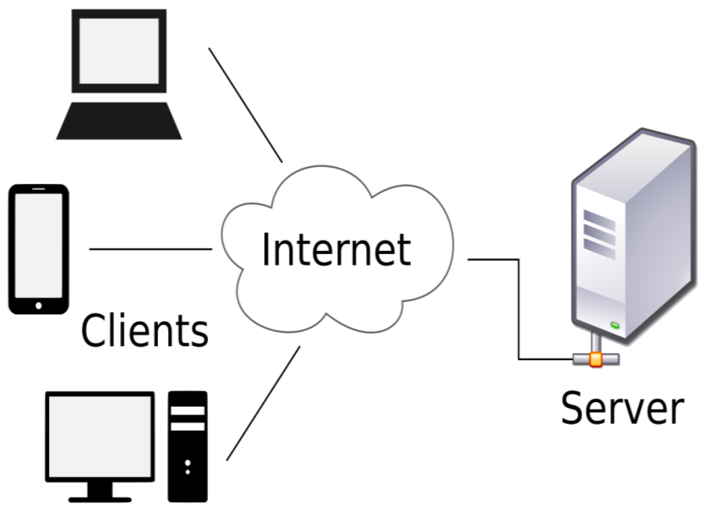
Assignment

Unit: One

1.How does client server architecture works?

Client-server architecture is a computer network design where multiple clients request and receive services from a centralized server. The client is typically a computer, smartphone, or web browser that sends requests to the server. The server is a powerful machine that processes these requests and sends back the required data or services to clients. This model is widely used in web applications, email systems, and online databases.



When a client wants to access a website, for example, it sends a request to the web server using the internet. The server then processes this request, retrieves the necessary files (such as HTML, CSS, and JavaScript) and sends them back to the client. The client's web browser then displays the website for the user. This communication happens using protocols like HTTP or HTTPS. The client and server communicate in a request-response cycle, ensuring smooth data transfer between them.

2.What are the factors to be considered when designing a software?

Some of the factors to be considered when designing software are :

a. User Requirements : The software should be user-friendly, solve real problems, and meet the expectations of end users.
b. Scalability : The software should be designed to handle growth in users, data, and features without any issues.
c. Security : Security is one of the most critical factors in software development. If a system is not secure, it becomes vulnerable to cyber attacks, data breaches, and unauthorized access.
d. Compatibility : The software should work across different devices, operating systems, and browsers to reach a wider audience.
e. Efficiency : Resources should be used efficiently by the program.
f. Maintainability : The design should be so simple so that it can be easily maintainable by other designers.

3. Why do we really need network programming tools and platform? Explain some of them.

Network programming tools and platform are essential for building applications that communicate with each other over a network. Some of them are:

a. Wireshark : Wireshark is the industry standard tool to collect and interpret this traffic, with almost one million downloads every month. It enables engineers to quickly get to the packet level of a problem. This allows them to quickly determine if the issue is due to the network, server, service or client. Rather than guess at the cause of an issue, they can utilize Wireshark to see the real truth.

b. Nmap : Nmap provides utilities to determine what hosts are available on the network, what ports are available on those hosts, what OS and firewalls are in use and much more. It has the capability to scan whole subnets and TCP port ranges, allowing engineers to spot problem devices and open sockets.

c. Cisco Packet Tracer : Cisco designed the Packet Tracer tool to help engineers simulate and test network environments before they are rolled out to the enterprise. These tools are especially helpful when preparing for industry certification exams, as large environments can be built and tested without the need for expensive hardware.

d. NetStat : NetStat is a command-line tool available on most operating systems that will display the current status of TCP and UDP conversations. This data is very helpful when tracking down server load, mapping connections to a specific process or monitoring the security of a system that is under attack.

