Bansilal Ramnath Agarwal Charitable Trust's

Vishwakarma Institute of Technology, Pune-37

(An autonomous Institute of Savitribai Phule Pune University)

Department Of CSE (AI AND AIML)

Lab Manual

Course Code	Course Name	Teaching Scheme (Hrs. / Week)	Credits
CSXXXX	Computer Network Technology Laboratory	2	1

Course Outcomes:

1. Select topology, essential components of physical layer and networking devices to design computer networks.
2. Build wired and wireless intranet with correct communication and service frameworks.
3. Develop Client-Servers by the means of correct standards, protocols and technologies.
4. Build single page applications using REACT as a reusable UI component technology.
5. Write Web API/RESTful API application programming interface to communicate with Springboot as a server side technology.
6. **[Group Assignment]** Design and develop three tier enterprise application using client side, server side and back end technologies.

Class: - TY

Branch: - Computer Engineering

Year: - 2024-25

Prepared By: - Dr. M L Dhore

Required S/W and H/W: C++, JAVA, Postman, REACT, Springboot. Wireshark, Linux Intel(R) Core(TM) i13 16 GB RAM

Contents

Unit-V Springboot **5)** Design and develop a responsive website to prepare one semester result of VIT students using REACT, Springboot and MySQL/ MongoDB/Oracle. Take any four subjects with MSE Marks (30%) ESE Marks (70%).	**CO4, CO5. CO6**	36
Unit-VI NODE JS **6)** Design and develop a responsive website for an online book store using REACT, Node JS/ PHP and MySQL/ MongoDB/Oracle having 1) Home Page2) Login Page 3) Catalogue Page: 4) Registration Page: (database)	**CO4, CO5. CO6**	39

Experiment Number: 01

Title: Setting up small wired and wireless computer networks and hands on networking commands

OBJECTIVES:

1. To learn how to setup a wired and wireless network and understand working of various internetworking devices

2. To learn the network commands

PROBLEM STATEMENT

1a) Setting up small wired computer network:
Set up a small wired network of 2 to 4 computers using Hub/Switch/. It includes Preparation of Cables and setting up wired network.

1b) Setting up small wireless computer network and hands-on networking command:
Set up a small wired network of 2 to 4 computers using access point and ask students to access it on their wireless gadgets.

Hands on for network commands - ping, pathping, ipconfig/ifconfig, arp, netstat, nbtstat, nslookup, route, traceroute/tracert, nmap.

THOERY: TYPES OF NETWORK
Common examples of area network types are:

Local Area Network (LAN):
LAN is privately owned networks use to interconnect computers within a single building or campus up to few kilometers in size. For the LAN, diameters spans over 550 meters to 2.5 Kilometers. Nowadays organizations have Campus Wide Network which is an extension of LAN using OFC at backbone and diameter IS up to 10 Kilometers. LANs are setup using IEEE802.3 standard. (**Active**).

Wide Area Network (WAN):
WAN is a telecommunication or computer network span over the area often country or continent. WAN: 100km to 1000 km Country and Continent. Uses Packet Switching and Data Switching Exchanges. Uses IEEE 802.1 standard.

Personal Area Network (PAN):
PAN is the interconnection of mobile devices within the range of an individual person. typically within a range of 10 meters. PAN : 10 meters. Uses IEEE 802.15 standard (**Active**).

Internet:
The Internet is the global system of interconnected computer networks that use the Internet protocol suite (TCP/IP) to link devices worldwide. Internet is WAN hence covers Country, Continent or entire planet. Uses Packet Switching and networks are connected by routers.

Internetwork:

Connecting homogeneous or heterogeneous LAN and MAN to extend network reach. Uses IEEE 802 standards. It covers LAN/MAN architecture, internetworking among 802 LANs, MANs and wide area networks, 802 Link Security and 802 overall network management

Network Architectures:

Client – Server:

Server: Powerful (High End) machine consisting databases, applications, Internet Protocol servers (Blade Server) at central place. **Client:** Employee having low end machine to access the information.

Peer to Peer Network**:**

A network of computers configured to allow certain files and folders to be shared with everyone or with selected users. Peer-to-peer networks are quite common in small offices that do not use a dedicated file server.

Distributed Network (DN):

Distributed network is a distributed computing network system in which computer programming functionality and the data to be worked on are spread out across more than one computer. Usually, this is implemented over a computer network.

Software Defined Networking (SDN): Software-defined networking (SDN) technology is an approach to computer networking that allows network administrators to programmatically initialize, control, change, and manage network behavior dynamically via open interfaces and abstraction of lower-level functionality.

Infrastructure network:

Infrastructure mode supports central connection points for clients. An Infrastructure mode network requires the use of an Access Point. The Access Point controls Wireless communication and offers several important advantages over an Ad-hoc network. For example, a Infrastructure based network supports increased levels of security, potentially faster data transmission speeds and integration with a wired network.

Ad-hoc network:

An Ad-hoc network allows each device to communicate directly with each other. There is no central Access Point controlling device communication. Ad-hoc networks are only able to communicate with other Ad-hoc devices, they are not able to communicate with any Infrastructure devices or any other devices connected to a wired network. In addition, Ad-hoc mode security is less sophisticated compared to an Infrastructure mode.

IEEE 802 standards covers only LAN, MAN, WAN, SAN, ISDN and Wireless

802.1Internetworking	802.10 : Network Security	802.18 :Radio Regulatory TAG
802.2 :LLC	802.11 : WLAN, Wi-FI	802.19 :Coexistance TAG
802.3 :Ethernet	802.12 : 100VG - Any LAN	802.20 : Mobile Broadband
802.4 :Token Bus x	802.13 :	802.21 : Media Independent
802.5 :Token Ring x	802.14 : Cable Modem	Handoff
802.6 :MAN	802.15.2 : Bluetooth	802.22 : Wireless Regional Area
802.7 :Broadband	802.15.4 : WSN, ZigBee	Networks
802.8 :Fiber Optics	802.16 : Wi-MAX, WMAN	802.25 : Omni-Range Area
802.9 :ISDN	802.17 : Resilient packet ring	Network

TYPES OF CABLES

Unshielded Twisted Pair (UTP) Cable

Twisted pair cabling comes in two varieties: shielded and unshielded. Unshielded twisted pair (UTP) is the most popular and is generally the best option for school networks

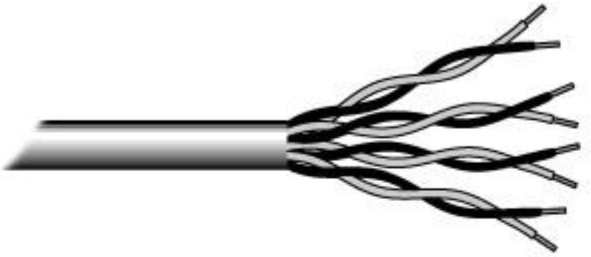


Fig.1. Unshielded twisted pair

The quality of UTP may vary from telephone-grade wire to extremely high-speed cable. The cable has four pairs of wires inside the jacket. Each pair is twisted with a different number of twists per inch to help eliminate interference from adjacent pairs and other electrical devices. The tighter the twisting, the higher the supported transmission rate and the greater the cost per foot. The EIA/TIA (Electronic Industry Association/Telecommunication Industry Association) has established standards of UTP and rated six categories of wire (additional categories are emerging).

Standards for Rating UTP Cable

- **Category 1 (CAT-1)** --For analog and digital voice (telephone) and low-speed data applications (Being used nowadays for telephone network only)

- **Category 2 (CAT-2)** --For voice, Integrated Services Digital Network (ISDN), and medium-speed data up to 4 Mbps. (obsolete)

- **Category 3 (CAT-3)** --For high-speed data and LAN traffic up to 16 Mbps (Used nowadays for Alarm Control Mechanisms in Apartments and Industry)

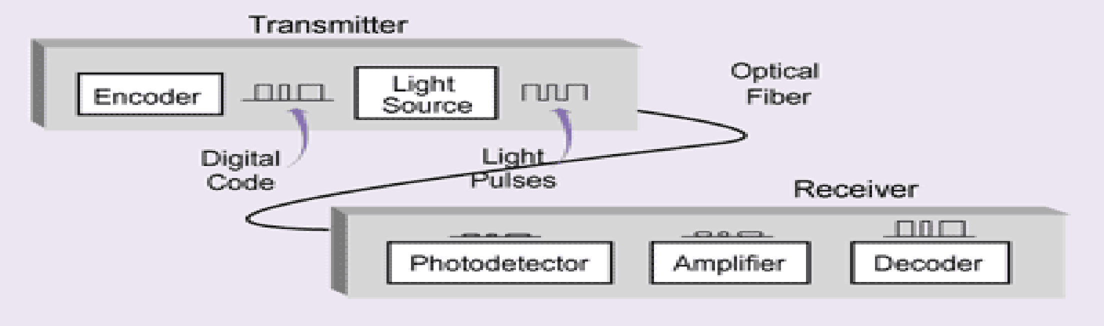
- **Category 4 (CAT-4)** --For long-distance LAN traffic up to 20 Mbps (obsolete)
- **Category 5 (CAT-5)** --For 100-Mbps LAN technologies such as 100-Mbps Ethernet over 100 MHz (Older LAN Standard. obsolete)
- **Category 5e (CAT-5e)** --Enhanced category 5 provides for full duplex Fast Ethernet support (used nowadays) with 1 Gbps upto 100 meters. Frequency 100 and 250 MHz.
- **Category 6 (CAT-6)** – 10 Gbps up to 164 feet 50/55 meters— anything beyond that will rapidly decay to only 1 Gigabit (the same as Cat5E). Frequency 350 MHz
- **Category 6a (CAT-6a)** -- 10 Gigabit speeds for the full 328 feet of Ethernet cable. Frequency 550 MHz.
- **Category 7 (CAT-7)** -- The maximum allowed length of a Cat-6 cable is 100 meters (330 ft) when used for 10/100/1000baseT and 55 meters (180 ft) when used for 10GbaseT. Frequency 600 MHz.
- Higher category UTP cables are made from higher quality materials.
- Each higher category is also made with tighter cable twists for increased resistance to interference

Fiber Optic Cable

- A fiber optic cable is a thin strand of glass or plastic, coated with a protective plastic jacket. It is so thin that even the glass fibers bend easily.
- A beam of light can be trapped within a fiber, so that the optical cable essentially becomes a pipe that carries light around corners. An optical fiber can carry a light signal for a long distance typically up to 2 km.
- Because light is not appreciably affected by electromagnetic fields, optical signals are immune to EMI/RFI. This makes fiber a good choice for "noisy" environments with many electrical motors, such as elevator shafts and factories.
- Because fiber does not corrode, it is well suited for high-humidity and underwater environments. Optical fiber is also a highly secure medium, because it is difficult to splice (cannot flow) into a fiber optic cable without detection.
- The primary disadvantage of fiber optic cable is its cost. Fiber optic cable and equipment are relatively expensive in terms of both materials cost and installation.
- However, industries that need the high capacity and secure features of fiber find it well worth the investment. For example, nearly all long-distance telecommunication lines are fiber optic.

Fiber Communication System

- The basic model for a communication system includes a transmitter and receiver, connected by optical fiber cabling.
- In typical fiber optic systems, each device contains both a transmitter and receiver, combined in a single transceiver unit.
- Because fiber optic cable must be cut to present the light beam to a receiver, only point-to-point connections can be made; a bus cannot be constructed

- 

Transmitter

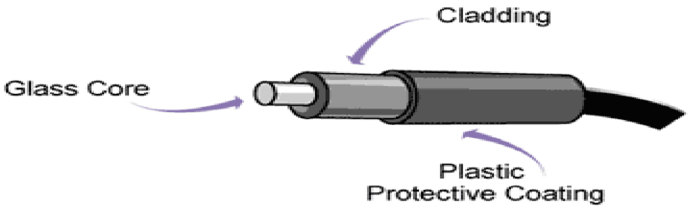
- Encoder that converts the input data signal into digital electrical pulses
- Light source that converts the digital electrical signal to light pulses
- Connector that couples the light source to the fiber through which the light rays travel
- The transmitter accepts digital electrical signals from a computer.
- A diode converts the digital code into a pattern of light pulses that are sent out to the receiver through the optical fiber.
- Light emitting diodes (LEDs) use less power and are considerably less expensive than lasers. LEDs can be used with multimode cable, and are the most common light source. LEDs provide a bandwidth of approximately 250MHz .
- Laser diodes are used with single-mode fiber for long-distance transmission. Laser light is more powerful because laser light waves are radiated in phase, which means the crests(H-peak) and troughs(L-peak) of all light waves are perfectly aligned with one another. This alignment or coherence creates a signal with much less attenuation and dispersion than noncoherent light. Laser diodes can provide much higher bandwidth 10 GHz.
- Frequency Range : 250 MHz to 10 GHz

Receiver

- A receiver converts the modulated light pulses back to electrical signals and decodes them. The receiver, contained within the destination computer system, includes:
    - Photo detector that converts the light pulses into electric signals
    - Amplifier, if needed
    - Message decoder

WARNING:

- Never look into a fiber optic cable to see whether light is present. The infrared laser light used in fiber optic LANs is invisible; however, it can permanently damage your eyesight in an instant.



- Core--A solid fiber of highly refractive clear glass or plastic that serves as the central conduit for light.

- Cladding--A layer of clear glass or plastic with a lower index of refraction. When light traveling down the core reaches the boundary between the core and cladding, the change in refractive index causes the light to completely refract or bend back into the core. The cladding of each fiber completely contains light signals within each core, preventing crosstalk. This effect is called "total internal reflection."

- Coating--A reinforced plastic outer jacket that protects the cable from damage.

Dimensions (Diameter)

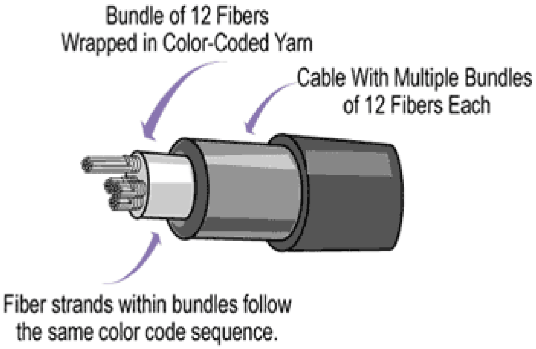
- Fiber optic cable is very thin. The diameters of fiber optic cores and cladding are specified in µm. The thinnest fiber optic cable (single-mode) typically has a core diameter of 5 to 10 µm I.e. 0.005 to 0.010 mm. Thicker fiber optic cable (multimode) ranges from 50 to 100 µm in core diameter. In comparison, human hair is approximately 100 µm thick.

- Fiber optic cable is specified in terms of its core and cladding diameter. For example, the most common type of fiber optic cable for LAN installations is 62.5/125-m cable, where 62.5 refers to the core diameter and 125 refers to the cladding diameter.

- The core diameter is also known as the aperture, because it determines the maximum angle from which the cable can accept light. Total internal reflection only occurs when light strikes the cladding at a shallow angle. If the angle is too steep, some or all of the light will penetrate the cladding itself, causing signal loss.

- Each fiber optic core conducts light in one direction only. Therefore, to send and receive, devices are usually connected by two fiber optic strands. These may be single strand simplex cables, or duplex cables containing two fiber optic strands. Duplex cables are more commonly used than simplex cables.

- Fiber cables can also consist of several bundles. These are used for high-capacity backbones for outdoor connections between campus buildings. Because light signals are completely contained within each fiber, no coating or shielding is necessary between fibers. However, reinforcing strands are usually added to increase the pulling strength of the cable.

