Chapter 3:
1.explain Client-Server architecture,
Client-Server Architecture is a shared architecture system where loads of client-server are divided. The client-server architecture is a centralized resource system where server holds all the resources. The server receives numerous performances at its edge for sharing resources to its clients when requested. Client and server may be on the same or in a network. The server is profoundly stable and scalable to return answers to clients. This Architecture is Service Oriented which means client service will not be interrupted. Client-Server Architecture subdues network traffic by responding to the inquiries of the clients rather than a complete file transfer. It restores the file server with the database server.

Client computers implement a bond to allow a computer user to request services of the server and to represent the results the server returns. Servers wait for requests to appear from clients and then return them. A server usually gives a standardized simple interface to clients to avoid a hardware/software confusion. Clients are located at workplaces or on personal machines, at the same time servers will be located somewhere powerful in the network. This architecture is useful mostly when clients and the server each have separate tasks that they routinely perform. Many clients can obtain the server's information concurrently, and also a client computer can execute other tasks, for instance, sending e-mails.

Types of Client Server Architecture

1-tier architecture

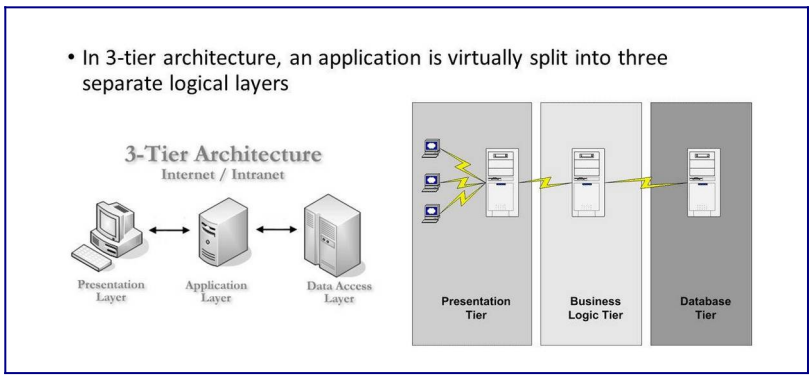
In this category of client-server setting, the user interface, marketing logic and data logic are present in the same system. This kind of service is reasonable but it is hard to manage due to data variance that allots replication of work. One-tier architecture consists of layers.

For example, Presentation, Business, Data Access layers within a single software package. The data is usually stored in the local system or a shared drive. Applications which handle all the three tiers such as MP3 player, MS Office come under one-tier application.

2-tier architecture

In this type of client-server environment, the user interface is stored at client machine and the database is stored on the server. Database logic and business logic are filed at either client or server but it needs to be maintained. If Business Logic and Data Logic are collected at a client side, it is named as **fat client thin server architecture**. If Business Logic and Data Logic are handled on the server, it is called thin client fat server architecture. This is considered as affordable.

In two-tier architecture, client and server have to come in direct incorporation. If a client is giving an input to the server there shouldn't be any intermediate. This is done for rapid results and to avoid confusion between different clients. For instance, online ticket reservations software use this two-tier architecture.



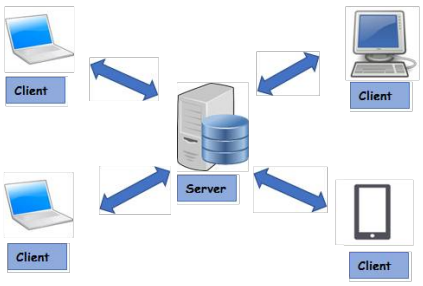
3-tier architecture

In this variety of client-server context, an extra middleware is used that means client request goes to the server through that middle layer and the response of server is received by middleware first and then to the client. This architecture protects 2-tier architecture and gives the best performance. This system comes expensive but it is simple to use. The middleware stores all the business logic and data passage logic. The idea of middleware is to database staging, queuing, application execution, scheduling etc. A Middleware improves flexibility and gives the best performance.

The **Three-tier architecture is split into 3 parts**, namely, The presentation layer (Client Tier), Application layer (Business Tier) and Database layer (Data Tier). The Client system manages Presentation layer; the Application server takes care of the Application layer, and the Server system supervises Database layer.

