port
25    server_socket.bind((DNS_SERVER_IP, DNS_SERVER_PORT))
26    print(f"DNS Server listening on {DNS_SERVER_IP}:{DNS_SERVER_PORT}")
27
28    while True:
29        # Receive the query from the client
30        query, client_address = server_socket.recvfrom(512) # 512 bytes for DNS query
31
32        print(f"Received query from {client_address}")
33
34        # Parse the query (for simplicity, let's assume it's a domain name in the query)
35        domain_name = query.decode().strip() # A real DNS would decode it properly
36
37        # Resolve the domain name
38        ip_address = resolve_domain(domain_name)
39
40        # Send the response back to the client
41        server_socket.sendto(ip_address.encode(), client_address)
42        print(f"Sent response: {ip_address} to {client_address}")
43
44    if __name__ == "__main__":
45        start_dns_server()

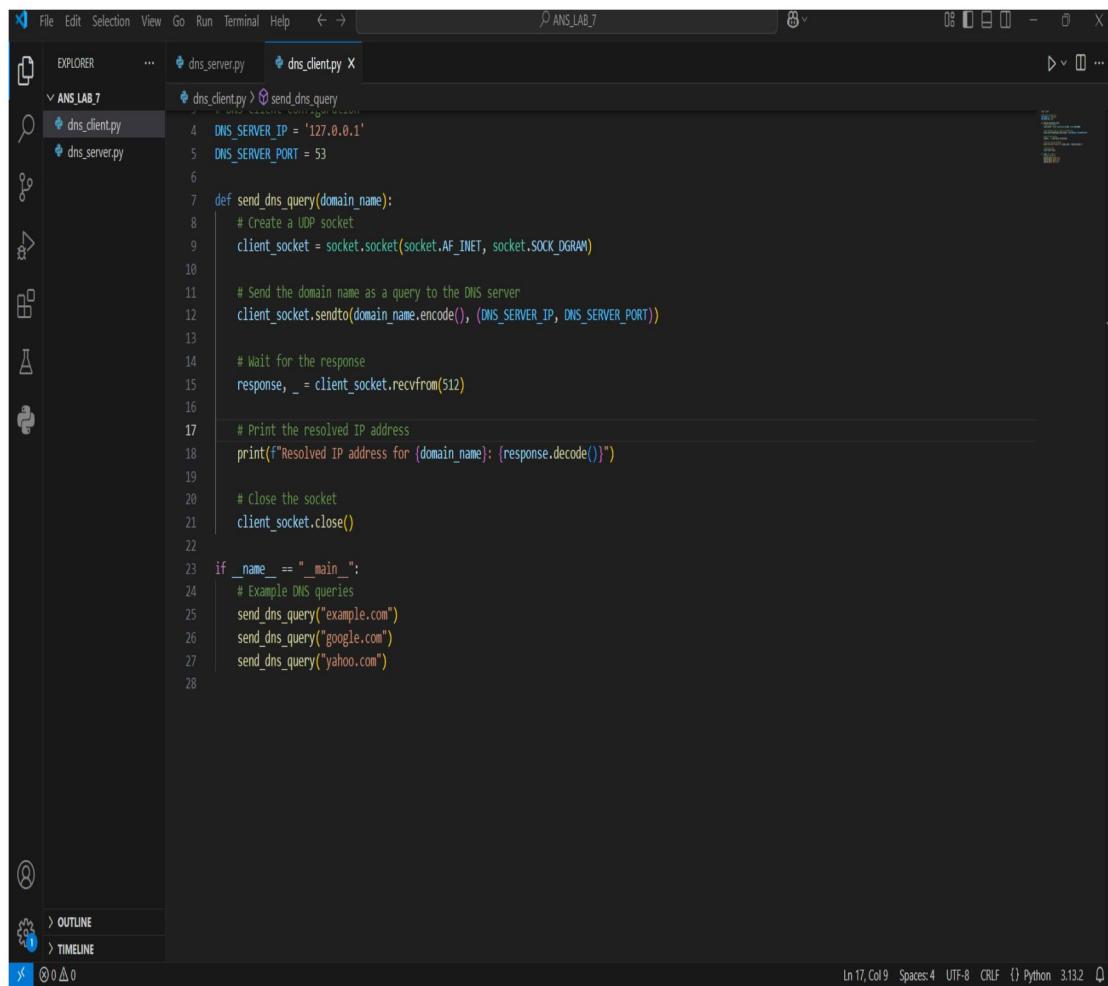
```

Ln 16 Col 5 Spaces:4 UTF-8 CRLF {} Python 3.13.2

**DNS Client:** A client that queries a DNS server for a domain's IP address.

This client will create a UDP socket, send a DNS query to a DNS server, and wait for the response.

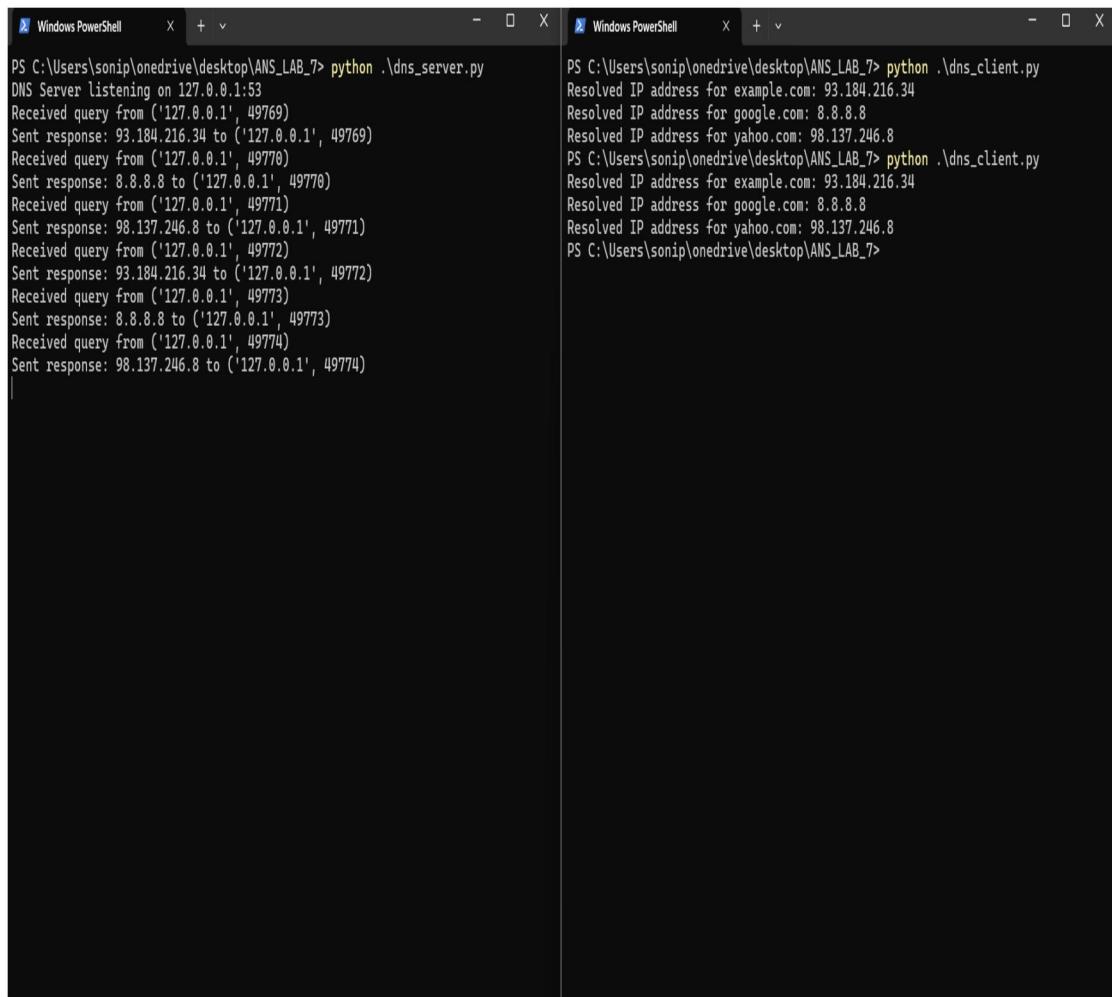
### Dns\_client.py



The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows two files: dns\_client.py and dns\_server.py.
- Editor:** The dns\_client.py file is open, displaying Python code for a DNS client. The code defines a function `send_dns_query` that sends a DNS query to a specified server and prints the resolved IP address. It also includes a main block with example queries for "example.com", "google.com", and "yahoo.com".
- Status Bar:** Shows the file has 17 lines, 9 columns, 4 spaces, and is in UTF-8 format, saved with CRLF, and is a Python file (Ln 17, Col 9, Spaces: 4, UTF-8, CRLF, Python, 3.13.2).

## Output:



The image shows two separate Windows PowerShell windows side-by-side. Both windows have a dark background and white text.

The left window (DNS Server) displays the following log output:

```
PS C:\Users\sonip\onedrive\Desktop\ANS_LAB_7> python .\dns_server.py
DNS Server listening on 127.0.0.1:53
Received query from ('127.0.0.1', 49769)
Sent response: 93.184.216.34 to ('127.0.0.1', 49769)
Received query from ('127.0.0.1', 49770)
Sent response: 8.8.8.8 to ('127.0.0.1', 49770)
Received query from ('127.0.0.1', 49771)
Sent response: 98.137.246.8 to ('127.0.0.1', 49771)
Received query from ('127.0.0.1', 49772)
Sent response: 93.184.216.34 to ('127.0.0.1', 49772)
Received query from ('127.0.0.1', 49773)
Sent response: 8.8.8.8 to ('127.0.0.1', 49773)
Received query from ('127.0.0.1', 49774)
Sent response: 98.137.246.8 to ('127.0.0.1', 49774)
```

The right window (DNS Client) displays the following log output:

```
PS C:\Users\sonip\onedrive\Desktop\ANS_LAB_7> python .\dns_client.py
Resolved IP address for example.com: 93.184.216.34
Resolved IP address for google.com: 8.8.8.8
Resolved IP address for yahoo.com: 98.137.246.8
PS C:\Users\sonip\onedrive\Desktop\ANS_LAB_7> python .\dns_client.py
Resolved IP address for example.com: 93.184.216.34
Resolved IP address for google.com: 8.8.8.8
Resolved IP address for yahoo.com: 98.137.246.8
PS C:\Users\sonip\onedrive\Desktop\ANS_LAB_7>
```

## Conclusion:

This simulation of DNS using UDP sockets, This demonstrated the basic functionality of a DNS system, where a client sends a domain name query to a server, and the server responds with a predefined IP address.