- 
Bundle of 12 Fibers
Wrapped in Color-Coded Yarn
Cable With Multiple Bundles
of 12 Fibers Each
Fiber strands within bundles follow
the same color code sequence.

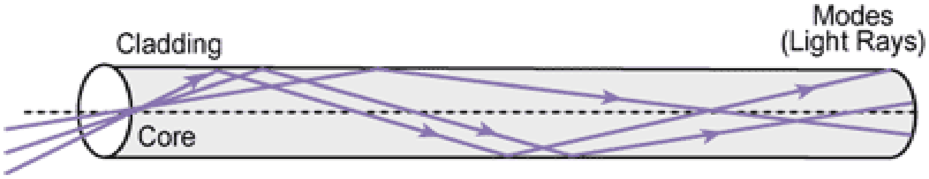
FOC General Types

- Multimode fiber is wide enough to carry more than one light signal. Each signal is called a "mode."

- Single-mode fiber is thin and can carry only one light signal.

Multimode Fiber

- Each light signal or light ray that passes through a cable is called a "mode." Multimode fiber optic cable is wider than single-mode cable, thus it has enough room for more than one light ray. These light signals are separated by different angles of reflection as they travel down the core.
- Because multimode signaling separates light signals by angle, not all light rays travel the same distance. Some light rays will travel nearly straight through the core, while others bounce off the cladding many times before reaching the far end of the fiber.
- With modes traveling different distances, but at the same speed, the spread of the signal increases over time, and can cause data errors due to the overlapping of light pulses. This problem is known as modal dispersion. The construction of a multimode fiber can either cause or fix this problem.



Types of multimode fiber

1. Step-Index Fiber

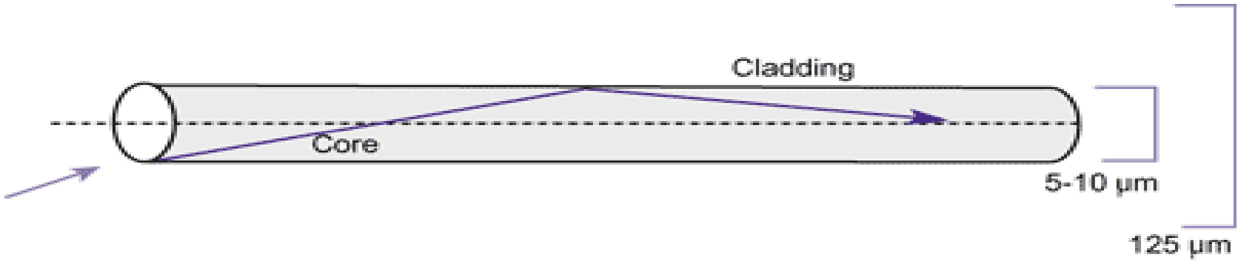
The standard type of optical fiber, called "step-index fiber", consists of only two transparent layers (core and cladding), and index of refraction is same.

2. Graded-Index Fiber

The core of a graded-index fiber cable has several transparent layers, each with a different refractive index. This planned inconsistency allows light modes to travel at different speeds through the core. The speed at which the modes travel depends upon the part of the core it is traveling through. Modes traveling down the center of the core do so at a slower speed than those refracting off the cladding.

Single Mode Fiber

- Single-mode fibers have diameters sized to the wavelength they are designed to carry. A typical single-mode fiber core diameter is 8 µm. Only one mode will propagate through fiber with this core diameter. The narrower fiber diameter causes a light signal to travel in a straighter path, with less reflection and dispersion. However, the narrower core also makes single-mode fiber more difficult and expensive to install.
- Single-mode fibers require laser diode transmitters. By using this coherent light source, single-mode fiber optic cable can support longer transmission distances than multimode fiber. Distances range from a few miles to as many as 20 miles.
- Fiber optic cable is difficult to install correctly; therefore, it requires well-trained, careful installation technicians. This, combined with the time-consuming nature of each connection, make fiber optic cable the most expensive cable to install. Because of this need for training and experience, many organizations hire specialists to install fiber optic networks.
- Connections and splices of fiber optic cable are particularly difficult to make. Each end of the cable must be cut off at perfect right angles, the ends polished by hand or machine, and the cable precisely aligned to the connector.
- Single-mode fibers are generally step-index fibers. Because only one mode travels along the fiber, the problem of diffusion does not occur in single-mode fibers.



Optical Carrier Levels: OC-n

Optical Carrier Level	**Data Rate**
OC-1	51.84 Mbps
OC-3	155.52 Mbps
OC-12	622.08 Mbps
OC-24	1.244 Gbps
OC-48	2.488 Gbps
OC-192	10 Gbps
OC-256	13.271 Gbps
OC-768	40 Gbps

What do the fiber terms 9/125, 50/125 and 62.5/125 refer to?

The first set of numbers - 9, 50 and 62.5 refer to the diameter of the fiber cable's core. The second set of numbers - 125 refer to the diameter of the outside of the fiber cable's cladding. The cladding is a special coating that keeps the light from escaping the glass core. 9/125 refers to a single mode fiber cable. 50/125 and 62.5/125 refer to multimode fiber cable

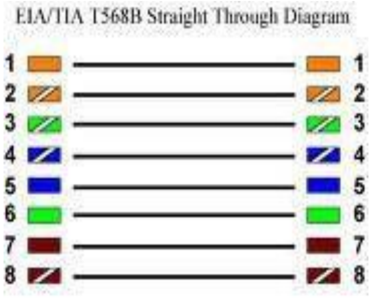
1a) We setup the network using 2 ways – wired connection and wireless connection. In wired mode, we connect 2 laptops/desktops via a hub/switch. This consisted of connecting the Ethernet ports of the laptop/desktop to the hub/switch and then modify the network settings in the control panel.

Cable Preparation:

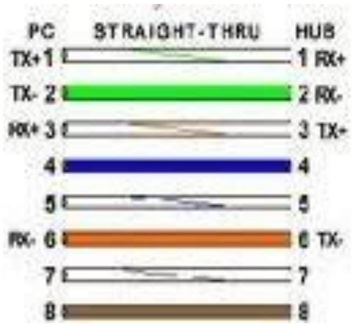
There are generally three main types of networking cables: straight-through, crossover, and rollover cables. Each cable type has a distinct use, and should not be used in place of another. So how do you know which cable to use for what you need?

The Purpose of Straight-Through Cables

Straight-through cables get their name from how they are made. Out of the 8 pins that exist on both ends of an Ethernet cable, each pin connects to the same pin on the opposite side. Review the diagram below for a visual example:



OR



Notice how each wire corresponds to the same pin. This kind of wiring diagram is part of the 568A standard. The 568B standard achieves the same thing, but through different wiring. It is generally accepted to use the 568A standard as pictured, since it allows compatibility with certain telephone hardware- while 568B doesn't.

Straight-through cables are primarily used for connecting unlike devices. A straight-through cable is typically used in the following situations:

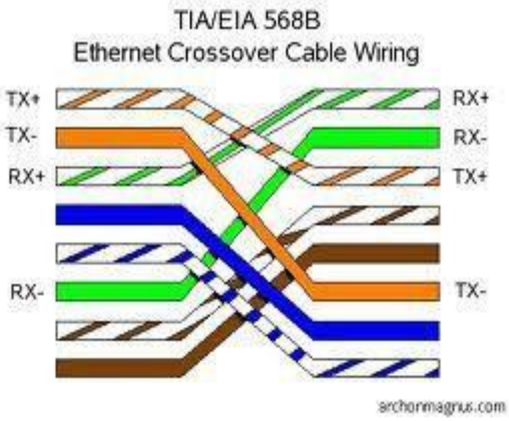
Use a straight-through cable when:

- Connecting a router to a hub
- Connecting a computer to a swtich
- Connecting a LAN port to a switch, hub, or computer

Note that some devices such as routers will have advanced circuitry, which enables them to use both crossover and straight-through cables. In general, however, straight-through cables will not connect a computer and router because they are not "unlike devices."

The purpose of Crossover Cables

Crossover cables are very similar to straight-through cables, except that they have pairs of wires that crisscross. This allows for two devices to communicate at the same time. Unlike straight-through cables, we use crossover cables to connect like devices. A visual example can be seen below:



Notice how all we did was switch the orange-white and green-white wires, and then the orange and green wires. This will enable like devices to communicate. Crossover cables are typically used in the following situations:

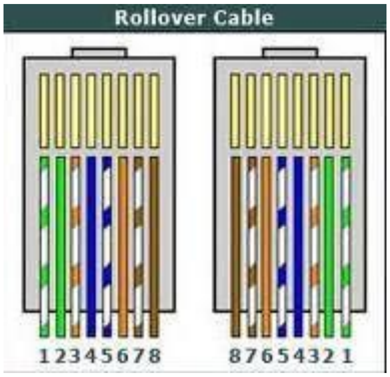
Use a crossover cable when:

- Connecting a computer to a router
- Connecting a computer to a computer
- Connecting a router to a router
- Connecting a switch to a switch
- Connecting a hub to a hub

While the rule of thumb is to use crossover cables with like devices, some devices do not follow standards. Others provide support for both types of cables. However, there is still something that both crossover and straight-through cables can’t do.

The Purpose of Rollover Cables

Rollover cables, like other cabling types, got their name from how they are wired. Rollover cables essentially have one end of the cable wired exactly opposite from the other. This essentially “rolls over” the wires- but why would we need to do such a thing? Rollover cables, also called Yost cables, usually connect a device to a router or switch’s console port. This allows a programmer to make a connection to the router or switch, and program it as needed. A visual example can be seen below:



Notice that each wire is simply “rolled over.” These types of cables are generally not used very much, so are usually colored differently from other types of cables.

1b) Wi-Fi setup Procedure

In the wireless connection mode, we setup the router or the access point first using theprovided instructio guidelines. Following is the procedure to configure Wi-Fi network.

Wi-Fi setup

Step 1: connect router to internet using yellow color port.

Step2: connect your laptop/ desktop by using non yellow port.

Step3: configure your desktop for DHCP .

Step 4: access the router using www.routerlogin.net.

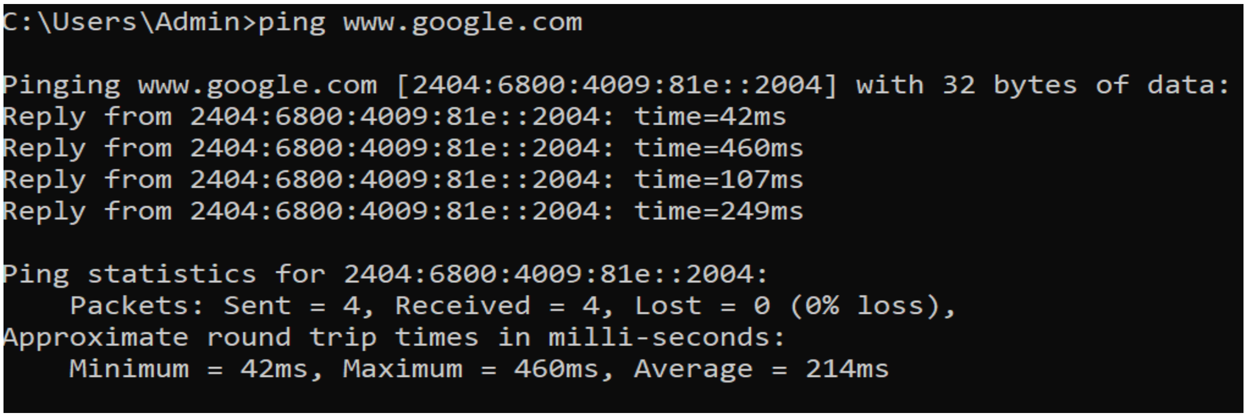
Step 5 : configure basic screen using internet option for IP credentials of desktop.

Step 6: access the internet through browser using 172.16…..access control of VIT

1c) Network Commands:

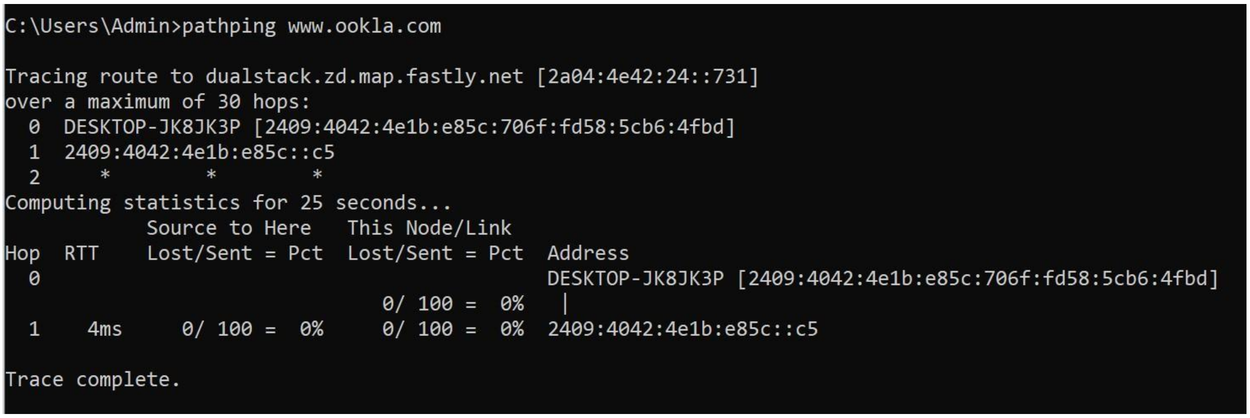
Ping command:

Ping is a computer network administration software utility used to test their reachability of a host on an Internet Protocol network.



```
C:\Users\Admin>ping www.google.com

Pinging www.google.com [2404:6800:4009:81e::2004] with 32 bytes of data:
Reply from 2404:6800:4009:81e::2004: time=42ms
Reply from 2404:6800:4009:81e::2004: time=460ms
Reply from 2404:6800:4009:81e::2004: time=107ms
Reply from 2404:6800:4009:81e::2004: time=249ms

Ping statistics for 2404:6800:4009:81e::2004:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 42ms, Maximum = 460ms, Average = 214ms
```