Unit: Two

1. Write a JAVA program to illustrate different types of address.

```
import java.net.InetAddress;
 import java.net.UnknownHostException;
public class two { public static void main(String[] args)
{
 try{ InetAddress iaddress = InetAddress.getByName("224.1.1.1");
if(iaddress.isAnyLocalAddress())
{
 System.out.println(iaddress+"is a local address.");
}
if(iaddress.isLoopbackAddress())
{
System.out.println(iaddress+"is loopback address.");
 }
if(iaddress.isLinkLocalAddress())
{
System.out.println(iaddress+"is a link-local address.");
}
else if(iaddress.isSiteLocalAddress())
{
System.out.println(iaddress+"is a site-local address.");
}
else{ System.out.println(iaddress+"is a global address.");
}
 // multicast network
if(iaddress.isMulticastAddress())
{
if(iaddress.isMCGlobal()){ System.out.println(iaddress+"is a global multicast address.");
 }
else if(iaddress.isMCOrgLocal())
{ System.out.println(iaddress+"is a organizational wide multicast address.");
}
else if(iaddress.isMCSiteLocal())
{ System.out.println(iaddress+"is a site wide multicast address.");
}
else if(iaddress.isMCLinkLocal())
{
 System.out.println(iaddress+"is a subnet wide multicast address.");
}
else if(iaddress.isMCNodeLocal())
{
System.out.println(iaddress+"is an interface-local multicast address.");
 }
Else
{
 System.out.println(iaddress+"is a unicast address.");
 }
} }
catch(UnknownHostException e)
{
System.out.println("could not resolve."+args[0]);
} } }
```

2. **In what ways getHostName() differs from getCannonicalHostName()? Write a JAVA program that, Displays both hostname and the cannonical hostname of a domain. Utilizes getAddress() to determine if the given address is IPv4 or IPv6.**
   - getHostName(): Returns the hostname of the IP address if available. It might return the same as the IP address if the hostname cannot be resolved.
   - getCanonicalHostName(): Returns the fully qualified domain name (FQDN) of the host. It provides a more precise and complete hostname by resolving the hostname through a DNS lookup.

Program

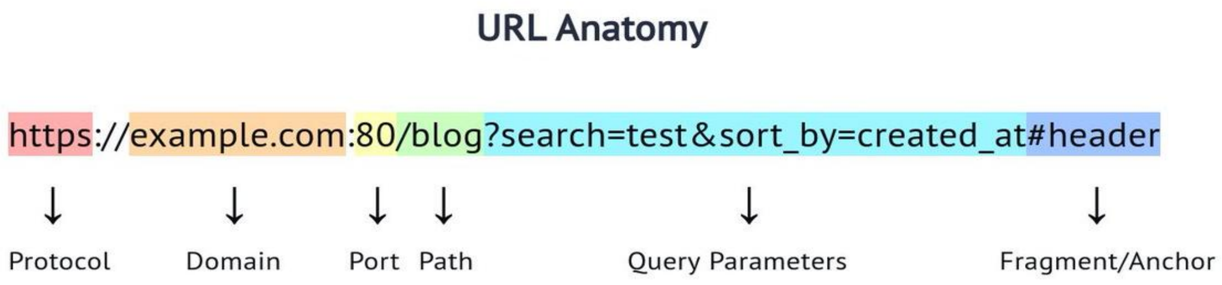
```
import java.net.InetAddress;
public class addressing {
   public static void main(String[] args) {
      try{
      InetAddress inetaddressins = InetAddress.getByName("facebook.com");
      String hostname=inetaddressins.getHostName();
```

```
        String canonicalhostname = inetaddressins.getCanonicalHostName();
        System.out.println(hostname);
        System.out.println(canonicalhostname);
        byte[] hfjsk = inetaddressins.getAddress();
        System.out.println(hfjsk);
        if(hfjsk.length==4){
          System.out.println("ipv4");
        }
        else if(hfjsk.length==6){
          System.out.println("ipv6");
        }
        else{
          System.out.println("invalid input.");
        }
        }catch(Exception e){
          System.out.println(e.getMessage());
        }
      }
    }
```

Unit: Three

1. What is URL? Give an example of a URL that shows each components of URL.

A URL or Uniform Resource Locator is a Unique identifier that is contained by all the resources available on the internet. It can help to locate a particular resource due to its uniqueness. It is also known as the web address. A URL consists of different parts like protocol, domain name, etc.


URL Anatomy
https://example.com:80/blog?search=test&sort_by=created_at#header
Protocol
Domain
Port
Path
Query Parameters
Fragment/Anchor

2. URLs vs URIs with examples for each. Can all URLs be URIs? Why?

URL	**URI**
URL is a type of URI.	URI is the superset of URL.
It comprises of protocol, domain, path, hash, and so on.	It comprises of scheme, authority, path, query and many more.
URL provides the details about what type of protocol is to be used.	URI doesn’t contains the protocol specification.
URL is used to describe the identity of an item.	URI provides a technique for defining the identity of an item.

Yes, all URLs are considered URIs because a URL (Uniform Resource Locator) is a specific type of URI (Uniform Resource Identifier), meaning that a URL provides location information which falls under the broader category of resource identification that a URI represents; essentially, every URL can be identified as a URI, but not every URI is a URL.