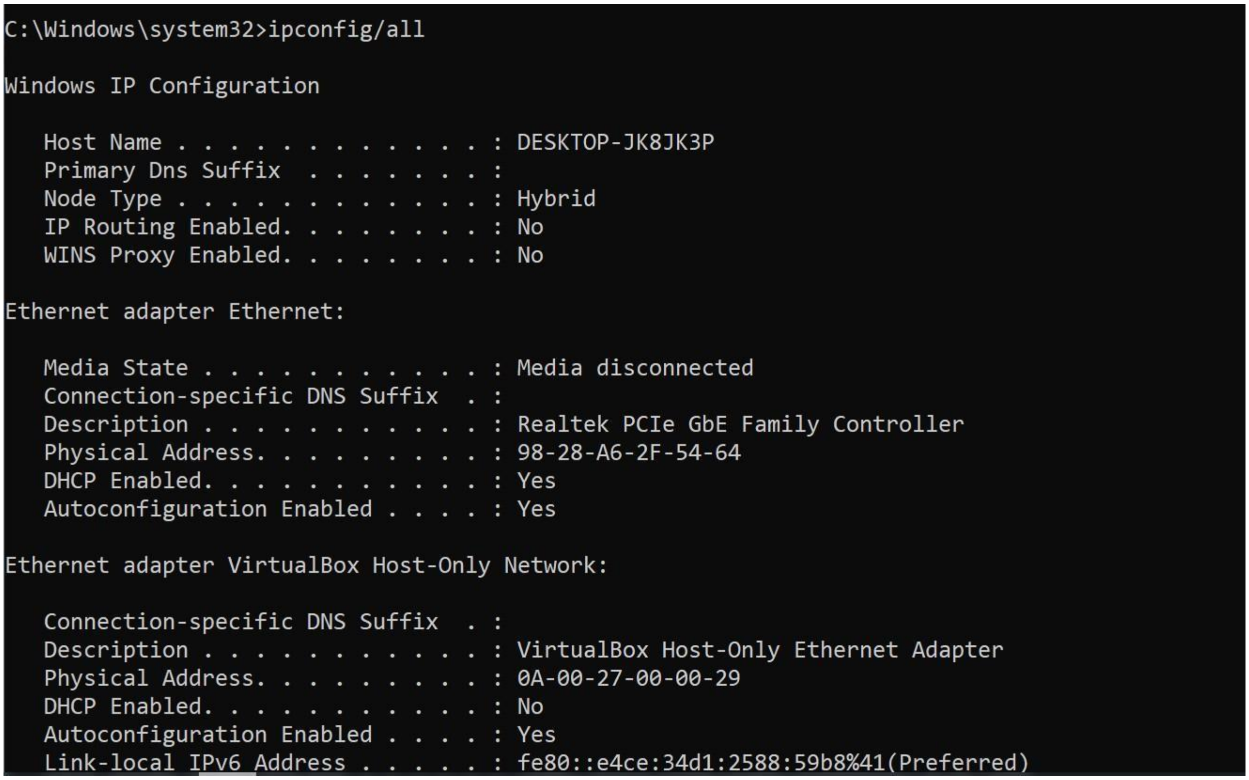
Pathping command: The PathPing command is a command-line network utility supplied in Windows2000 andbeyond that combines the functionality of ping with that of tracer



```
C:\Users\Admin>pathping www.ookla.com

Tracing route to dualstack.zd.map.fastly.net [2a04:4e42:24::731]
over a maximum of 30 hops:
  0  DESKTOP-JK8JK3P [2409:4042:4e1b:e85c:706f:fd58:5cb6:4fbd]
  1  2409:4042:4e1b:e85c::c5
  2     *        *        *
Computing statistics for 25 seconds...
            Source to Here   This Node/Link
Hop  RTT    Lost/Sent = Pct  Lost/Sent = Pct  Address
  0                                           DESKTOP-JK8JK3P [2409:4042:4e1b:e85c:706f:fd58:5cb6:4fbd]
                                0/ 100 =  0%   |
  1    4ms     0/ 100 =  0%     0/ 100 =  0%  2409:4042:4e1b:e85c::c5

Trace complete.
```

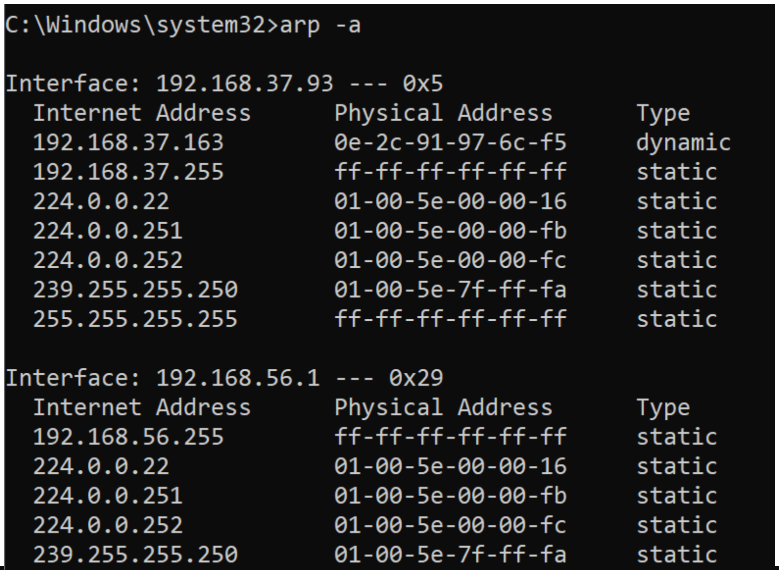
ipconfig command:

The ipconfig command, which displays all current TCP/IP networkconfiguration values.

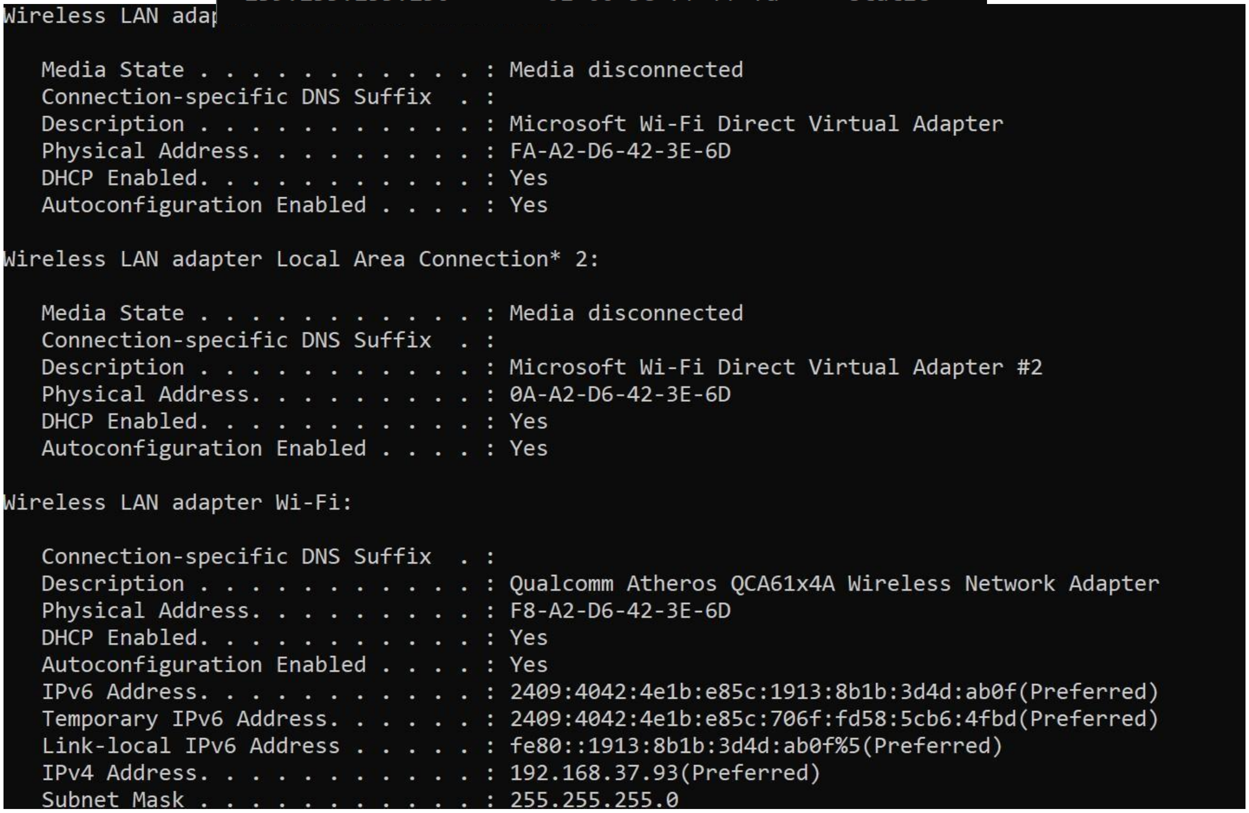


```
C:\Windows\system32>ipconfig/all

Windows IP Configuration

   Host Name . . . . . . . . . . . . : DESKTOP-JK8JK3P
   Primary Dns Suffix  . . . . . . . :
   Node Type . . . . . . . . . . . . : Hybrid
   IP Routing Enabled. . . . . . . . : No
   WINS Proxy Enabled. . . . . . . . : No

Ethernet adapter Ethernet:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Realtek PCIe GbE Family Controller
   Physical Address. . . . . . . . . : 98-28-A6-2F-54-64
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes

Ethernet adapter VirtualBox Host-Only Network:

   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : VirtualBox Host-Only Ethernet Adapter
   Physical Address. . . . . . . . . : 0A-00-27-00-00-29
   DHCP Enabled. . . . . . . . . . . : No
   Autoconfiguration Enabled . . . . : Yes
   Link-local IPv6 Address . . . . . : fe80::e4ce:34d1:2588:59b8%41(Preferred)
```

arp command : The arp command displays and modifies the Internet-to-adapter address translation tables used by the Address in Networks and communication management. The arp command displays the current ARP entry for the host specified by the Hostname variable.

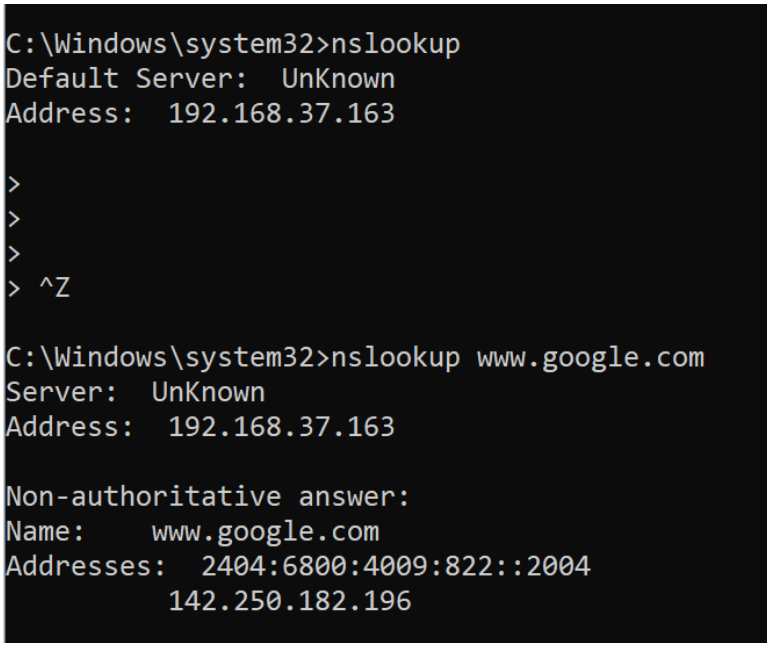


```
C:\Windows\system32>arp -a

Interface: 192.168.37.93 --- 0x5
  Internet Address      Physical Address      Type
  192.168.37.163        0e-2c-91-97-6c-f5     dynamic
  192.168.37.255        ff-ff-ff-ff-ff-ff     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static

Interface: 192.168.56.1 --- 0x29
  Internet Address      Physical Address      Type
  192.168.56.255        ff-ff-ff-ff-ff-ff     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
```



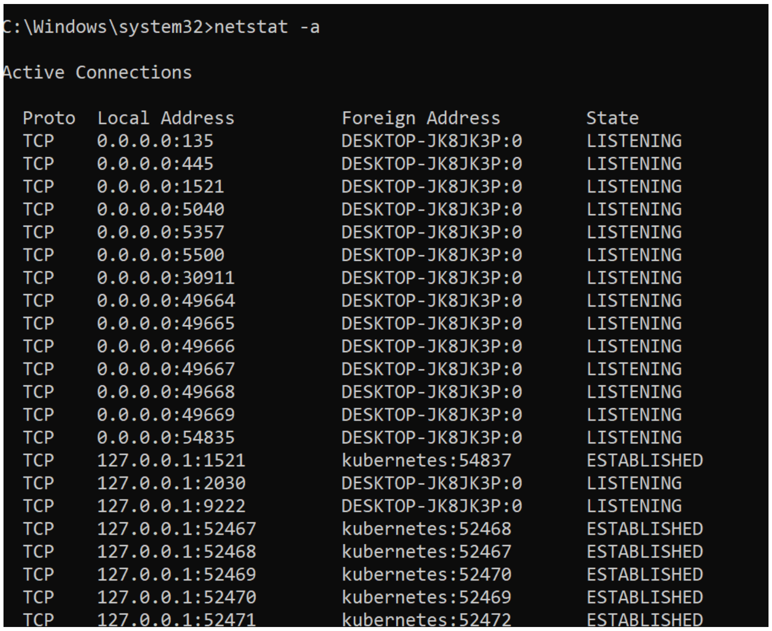
```
Wireless LAN adap

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Microsoft Wi-Fi Direct Virtual Adapter
   Physical Address. . . . . . . . . : FA-A2-D6-42-3E-6D
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes

Wireless LAN adapter Local Area Connection* 2:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Microsoft Wi-Fi Direct Virtual Adapter #2
   Physical Address. . . . . . . . . : 0A-A2-D6-42-3E-6D
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes

Wireless LAN adapter Wi-Fi:

   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Qualcomm Atheros QCA61x4A Wireless Network Adapter
   Physical Address. . . . . . . . . : F8-A2-D6-42-3E-6D
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes
   IPv6 Address. . . . . . . . . . . : 2409:4042:4e1b:e85c:1913:8b1b:3d4d:ab0f(Preferred)
   Temporary IPv6 Address. . . . . . : 2409:4042:4e1b:e85c:706f:fd58:5cb6:4fbd(Preferred)
   Link-local IPv6 Address . . . . . : fe80::1913:8b1b:3d4d:ab0f%5(Preferred)
   IPv4 Address. . . . . . . . . . . : 192.168.37.93(Preferred)
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
```

nslookup command: Nslookup (stands for “Name Server Lookup”) is a useful command for getting information from the DNS server. It is a network administration toolfor querying the DomainName System (DNS) to obtain domain name or IP address mapping or any other specific DNSrecord. It is also used to troubleshoot DNS-related problems.



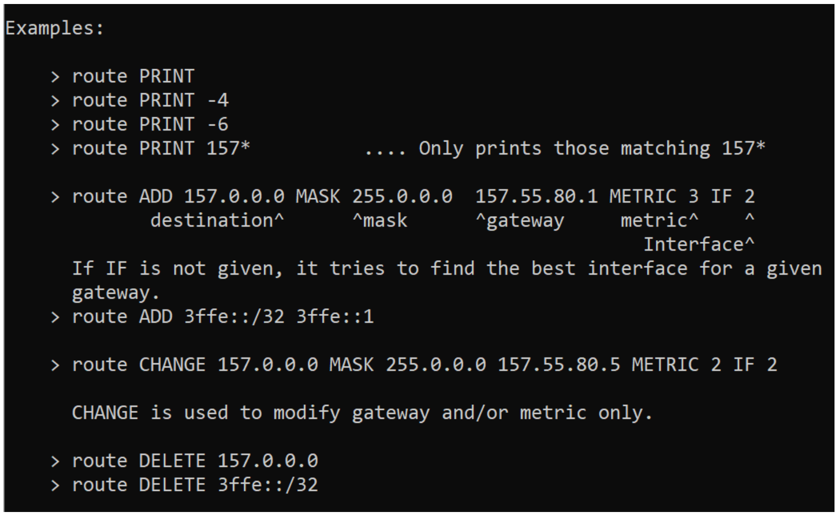
```
C:\Windows\system32>nslookup
Default Server:  UnKnown
Address:  192.168.37.163

>
>
>
> ^Z

C:\Windows\system32>nslookup www.google.com
Server:  UnKnown
Address:  192.168.37.163

Non-authoritative answer:
Name:    www.google.com
Addresses:  2404:6800:4009:822::2004
          142.250.182.196
```

netstat command : Netstat command displays various network related information such as network connections, routing tables, interface statistics, masquerade connections, multicast memberships etc.,



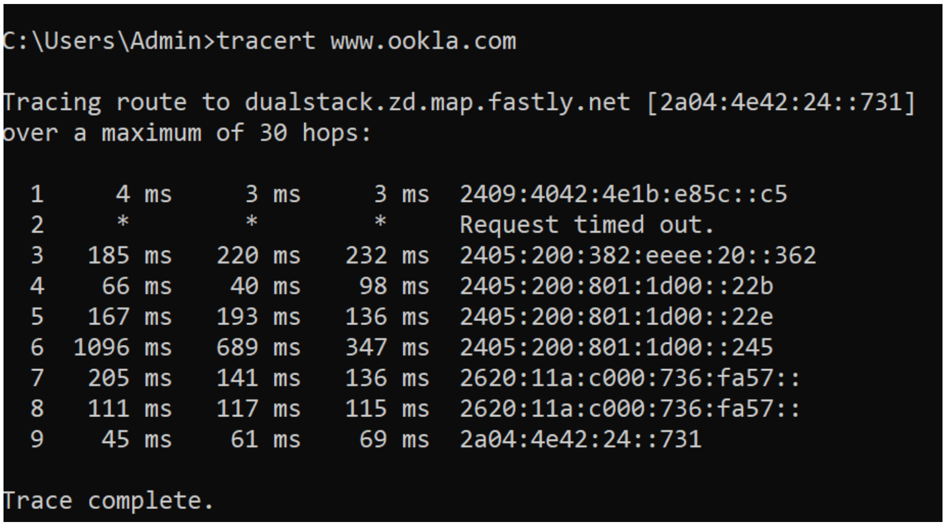
```
C:\Windows\system32>netstat -a

Active Connections

  Proto  Local Address          Foreign Address        State
  TCP    0.0.0.0:135            DESKTOP-JK8JK3P:0      LISTENING
  TCP    0.0.0.0:445            DESKTOP-JK8JK3P:0      LISTENING
  TCP    0.0.0.0:1521           DESKTOP-JK8JK3P:0      LISTENING
  TCP    0.0.0.0:5040           DESKTOP-JK8JK3P:0      LISTENING
  TCP    0.0.0.0:5357           DESKTOP-JK8JK3P:0      LISTENING
  TCP    0.0.0.0:5500           DESKTOP-JK8JK3P:0      LISTENING
  TCP    0.0.0.0:30911          DESKTOP-JK8JK3P:0      LISTENING
  TCP    0.0.0.0:49664          DESKTOP-JK8JK3P:0      LISTENING
  TCP    0.0.0.0:49665          DESKTOP-JK8JK3P:0      LISTENING
  TCP    0.0.0.0:49666          DESKTOP-JK8JK3P:0      LISTENING
  TCP    0.0.0.0:49667          DESKTOP-JK8JK3P:0      LISTENING
  TCP    0.0.0.0:49668          DESKTOP-JK8JK3P:0      LISTENING
  TCP    0.0.0.0:49669          DESKTOP-JK8JK3P:0      LISTENING
  TCP    0.0.0.0:54835          DESKTOP-JK8JK3P:0      LISTENING
  TCP    127.0.0.1:1521         kubernetes:54837       ESTABLISHED
  TCP    127.0.0.1:2030         DESKTOP-JK8JK3P:0      LISTENING
  TCP    127.0.0.1:9222         DESKTOP-JK8JK3P:0      LISTENING
  TCP    127.0.0.1:52467        kubernetes:52468       ESTABLISHED
  TCP    127.0.0.1:52468        kubernetes:52467       ESTABLISHED
  TCP    127.0.0.1:52469        kubernetes:52470       ESTABLISHED
  TCP    127.0.0.1:52470        kubernetes:52469       ESTABLISHED
  TCP    127.0.0.1:52471        kubernetes:52472       ESTABLISHED
```

route command : The route command allows you to make manual entries into the network routing tables.The route command distinguishes between routes to hosts and routes to networks by interpreting the network address of the Destination variable, which can be specified either bysymbolic name or numeric address.



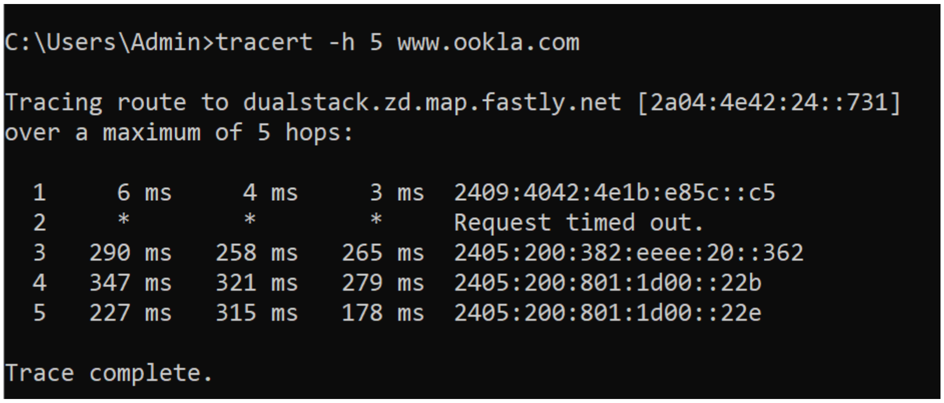
```
Examples:

    > route PRINT
    > route PRINT -4
    > route PRINT -6
    > route PRINT 157*          .... Only prints those matching 157*

    > route ADD 157.0.0.0 MASK 255.0.0.0  157.55.80.1 METRIC 3 IF 2
             destination^      ^mask      ^gateway     metric^    ^
                                                         Interface^
      If IF is not given, it tries to find the best interface for a given
      gateway.
    > route ADD 3ffe::/32 3ffe::1

    > route CHANGE 157.0.0.0 MASK 255.0.0.0 157.55.80.5 METRIC 2 IF 2

      CHANGE is used to modify gateway and/or metric only.

    > route DELETE 157.0.0.0
    > route DELETE 3ffe::/32
```

tracert command : The traceroute command is used to determine the path between two connections. Often a connection to another device will have to go through multiple routers. The traceroute command will return the names or IP addressesof all the routers between two devices.



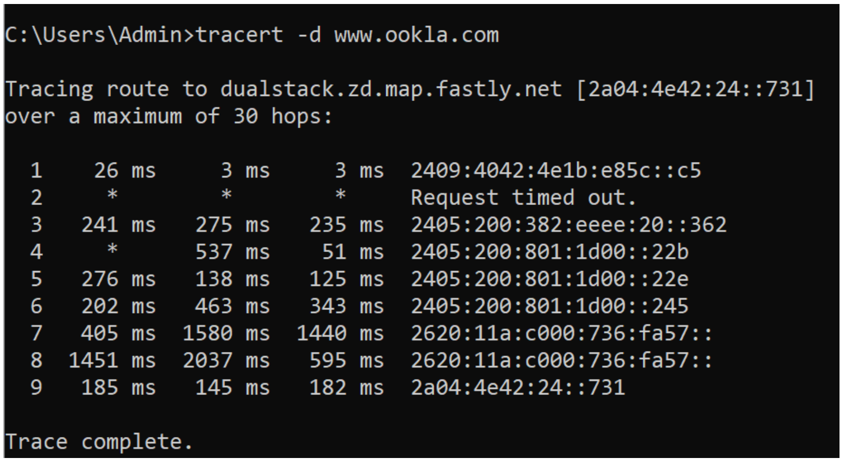
```
C:\Users\Admin>tracert www.ookla.com

Tracing route to dualstack.zd.map.fastly.net [2a04:4e42:24::731]
over a maximum of 30 hops:

  1     4 ms     3 ms     3 ms  2409:4042:4e1b:e85c::c5
  2     *        *        *     Request timed out.
  3   185 ms   220 ms   232 ms  2405:200:382:eeee:20::362
  4    66 ms    40 ms    98 ms  2405:200:801:1d00::22b
  5   167 ms   193 ms   136 ms  2405:200:801:1d00::22e
  6  1096 ms   689 ms   347 ms  2405:200:801:1d00::245
  7   205 ms   141 ms   136 ms  2620:11a:c000:736:fa57::
  8   111 ms   117 ms   115 ms  2620:11a:c000:736:fa57::
  9    45 ms    61 ms    69 ms  2a04:4e42:24::731

Trace complete.
```

-h:

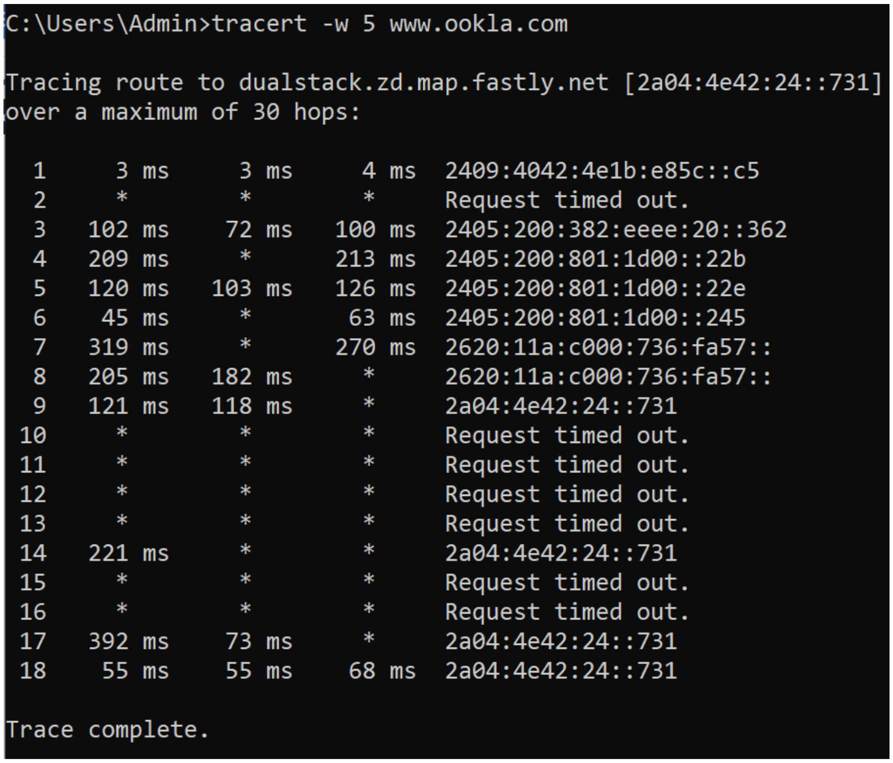


```
C:\Users\Admin>tracert -h 5 www.ookla.com

Tracing route to dualstack.zd.map.fastly.net [2a04:4e42:24::731]
over a maximum of 5 hops:

  1     6 ms     4 ms     3 ms  2409:4042:4e1b:e85c::c5
  2     *        *        *     Request timed out.
  3   290 ms   258 ms   265 ms  2405:200:382:eeee:20::362
  4   347 ms   321 ms   279 ms  2405:200:801:1d00::22b
  5   227 ms   315 ms   178 ms  2405:200:801:1d00::22e

Trace complete.
```

-d:



```
C:\Users\Admin>tracert -d www.ookla.com

Tracing route to dualstack.zd.map.fastly.net [2a04:4e42:24::731]
over a maximum of 30 hops:

  1    26 ms     3 ms     3 ms  2409:4042:4e1b:e85c::c5
  2     *        *        *     Request timed out.
  3   241 ms   275 ms   235 ms  2405:200:382:eeee:20::362
  4     *      537 ms    51 ms  2405:200:801:1d00::22b
  5   276 ms   138 ms   125 ms  2405:200:801:1d00::22e
  6   202 ms   463 ms   343 ms  2405:200:801:1d00::245
  7   405 ms  1580 ms  1440 ms  2620:11a:c000:736:fa57::
  8  1451 ms  2037 ms   595 ms  2620:11a:c000:736:fa57::
  9   185 ms   145 ms   182 ms  2a04:4e42:24::731

Trace complete.
```



```
C:\Users\Admin>tracert -w 5 www.ookla.com

Tracing route to dualstack.zd.map.fastly.net [2a04:4e42:24::731]
over a maximum of 30 hops:

  1     3 ms     3 ms     4 ms  2409:4042:4e1b:e85c::c5
  2     *        *        *     Request timed out.
  3   102 ms    72 ms   100 ms  2405:200:382:eeee:20::362
  4   209 ms     *      213 ms  2405:200:801:1d00::22b
  5   120 ms   103 ms   126 ms  2405:200:801:1d00::22e
  6    45 ms     *       63 ms  2405:200:801:1d00::245
  7   319 ms     *      270 ms  2620:11a:c000:736:fa57::
  8   205 ms   182 ms     *     2620:11a:c000:736:fa57::
  9   121 ms   118 ms     *     2a04:4e42:24::731
 10     *        *        *     Request timed out.
 11     *        *        *     Request timed out.
 12     *        *        *     Request timed out.
 13     *        *        *     Request timed out.
 14   221 ms     *        *     2a04:4e42:24::731
 15     *        *        *     Request timed out.
 16     *        *        *     Request timed out.
 17   392 ms    73 ms     *     2a04:4e42:24::731
 18    55 ms    55 ms    68 ms  2a04:4e42:24::731

Trace complete.
```

Experiment Number: 02

TITLE: Routing in the Internet

PROBLEM STATEMENT

Write a program to find the shortest path using Dijkstra Equation for Link State Routing Protocol which is used by Open Shortest Path First Protocol (OSPF) in the Internet for the network flow provided by instructor.

Theory:

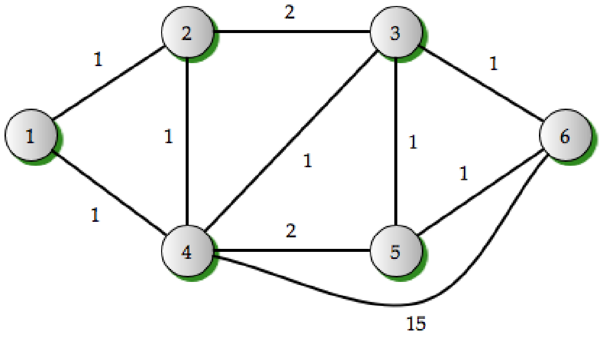
The Link State Routing Protocol is a type of routing protocol used in computer networks to determine the shortest path from one node to another. It is based on the concept of each router or node in the network maintaining a database of its neighbors and the cost or distance to reach them. Dijkstra's algorithm is commonly employed to compute the shortest path in Link State Routing Protocol. Below is a write-up explaining how Dijkstra's algorithm is used to find the shortest path in a network.

Overview of Dijkstra's Algorithm:

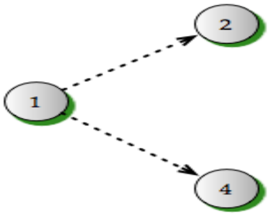
Dijkstra's algorithm is a graph search algorithm that efficiently finds the shortest path from a source node to all other nodes in a weighted graph. It starts by initializing distances to all nodes as infinity, except for the source node, which is set to zero. Then, it iteratively selects the node with the smallest distance from the source and updates the distances to its neighboring nodes. The process continues until all nodes have been visited.

Steps to Find the Shortest Path:

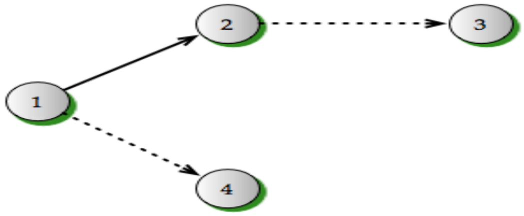
- **Initialization**: Initialize a distance vector for each node, setting the distance to the source node as zero and all other distances as infinity. Also, maintain a set of unvisited nodes.
- **Selecting the Nearest Node**: At each iteration, select the node with the smallest distance from the source node among the unvisited nodes.
- **Updating Distances**: For the selected node, update the distances to its neighboring nodes if the distance through the selected node is shorter than the current distance. This involves comparing the sum of the distance to the selected node and the distance from the selected node to its neighbor with the current distance to the neighbor.
- **Marking Visited Nodes**: Mark the selected node as visited and remove it from the set of unvisited nodes.
- **Repeat**: Repeat steps 2 to 4 until all nodes have been visited.
- **Shortest Path Tree**: Once the algorithm is complete, a shortest path tree is constructed, with the source node as the root and edges representing the shortest paths to each node.



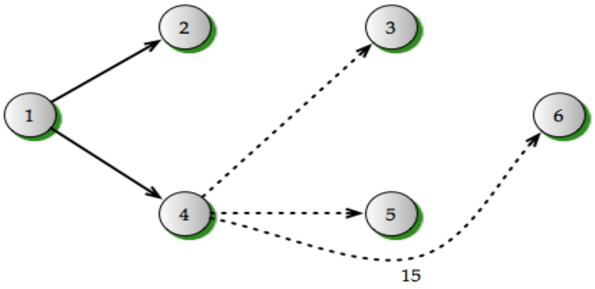
S = { 1 }



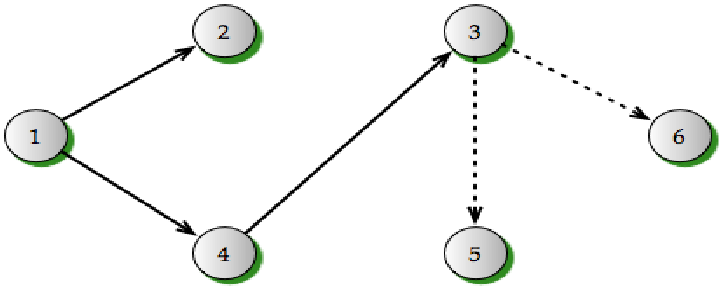
S = { 1, 2 }



S = { 1, 2, 4 }



S = { 1, 2, 4, 3 }



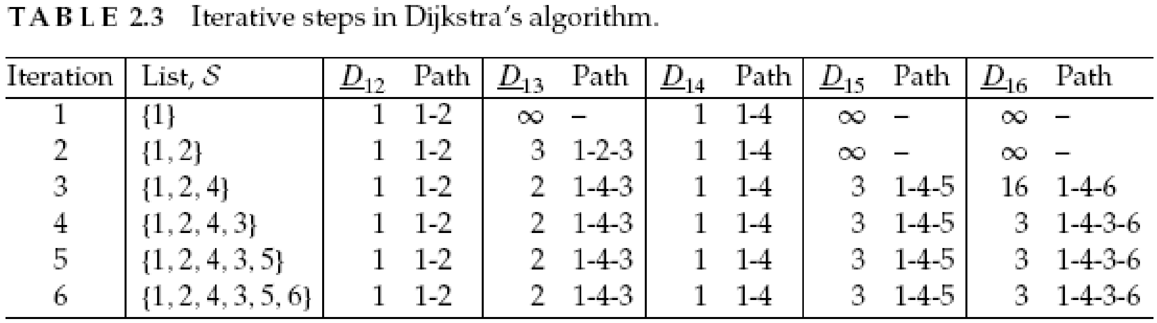


TABLE 2.3 Iterative steps in Dijkstra's algorithm.

Iteration	List, S	$\underline{D}_{12}$	Path	$\underline{D}_{13}$	Path	$\underline{D}_{14}$	Path	$\underline{D}_{15}$	Path	$\underline{D}_{16}$	Path
1	{1}	1	1-2	∞	–	1	1-4	∞	–	∞	–
2	{1, 2}	1	1-2	3	1-2-3	1	1-4	∞	–	∞	–
3	{1, 2, 4}	1	1-2	2	1-4-3	1	1-4	3	1-4-5	16	1-4-6
4	{1, 2, 4, 3}	1	1-2	2	1-4-3	1	1-4	3	1-4-5	3	1-4-3-6
5	{1, 2, 4, 3, 5}	1	1-2	2	1-4-3	1	1-4	3	1-4-5	3	1-4-3-6
6	{1, 2, 4, 3, 5, 6}	1	1-2	2	1-4-3	1	1-4	3	1-4-5	3	1-4-3-6

Application in Link State Routing Protocol:

In Link State Routing Protocol, each router exchanges information about its directly connected links with other routers in the network. This information includes the link cost or distance. Using this information, Dijkstra's algorithm is applied at each router to compute the shortest path to all other routers in the network.

Conclusion:

Dijkstra's algorithm plays a crucial role in the Link State Routing Protocol by enabling routers to compute the shortest paths efficiently. By maintaining a database of network topology information and applying Dijkstra's algorithm, routers can make informed routing decisions, leading to efficient and reliable communication within the network.

Experiment Number: 03

TITLE: Socket programming

OBJECTIVES:

1. To understand system socket call in Linux for TCP programming.
2. Use UNIX/LINUX socket programming to exchange data between two machines.

PROBLEM STATEMENT

3a) Write the client server programs using TCP Berkeley socket primitives for wired /wireless network for following
a. To say Hello to Each other
b. File transfer

3b) Write the client server programs using UDP Berkeley socket primitives for wired /wireless network for following
a. To say Hello to Each other
b. Calculator (Trigonometry)

3c) Understanding protocol stack of Intranet
Analyze packet formats of Ethernet, IP, TCP and UDP captured through Wireshark for wired networks.

LINUX SOCKET PROGRAMMING

The Berkeley socket interface, an API, allows communications between hosts or between processes on one computer, using the concept of a socket. It can work with many different I/O devices and drivers, although support for these depends on the operating- system implementation. This interface implementation is implicit for TCP/IP, and it is therefore one of the fundamental technologies underlying the Internet. It was first developed at the University of California, Berkeley for use on UNIX systems. All modern operating systems now have some implementation of the Berkeley socket interface, as it has become the standard interface for connecting to the Internet.

Programmers can make the socket interfaces accessible at three different levels, most powerfully and fundamentally at the RAW socket level. Very few applications need the degree of control over outgoing communications that this provides, so RAW sockets support was intended to be available only on computers used for developing Internet- related technologies. In recent years, most operating systems have implemented support for it anyway, including Windows XP.

The header files

The Berkeley socket development library has many associated header files. They include:

<sys/socket.h>
Definitions for the most basic of socket structures with the BSD socket API
<sys/types.h>
Basic data types associated with structures within the BSD socket API

<netinet/in.h>
Definitions for the socketaddr_in{} and other base data structures.
<sys/un.h>
Definitions and data type declarations for SOCK_UNIX streams

3a) TCP

TCP provides the concept of a connection. A process creates a TCP socket by calling the socket() function with the parameters PF_INET or PF_INET6 and SOCK_STREAM.

Server
Setting up a simple TCP server involves the following steps:

- Creating a TCP socket, with a call to socket().
- Binding the socket to the listen port, with a call to bind(). Before calling bind(), a programmer must declare a sockaddr_in structure, clear it (with bzero() or memset()), and the sin_family (AF_INET or AF_INET6), and fill its sin_port (the listening port, in network byte order) fields. Converting a short int to network byte order can be done by calling the function htons() (host to network short).
- Preparing the socket to listen for connections (making it a listening socket), with a call to listen().
- Accepting incoming connections, via a call to accept(). This blocks until an incoming connection is received, and then returns a socket descriptor for the accepted connection. The initial descriptor remains a listening descriptor, and accept() can be called again at any time with this socket, until it is closed.
- Communicating with the remote host, which can be done through send() and recv().
- Eventually closing each socket that was opened, once it is no longer needed, using close(). Note that if there were any calls to fork(), each process must close the sockets it knew about (the kernel keeps track of how many processes have a descriptor open), and two processes should not use the same socket at once.

Client
Setting up a TCP client involves the following steps:

- Creating a TCP socket, with a call to socket().
- Connecting to the server with the use of connect, passing a sockaddr_in structure with the sin_family set to AF_INET or AF_INET6, sin_port set to the port the endpoint is listening (in network byte order), and sin_addr set to the IPv4 or IPv6 address of the listening server (also in network byte order.)
- Communicating with the server by send()ing and recv()ing.

- Terminating the connection and cleaning up with a call to close(). Again, if there were any calls to fork(), each process must close() the socket.

Functions

socket()

socket() creates an endpoint for communication and returns a descriptor. socket() takes three arguments:

- *domain*, which specifies the protocol family of the created socket. For example:
  - PF_INET for network protocol IPv4 or
  - PF_INET6 for IPv6).
- *type*, one of:
  - SOCK_STREAM (reliable stream-oriented service)
  - SOCK_DGRAM (datagram service)
  - SOCK_SEQPACKET (reliable sequenced packet service), or
  - SOCK_RAW (raw protocols atop the network layer).
- *protocol*, usually set to 0 to represent the default transport protocol for the specified *domain* and *type* values (TCP for PF_INET or PF_INET6 and SOCK_STREAM, UDP for those PF_ values and SOCK_DGRAM), but which can also explicitly specify a protocol.

The function returns -1 if an error occurred. Otherwise, it returns an integer representing the newly-assigned descriptor.

Prototype:
int socket(int domain, int type, int protocol);

connect()

connect() It returns an integer representing the error code: 0 represents success, while -1 represents an error.

Certain types of sockets are *connectionless*, most commonly user datagram protocol sockets. For these sockets, connect takes on a special meaning: the default target for sending and receiving data gets set to the given address, allowing the use of functions such as send() and recv() on connectionless sockets.

Prototype:
int connect(int sockfd, const struct sockaddr *serv_addr, socklen_t addrlen);

bind()

bind() assigns a socket an address. When a socket is created using socket(), it is given an address family, but not assigned an address. Before a socket may accept incoming connections, it must be bound. bind() takes three arguments:

- sockfd, a descriptor representing the socket to perform the bind on
- my_addr, a pointer to a sockaddr structure representing the address to bind to.
- addrlen, a socklen_t field representing the length of the sockaddr structure.

It returns 0 on success and -1 if an error occurs.

Prototype:
int bind(int sockfd, struct sockaddr *my_addr, socklen_t addrlen);

listen()

listen() prepares a bound socket to accept incoming connections. This function is only applicable to the SOCK_STREAM and SOCK_SEQPACKET socket types. It takes two arguments:

- sockfd, a valid socket descriptor.
- backlog, an integer representing the number of pending connections that can be queued up at any one time. The operating system usually places a cap on this value.

Once a connection is accepted, it is dequeued. On success, 0 is returned. If an error occurs, -1 is returned.

Prototype:
int listen(int sockfd, int backlog);

accept()

Programmers use accept() to accept a connection request from a remote host. It takes the following arguments:

- sockfd, the descriptor of the listening socket to accept the connection from.
- cliaddr, a pointer to the sockaddr structure that accept() should put the client's address information into.
- addrlen, a pointer to the socklen_t integer that will indicate to accept() how large the sockaddr structure pointed to by cliaddr is. When accept() returns, the

socklen_t integer then indicates how many bytes of the cliaddr structure were actually used. The function returns a socket corresponding to the accepted connection, or -1 if an error occurs.

Prototype:
int accept(int sockfd, struct sockaddr *cliaddr, socklen_t *addrlen);

Blocking *vs.* nonblocking

Berkeley sockets can operate in one of two modes: blocking or non-blocking. A *blocking* socket will not "return" until it has sent (or received) all the data specified for the operation. This may cause problems if a socket continues to listen: a program may hang as the socket waits for data that may never arrive.

A socket is typically set to blocking or nonblocking mode using the fcntl() or ioctl() functions.

Cleaning up

The system will not release the resources allocated by the socket() call until a close() call occurs. This is especially important if the connect() call fails and may be retried. Each call to socket() must have a matching call to close() in all possible execution paths.

Here's a the algorithm to implement TCP socket programming to transfer a file between two machines:

Server Side:

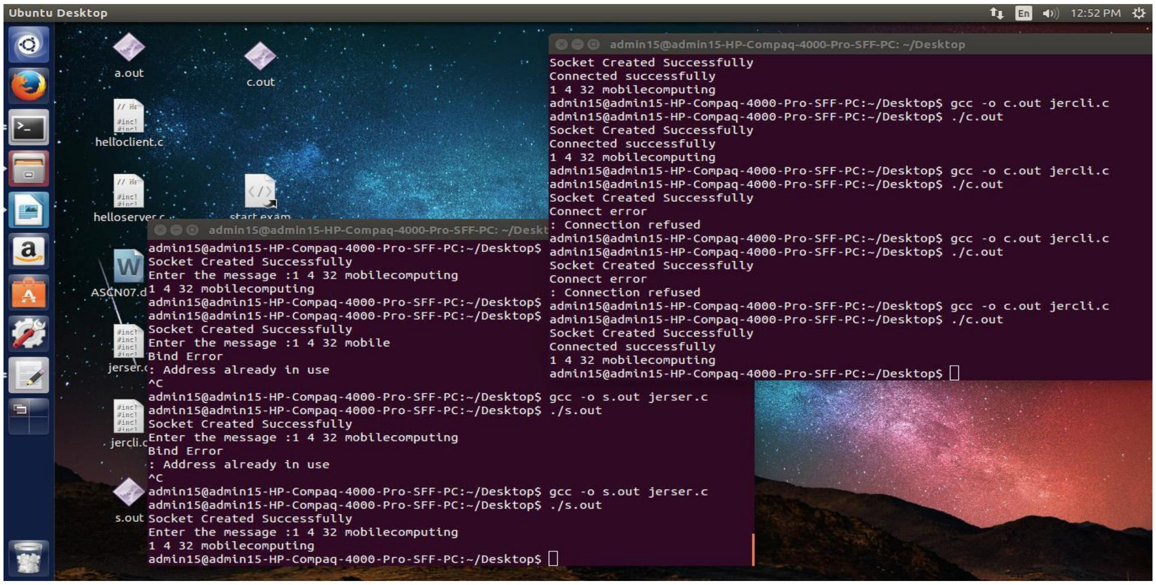
1. Create a socket
2. Bind the socket to an address and port
3. Listen for incoming connections
4. Accept a connection from a client
5. Open the file to be transferred
6. Read the file contents and send them over the socket to the client
7. Close the file and the socket

For Hello client just send the : Hello Client and Hello Server Strings instead of file operations.

Client Side:

1. Create a socket
2. Connect the socket to the server address and port
3. Receive the file contents from the server
4. Write the received data to a file
5. Close the socket

This pseudo code provides a high-level overview of the steps involved in implementing TCP socket programming to transfer a file between two machines. The actual implementation will require writing code to perform each of these steps in the programming language of your choice (e.g., C, Python, Java).


Ubuntu Desktop
En
12:52 PM
a.out
c.out
helloclient.c
helloserver.c
start exam
ASCN07.d
jerser.c
jercli.c
s.out
admin15@admin15-HP-Compaq-4000-Pro-SFF-PC: ~/Desktop
admin15@admin15-HP-Compaq-4000-Pro-SFF-PC: ~/Deskt

```
Socket Created Successfully
Connected successfully
1 4 32 mobilecomputing
admin15@admin15-HP-Compaq-4000-Pro-SFF-PC:~/Desktop$ gcc -o c.out jercli.c
admin15@admin15-HP-Compaq-4000-Pro-SFF-PC:~/Desktop$ ./c.out
Socket Created Successfully
Connected successfully
1 4 32 mobilecomputing
admin15@admin15-HP-Compaq-4000-Pro-SFF-PC:~/Desktop$ gcc -o c.out jercli.c
admin15@admin15-HP-Compaq-4000-Pro-SFF-PC:~/Desktop$ ./c.out
Socket Created Successfully
Connect error
: Connection refused
admin15@admin15-HP-Compaq-4000-Pro-SFF-PC:~/Desktop$ gcc -o c.out jercli.c
admin15@admin15-HP-Compaq-4000-Pro-SFF-PC:~/Desktop$ ./c.out
Socket Created Successfully
Connect error
: Connection refused
admin15@admin15-HP-Compaq-4000-Pro-SFF-PC:~/Desktop$ gcc -o c.out jercli.c
admin15@admin15-HP-Compaq-4000-Pro-SFF-PC:~/Desktop$ ./c.out
Socket Created Successfully
Connected successfully
1 4 32 mobilecomputing
admin15@admin15-HP-Compaq-4000-Pro-SFF-PC:~/Desktop$ 
```

```
admin15@admin15-HP-Compaq-4000-Pro-SFF-PC:~/Desktop$
Socket Created Successfully
Enter the message :1 4 32 mobilecomputing
1 4 32 mobilecomputing
admin15@admin15-HP-Compaq-4000-Pro-SFF-PC:~/Desktop$
admin15@admin15-HP-Compaq-4000-Pro-SFF-PC:~/Desktop$
Socket Created Successfully
Enter the message :1 4 32 mobile
Bind Error
: Address already in use
^C
admin15@admin15-HP-Compaq-4000-Pro-SFF-PC:~/Desktop$ gcc -o s.out jerser.c
admin15@admin15-HP-Compaq-4000-Pro-SFF-PC:~/Desktop$ ./s.out
Socket Created Successfully
Enter the message :1 4 32 mobilecomputing
Bind Error
: Address already in use
^C
admin15@admin15-HP-Compaq-4000-Pro-SFF-PC:~/Desktop$ gcc -o s.out jerser.c
admin15@admin15-HP-Compaq-4000-Pro-SFF-PC:~/Desktop$ ./s.out
Socket Created Successfully
Enter the message :1 4 32 mobilecomputing
1 4 32 mobilecomputing
admin15@admin15-HP-Compaq-4000-Pro-SFF-PC:~/Desktop$ 
```

3b) UDP

UDP consists of a connectionless protocol with no guarantee of delivery. UDP packets may arrive out of order, become duplicated and arrive more than once, or even not arrive at all. Due to the minimal guarantees involved, UDP has considerably less overhead than TCP. Being connectionless means that there is no concept of a stream or connection between two hosts, instead, data arrives in datagram's.
UDP address space, the space of UDP port numbers (in ISO terminology, the TSAPs), is completely disjoint from that of TCP ports.

Here's a simple pseudo code demonstrating how to implement UDP socket programming to transfer a file between two machines:

```
Server Side:
1. Create a UDP socket
2. Bind the socket to an address and port
3. Open the file to be transferred
4. Read the file contents in chunks
5. Send each chunk of data over the socket to the client
6. Close the file and the socket
```

For Hello client just send the : Hello Client and Hello Server Strings instead of file operations.

```
Client Side:
1. Create a UDP socket
2. Send a request to the server for the file
3. Receive data from the server in chunks
4. Write the received data to a file
5. Close the socket
```

This pseudo code provides a high-level overview of the steps involved in implementing UDP socket programming to transfer a file between two machines. The actual implementation will require writing code to perform each of these steps in the programming language of your choice (e.g., C, Python, Java). Keep in mind that with UDP, you'll need to handle potential packet loss, reordering, and fragmentation, as UDP does not guarantee reliable delivery like TCP does.

APPLICATION

1. Socket programming is essential in developing any application over a network.

FAQS

1. What is a socket?
2. What is the difference between a connection-less and a connection-oriented communication system? How are they implemented?
3. Which of them is more reliable?
4. Why is the port number required?
5. How is the socket programming in linux different from that in windows?
6. What is datagram?
7. Differentiate between TCP and UDP socket.

3c) Study of protocols and packet analyzer using Wireshark.

OBJECTIVES:

2. To study an existing protocol analyzer in order to get an idea about the various utilities that it provides
3. To understand the different ways of network monitoring

PROBLEM STATEMENT

Analyze packet formats of Ethernet, IP, TCP and UDP captured using Wireshark/Fiddler for traffic analysis tool in peer to peer mode for wired and wireless networks.

PROTOCOL ANALYZER

A network protocol analyzer also known as packet sniffer or ethernet sniffer is computer software (usually) or computer hardware that can intercept and log traffic passing over a digital network or part of a network. As data streams travel back and forth over the network, the sniffer captures each packet and eventually decodes and analyzes its content according to the appropriate RFC or other specifications. Depending on the network structure (hub or switch) one can sniff all or just parts of the traffic from a single machine within the network; however, there are some methods to avoid traffic narrowing by switches to gain access to traffic from other systems on the network (e.g. ARP spoofing). For network monitoring purposes it may also be desirable to monitor all data packets in a LAN by using a network switch with a so-called monitoring port (it mirrors all packets passing through all ports of the switch).

The special network device driver used for some packet sniffing software is said to operate in "promiscuous mode" as it listens to everything on the wire.

The versatility of packet sniffers means they can be used to:

- Analyze network problems
- Detect network intrusion attempts
- Gain information for effecting a network intrusion
- Monitor network usage
- Gather and report network statistics
- Filter suspect content from network traffic
- Spy on other network users and collect sensitive information such as passwords (Depending on any content encryption methods which may be in use)

- Reverse engineer protocols used over the network
- Debug client/server communications

Wireshark

Wireshark is a powerful network packet analyzer widely used by network administrators, security professionals, developers, and enthusiasts to capture, analyze, and troubleshoot network traffic. It allows users to inspect data packets in real-time or from saved capture files, providing insights into network behavior, identifying performance issues, diagnosing network problems, and detecting security threats.

Features:

- **Packet Capture**: Wireshark captures packets from a wide range of network interfaces including Ethernet, Wi-Fi, and Bluetooth. It supports live packet capture as well as the ability to read packets from previously captured files.

- **Packet Analysis**: Wireshark provides detailed information about each captured packet, including protocol headers, payload data, packet timing, and other metadata. It supports a vast array of protocols including TCP/IP, UDP, HTTP, DNS, SSL/TLS, and many more.

- **Filtering and Search**: Wireshark offers powerful filtering and search capabilities to isolate specific packets or protocols of interest. Filters can be applied based on various criteria such as IP address, port number, protocol, packet length, and packet content.

- **Packet Decoding**: Wireshark decodes packet data according to the selected protocols, providing a human-readable representation of the captured information. It supports dissecting complex protocols and can handle both standard and custom protocols.

- **Statistics**: Wireshark provides various statistical tools to analyze network traffic patterns, including packet count, traffic volume, throughput, latency, packet loss, and protocol distribution. These statistics help in identifying performance bottlenecks and abnormal network behavior.

- **Protocol Analysis**: Wireshark includes built-in protocol dissectors that decode and analyze the behavior of different network protocols. It highlights protocol-specific issues, errors, and anomalies, aiding in protocol debugging and troubleshooting.

- **Colorized Packet Display**: Wireshark colorizes packets based on their types and statuses, making it easier to visually identify different types of packets and potential problems.

- **Export and Save**: Wireshark allows users to export captured packets in various formats including plain text, CSV, XML, and PCAP. It also supports saving captured data for later analysis or sharing with colleagues.

- **Scripting and Automation**: Wireshark provides scripting capabilities using Lua, allowing users to automate tasks, extend functionality, and customize the user interface.

Use Cases:

- **Network Troubleshooting**: Wireshark helps diagnose network issues such as packet loss, latency, and connectivity problems by analyzing packet traces and identifying abnormal behavior.

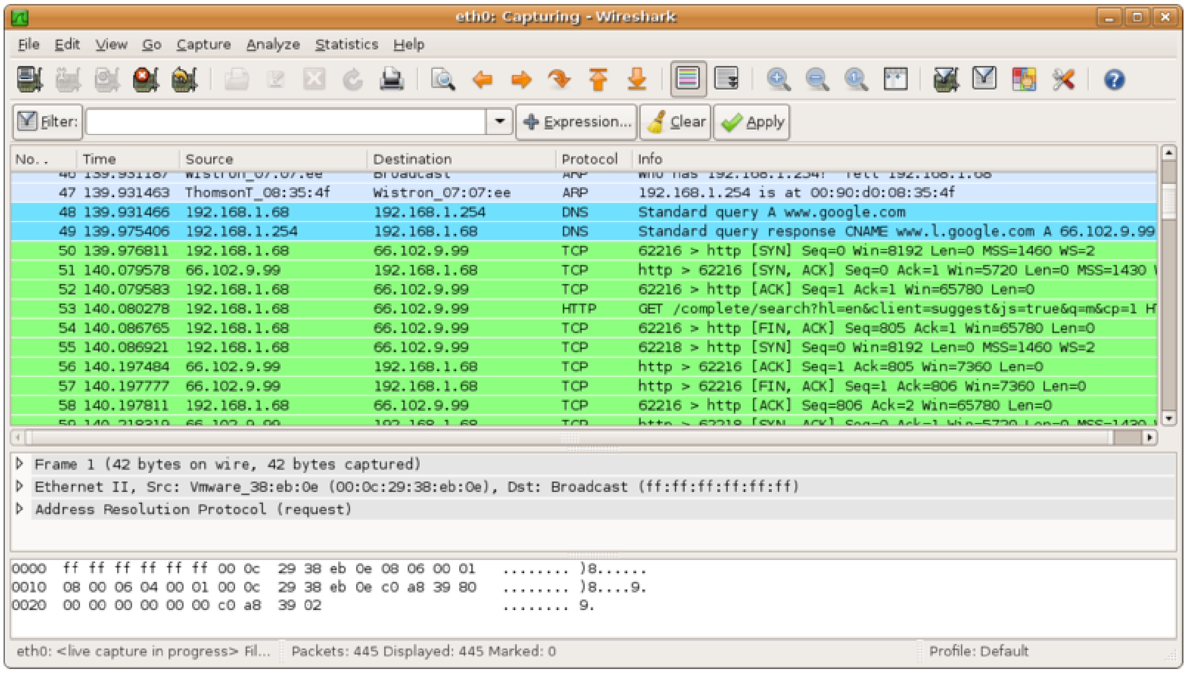
- **Security Analysis**: Wireshark assists in detecting and investigating security threats such as network attacks, malware infections, and data breaches by inspecting network traffic for suspicious patterns and anomalies.

- **Performance Optimization**: Wireshark enables network administrators to optimize network performance by analyzing traffic patterns, identifying bottlenecks, and optimizing network configurations.

- **Protocol Development**: Wireshark is used by protocol developers to debug and test new network protocols or protocol implementations by inspecting packet traces and verifying protocol behavior.

- **Education and Training**: Wireshark is a valuable educational tool for learning about networking concepts, protocols, and packet analysis techniques. It is widely used in networking courses and training programs.

Conclusion: Wireshark is an indispensable tool for network analysis and troubleshooting, offering a rich set of features for capturing, analyzing, and dissecting network traffic. Its intuitive user interface, extensive protocol support, and powerful analysis capabilities make it an essential tool for network professionals across various industries. Whether diagnosing network issues, investigating security incidents, or optimizing network performance, Wireshark provides the necessary tools to understand and interpret network traffic effectively.



No. .	Time	Source	Destination	Protocol	Info
46	139.931187	Wistron_07:07:ee	Broadcast	ARP	Who has 192.168.1.254? Tell 192.168.1.68
47	139.931463	ThomsonT_08:35:4f	Wistron_07:07:ee	ARP	192.168.1.254 is at 00:90:d0:08:35:4f
48	139.931466	192.168.1.68	192.168.1.254	DNS	Standard query A www.google.com
49	139.975406	192.168.1.254	192.168.1.68	DNS	Standard query response CNAME www.l.google.com A 66.102.9.99
50	139.976811	192.168.1.68	66.102.9.99	TCP	62216 > http [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=2
51	140.079578	66.102.9.99	192.168.1.68	TCP	http > 62216 [SYN, ACK] Seq=0 Ack=1 Win=5720 Len=0 MSS=1430
52	140.079583	192.168.1.68	66.102.9.99	TCP	62216 > http [ACK] Seq=1 Ack=1 Win=65780 Len=0
53	140.080278	192.168.1.68	66.102.9.99	HTTP	GET /complete/search?hl=en&client=suggest&js=true&q=m&cp=1 H
54	140.086765	192.168.1.68	66.102.9.99	TCP	62216 > http [FIN, ACK] Seq=805 Ack=1 Win=65780 Len=0
55	140.086921	192.168.1.68	66.102.9.99	TCP	62218 > http [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=2
56	140.197484	66.102.9.99	192.168.1.68	TCP	http > 62216 [ACK] Seq=1 Ack=805 Win=7360 Len=0
57	140.197777	66.102.9.99	192.168.1.68	TCP	http > 62216 [FIN, ACK] Seq=1 Ack=806 Win=7360 Len=0
58	140.197811	192.168.1.68	66.102.9.99	TCP	62216 > http [ACK] Seq=806 Ack=2 Win=65780 Len=0
59	140.218319	66.102.9.99	192.168.1.68	TCP	http > 62218 [SYN, ACK] Seq=0 Ack=1 Win=5720 Len=0 MSS=1430

```
▷ Frame 1 (42 bytes on wire, 42 bytes captured)
▷ Ethernet II, Src: Vmware_38:eb:0e (00:0c:29:38:eb:0e), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
▷ Address Resolution Protocol (request)
```

```
0000  ff ff ff ff ff ff 00 0c  29 38 eb 0e 08 06 00 01   ........ )8......
0010  08 00 06 04 00 01 00 0c  29 38 eb 0e c0 a8 39 80   ........ )8....9.
0020  00 00 00 00 00 00 c0 a8  39 02                     ........ 9.
```

eth0: <live capture in progress> Fil... Packets: 445 Displayed: 445 Marked: 0 Profile: Default

APPLICATION

1. Protocol analyzers are very useful in keeping track network activity and debugging network related issues

FAQS

Please refer the above theory for answers

1. What is a Protocol analyzer? Why is it used?
2. What are the features of Ethereal?
3. What other analyzers did you study?
4. What is packet sniffing?
5. What is promiscuous mode?
6. Can traffic be generated using a protocol analyzer?

Experiment Number: 04

Title:. . Design and develop a website using toggled or dynamic tabs or pills with bootstrap and JQuery to show the relevance of SDP, EDI, DT and Course projects in VIT.

Theory: Creating a website with toggled or dynamic tabs/pills using Bootstrap and jQuery involves setting up the necessary HTML structure, applying Bootstrap classes for styling, and writing JavaScript/jQuery to handle the toggling functionality.

Menus:

Most web pages have some kind of a menu. In HTML, a menu is often defined in an unordered list <ul> (and styled afterwards) shown below.

```
<ul>
<li><a href="#">Home</a></li>
<li><a href="#">Menu 1</a></li>
<li><a href="#">Menu 2</a></li>
<li><a href="#">Menu 3</a></li>
</ul>
```

If you want to create a horizontal menu of the list above, add the .list-inline class to <ul>:

```
<ul class="list-inline">
```

Home
Menu 1
Menu 2
Menu 3

Nav-Tabs:

Tabs are created with <ul class="nav nav-tabs">:

Tip: Also mark the current page with <li class="active">.

The following example creates navigation tabs:

Example

```
<ul class="nav nav-tabs">
<li class="active"><a href="#">Home</a></li>
<li><a href="#">Menu 1</a></li>
<li><a href="#">Menu 2</a></li>
<li><a href="#">Menu 3</a></li>
</ul>
```

Nav-Tabs with Dropdown Menu

Home
Menu 1
Menu 2
Menu 3
Tabs can also hold dropdown menus.

The following example adds a dropdown menu to "Menu 1":

Example

```
<ul class="nav nav-tabs">
<li class="active"><a href="#">Home</a></li>
<li class="dropdown">
<a class="dropdown-toggle" data-toggle="dropdown" href="#">Menu 1
<span class="caret"></span></a>
<ul class="dropdown-menu">
<li><a href="#">Submenu 1-1</a></li>
<li><a href="#">Submenu 1-2</a></li>
<li><a href="#">Submenu 1-3</a></li>
</ul>
</li>
<li><a href="#">Menu 2</a></li>
<li><a href="#">Menu 3</a></li>
</ul>
```

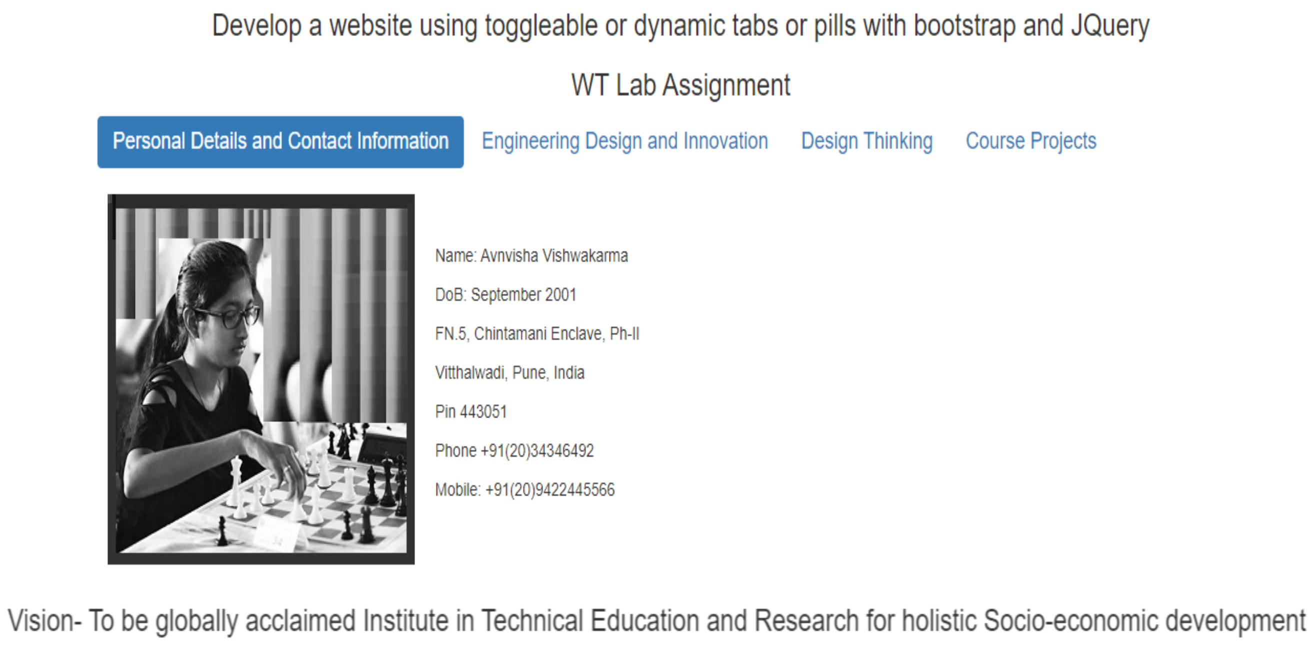
Using Nav Pills

Home
Menu 1
Menu 2
Menu 3
Pills are created with <ul class="nav nav-pills">. Also mark the current page with <li class="active">:

Example

```
<ul class="nav nav-pills">
<li class="active"><a href="#">Home</a></li>
<li><a href="#">Menu 1</a></li>
<li><a href="#">Menu 2</a></li>
<li><a href="#">Menu 3</a></li>
</ul>
```

Sample Output:



Develop a website using toggleable or dynamic tabs or pills with bootstrap and JQuery

WT Lab Assignment

Personal Details and Contact Information
Engineering Design and Innovation
Design Thinking
Course Projects

Name: Avnvisha Vishwakarma

DoB: September 2001

FN.5, Chintamani Enclave, Ph-II

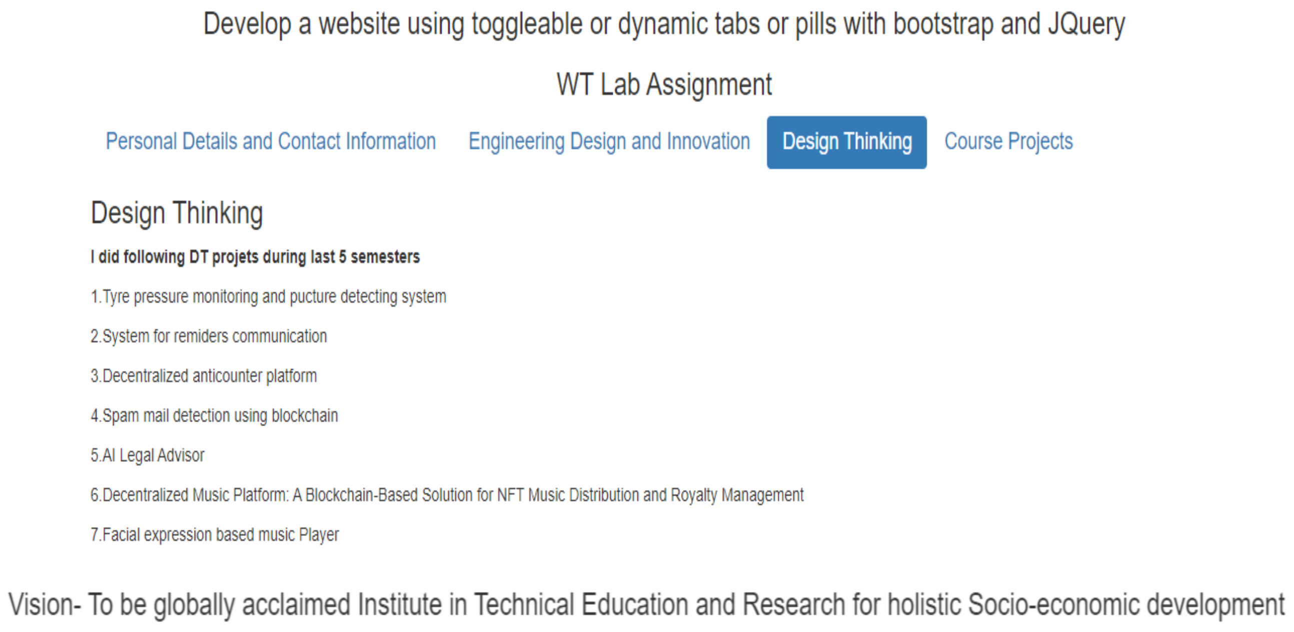
Vitthalwadi, Pune, India

Pin 443051

Phone +91(20)34346492

Mobile: +91(20)9422445566

Vision- To be globally acclaimed Institute in Technical Education and Research for holistic Socio-economic development



Develop a website using toggleable or dynamic tabs or pills with bootstrap and JQuery

WT Lab Assignment

Personal Details and Contact Information
Engineering Design and Innovation
Design Thinking
Course Projects

Design Thinking

I did following DT projets during last 5 semesters

1.Tyre pressure monitoring and pucture detecting system

2.System for remiders communication

3.Decentralized anticounter platform

4.Spam mail detection using blockchain

5.AI Legal Advisor

6.Decentralized Music Platform: A Blockchain-Based Solution for NFT Music Distribution and Royalty Management

7.Facial expression based music Player

Vision- To be globally acclaimed Institute in Technical Education and Research for holistic Socio-economic development

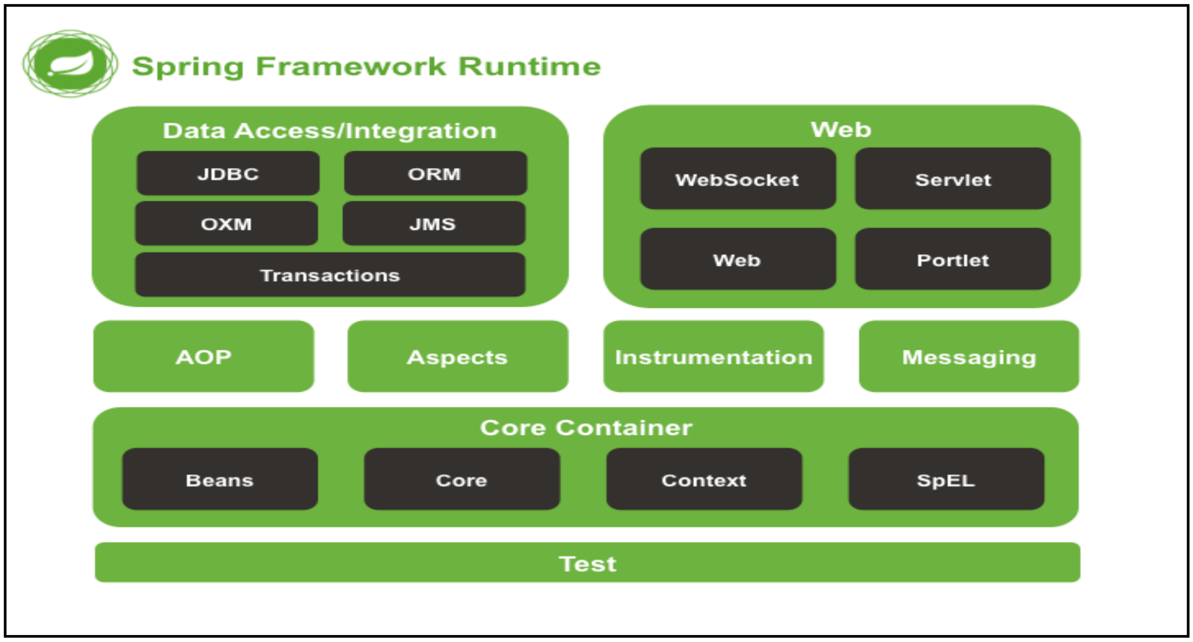
Experiment Number: 05

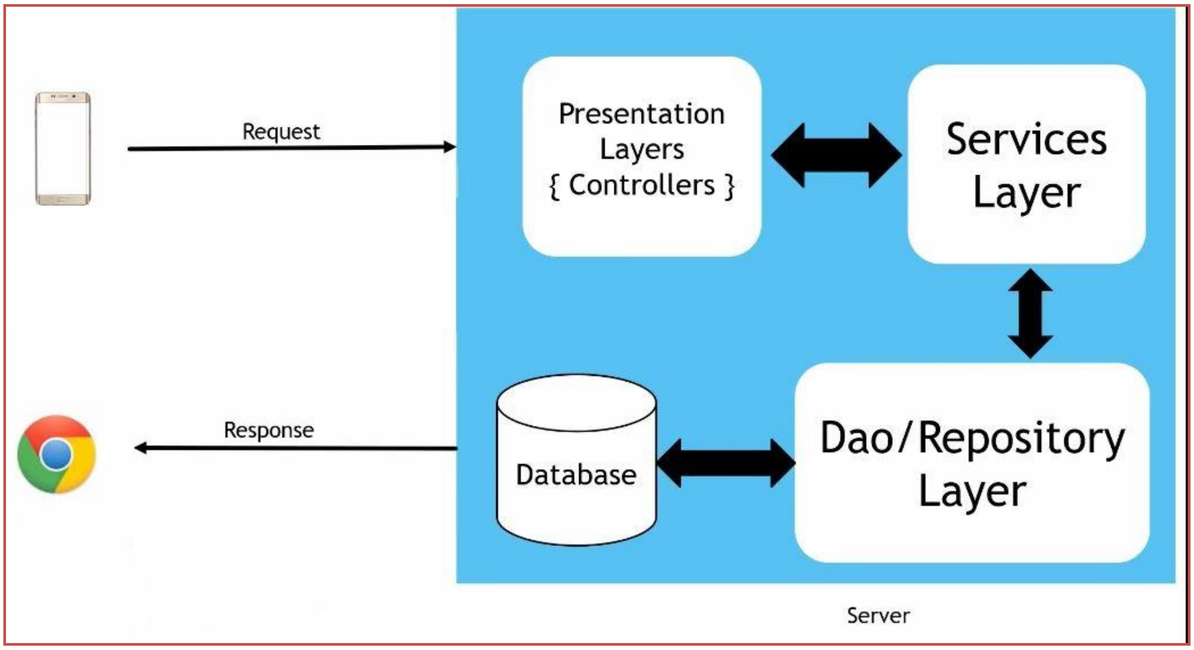
Title:. Design and develop a responsive website to prepare one semester result of VIT students using REACT, Django/Springboot/Node JS/ PHP and MySQL/ MongoDB/Oracle. Take any four subjects with MSE Marks (30%) ESE Marks (70%).

Theory:

Software Required: One ID: Eclipse/Intellij/Netbeans/STS/Visual Code: STS Download : Spring Tools and Postman for client testing.

Spring Framework Runtime:





What is POSTMAN Tool?

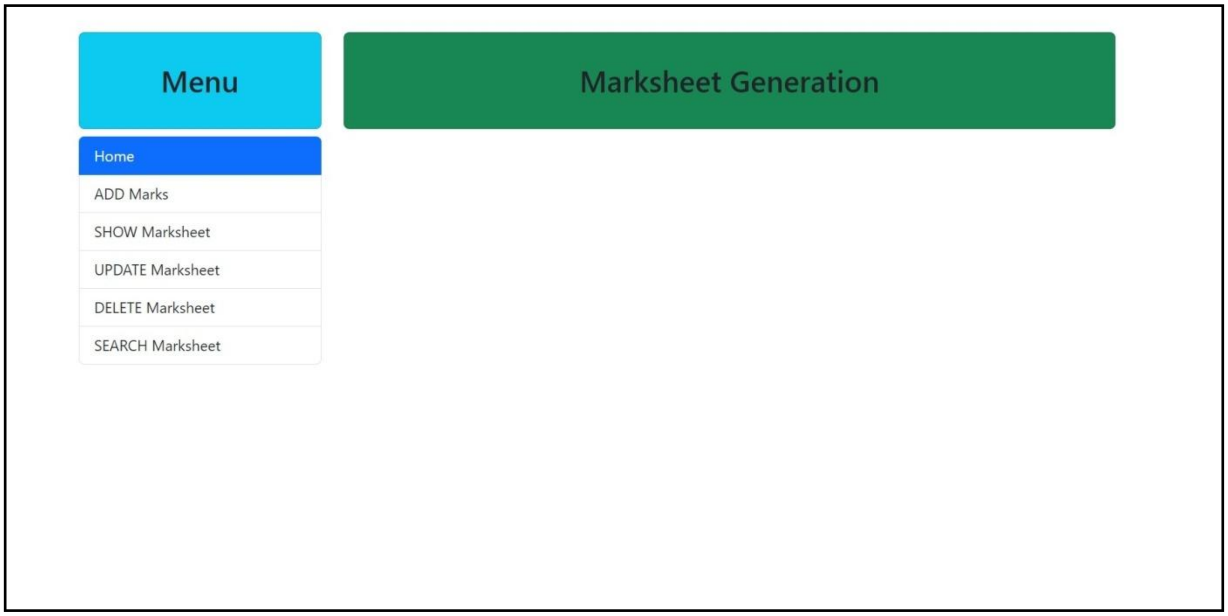
Postman is an interactive and automatic tool for verifying the APIs of your project. Postman is a Google Chrome app for interacting with HTTP APIs. It presents you with a friendly GUI for constructing requests and reading responses.

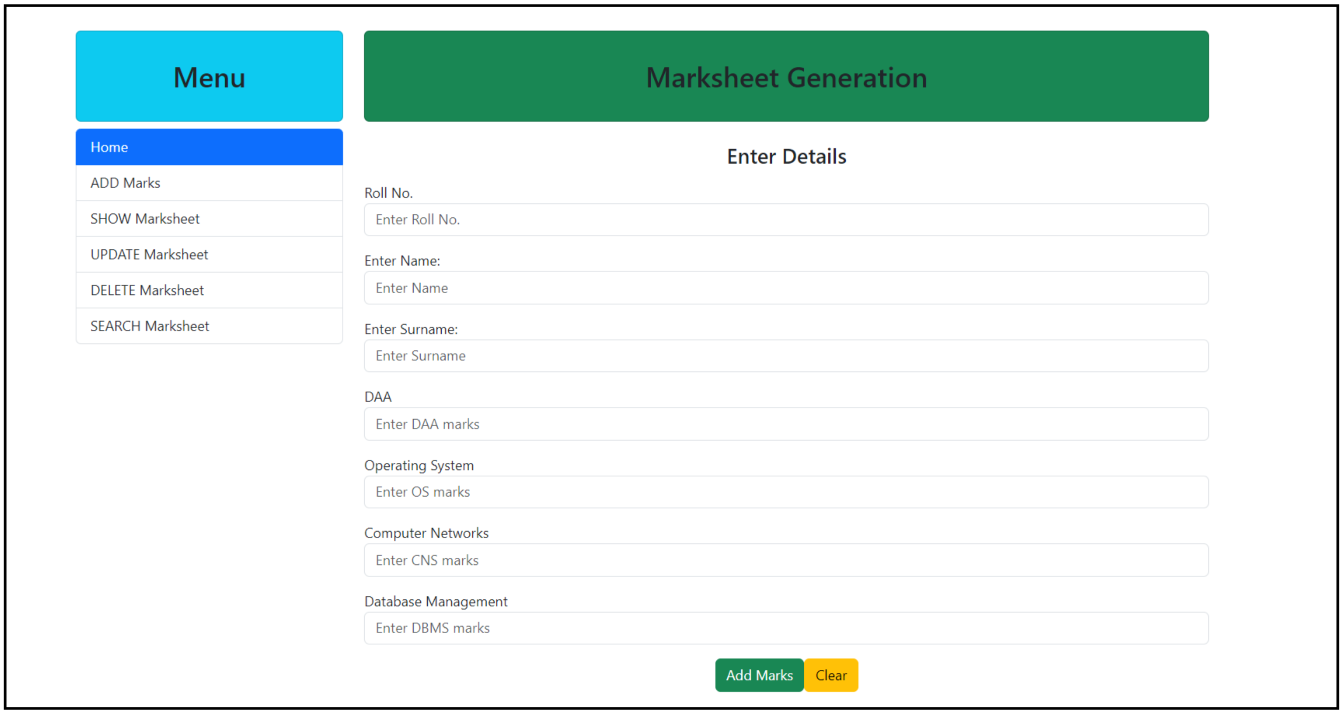
Request Method	Response
GET	Get one or multiple records from database
POST	Add one or more records in database
PUT	Update one or more records in database
DELETE	Delete one or more records from database

Procedure:

- Create a project using spring boot initialize website
- **Add Dependencies:** Spring Web.......
- **Click on:** Generate ....it will generate jar file in download folder
- Extract it
- Open it in ID
- Import - as a maven project
- Tick on ------ pom.xml
- It will update
- **Let us build controller : Controller for REST API** (REST - Representational State Transfer)
- **Create entity class**
- **Generate constructors, getters and setters for entity class**
- **Create interface class**
- **Create service class**
- **Database Connectivity: using JPA**
- **Extends it to JPA Repository:**
- **Service Implementation:**

Sample Output:





Experiment Number: 06

Title:. Design and develop a responsive website for students CV Management or an online book store using REACT, Django/Springboot/Node JS/ PHP and MySQL/ MongoDB/Oracle having 1) Home Page2) Login Page 3) Catalogue Page: 4) Registration Page: (database)

Theory:

Understanding J2EE / Spring Boot Communication Client Server Architecture (Horizontal Slicing)

Client		Server				Database
Browser - Chrome, Mozilla, Mobile Phone, Laptop, Desktop **Postman/React**	**----->** Request **<-----** Reply	Presentation layer **Controller uses**	Service Layer	DAO Layer	**----->** Request **<-----** Reply	Repository layer
		To accept Request (What client wants)	**Business Logic** (Classess, Methods)	Database Connectivity		Database Oracle, MySQL

Spring allows you:

--Make a Java method execute in a database transaction without dealing with transaction APIs.

--Make a local Java method a remote procedure without having to deal with remote APIs.

--Make a local Java method a management operation without having to deal with JMX APIs.

--Make a local Java method a message handler without having to deal with JMS APIs.

-- The Spring Framework Inversion of Control (IoC) component provides a formalized means of composing disparate components into a fully working application ready for use.

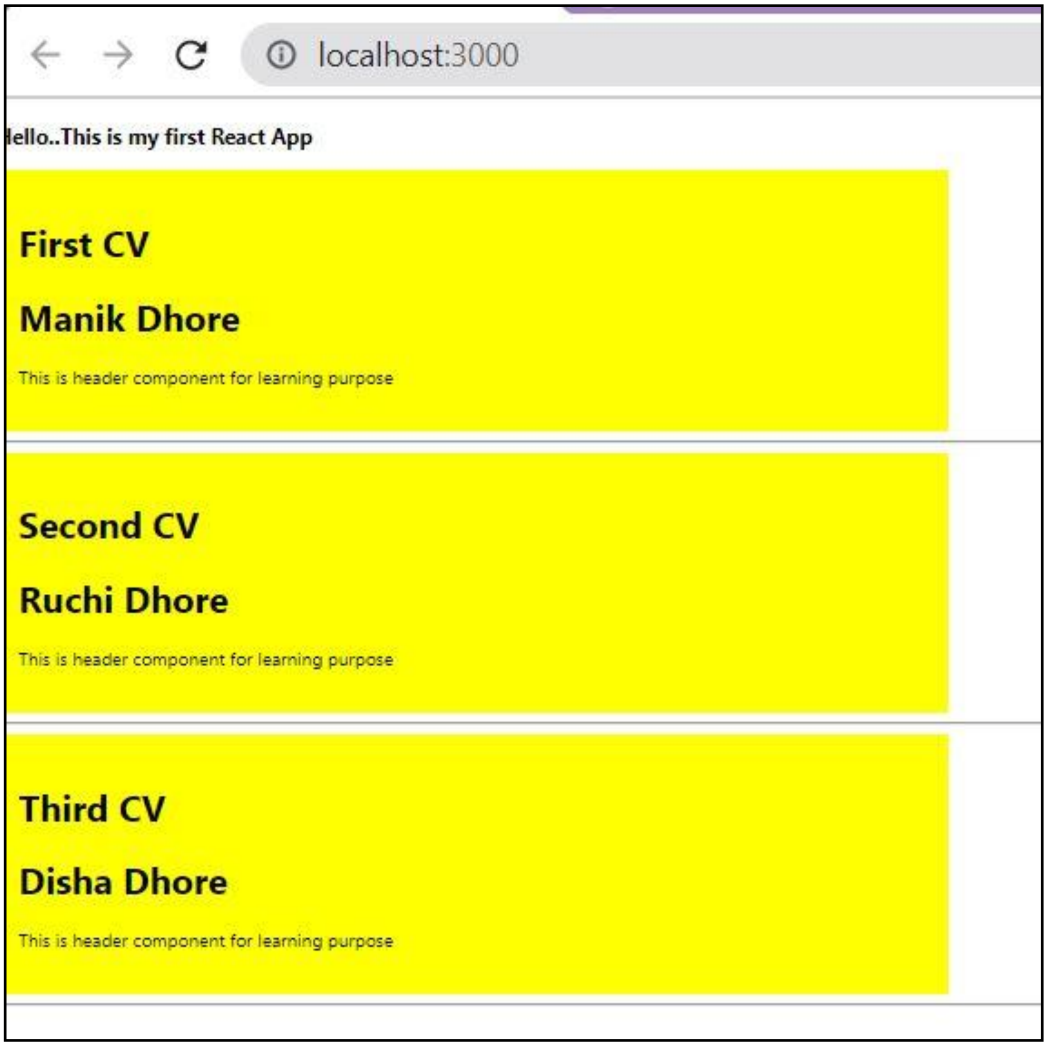
Procedure:

- Create a project using spring boot initialize website
- **Add Dependencies:** Spring Web.......MySQL Driver......Spring Data JPA
- **Click on:** Generate ....it will generate jar file in download folder
- Extract it
- Open it in ID
- Import - as a maven project
- Tick on ------ pom.xml
- It will update
- **Let us build controller : Controller for REST API** (REST - Representational State Transfer)
- **Create entity class**
- **Generate constructors, getters and setters for entity class**
- **Create interface class**
- **Create service class**
- **Database Connectivity: using JPA**

- **Extends it to JPA Repository:**
- **Service Implementation:**

- **Use – React JS to create front end and axios for routing**

Sample Output: Header Component for CV Management



lello..This is my first React App

First CV

Manik Dhore

This is header component for learning purpose

Second CV

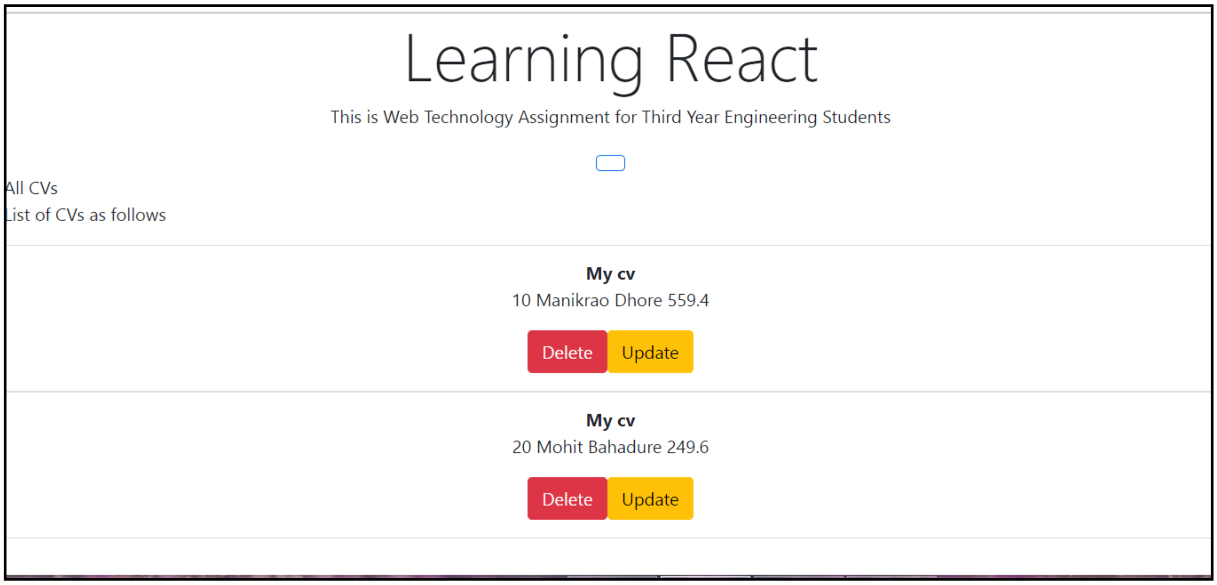
Ruchi Dhore

This is header component for learning purpose

Third CV

Disha Dhore

This is header component for learning purpose



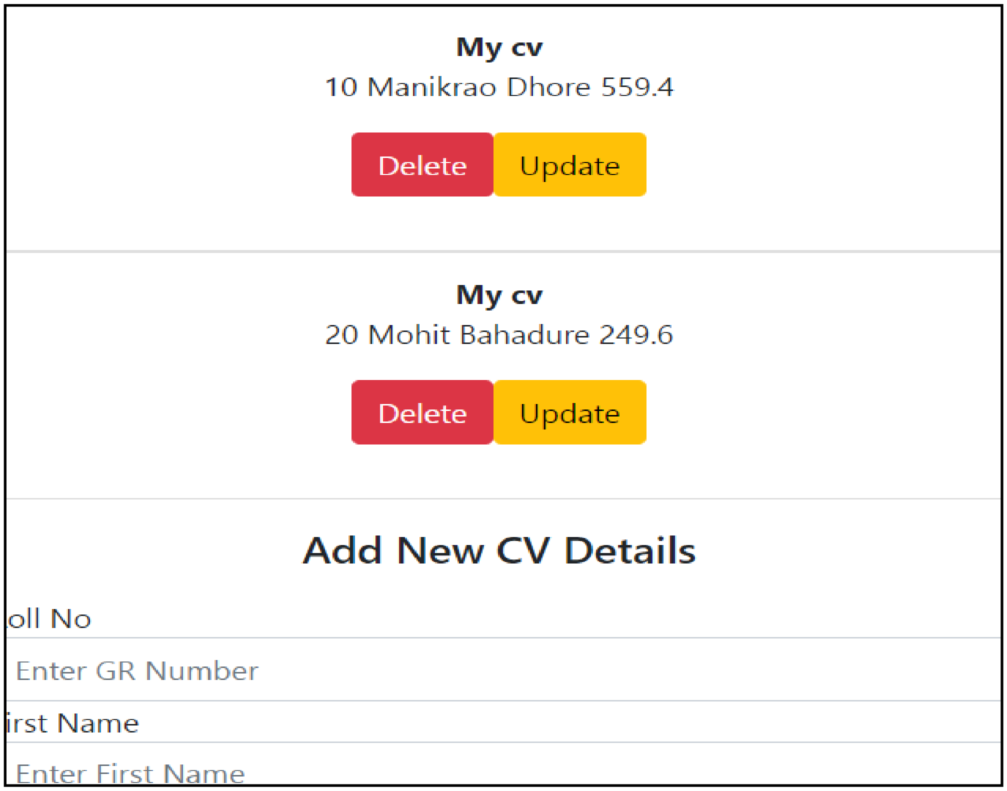
Learning React

This is Web Technology Assignment for Third Year Engineering Students

All CVs
List of CVs as follows

My cv
10 Manikrao Dhore 559.4

My cv
20 Mohit Bahadure 249.6



My cv
10 Manikrao Dhore 559.4

My cv
20 Mohit Bahadure 249.6

Add New CV Details

oll No
Enter GR Number
irst Name
Enter First Name