In the present scenario of online business, there has been growing demands for the quick responses and quality services. Therefore, the complex client architecture is crucial for the business activities. Companies usually explore possibilities to keep service and quality meet to maintain its marketplace with the help of client-server architecture. The architecture increases productivity through the practice of cost-efficient user interfaces, improved data storage, expanded connectivity and secure services.

2explain Client-Server communication scenario.?



In the age of Computers, a client and a server are two computers separated by miles but connected by Web (Internet). However, important point to note here is that it is not necessary that Client and a Server should be miles apart, it could be that Client and Server programs are running as two processes on the same computer.

For understanding the Client-Server architecture, we would assume that Client and Server are separated located in different geographies and are connected via Web. Let's try to visualize this using a small diagram:

To Summarize: A Client and a Server establishes a connection using HTTP protocol. Once the connection is established, Client sends across the request to the Server in the form of XML or JSON which both entities (Client and Server) understand. After understanding the request Server responds with appropriate data by sending back a Response.

To further understand this discussion, lets elaborate on the example of Weather details that we had talked about earlier. If we want to know about the Weather details of a place, we must tell the name of the Place for the Server to tell us the Weather details of that place.

When you *Request* for the weather details, you specify the **place name** in the Request.

In turn, the **Server** send the **Response** back. This response contains the actual Temperature of the city.

Here is a small step by step diagram to illustrate this process.

3.Explain concept of ports and sockets.

PORT:

A host is a machine which is connected to an IP network. A host can be identified by a host name (such as scar.dcs.ed.ac.uk) which translates into a numeric IP address (such as 129.215.216.18). The textual form of the name is much easier to remember than the numeric address and can be translated into the numeric address using a service called the Domain Name Service (DNS).

A port number is used to communicate with a process which is running on a host. Particular services provided by a host will be associated with a particular port number. For example, port 25 is standardly used for electronic mail and port 80 is used for Web communication using HTTP. On Unix systems (including Linux) the meanings of the assigned ports can be found by examining the file /etc/services. Together, a host name (or IP address) and a port number uniquely identify a partic- ular process running on a particular machine on the network. An analogy is that the IP address is rather like a telephone number, connecting you to a particular location, and the port number is rather like an telephone extension number, connecting you to a particular telephone receiver at that location. Together a host name and a port number allow us to create a socket.

SOCKET :

A socket is a connection to another machine. A server will listen for connections on a particular port. When a client tries to connect to the same port a socket connection is established. The socket behaves as two pairs of an input stream and an output stream. Viewed from the server side, the input stream is read to get commands from the client and the output stream is used to write results back to the client. Viewed from the client side, the output stream is used to write commands to the server and the input stream is used to read the results from the server. Of course these pairs of streams match up to make a bi-directional communication channel between client and server. Keeping in mind that sockets are made up of input and output streams is useful be- cause we are always aware that input and output operations can fail. Communication between hosts can fail also, perhaps because we do not have permission to connect to a particular host or because we do not have permission to use a particular port.

4.what is Protocol – Meaning, definition, examples,

A protocol is a set of rules and guidelines for communicating data. Rules are defined for each step and process during communication between two or more computers. Networks have to follow these rules to successfully transmit data.

Protocols exist for several different applications. Examples include wired networking (e.g., Ethernet), wireless networking (e.g., 802.11ac), and Internet communication (e.g., IP). The Internet protocol suite, which is used for transmitting data over the Internet, contains dozens of protocols. These protocols may be broken up into four catagories:

1. **Link layer** - PPP, DSL, Wi-Fi, etc.
2. **Internet layer** - IPv4, IPv6, etc.
3. **Transport layer** - TCP, UDP, etc.
4. **Application layer** - HTTP, IMAP, FTP, etc.

5.HTTP protocol – meaning, ?

The Hypertext Transfer Protocol (HTTP) is an application-level protocol for distributed, collaborative, hypermedia information systems. This is the foundation for data communication for the World Wide Web (i.e. internet) since 1990. HTTP is a generic and stateless protocol which can be used for other purposes as well using extensions of its request methods, error codes, and headers.

Basically, HTTP is a TCP/IP based communication protocol, that is used to deliver data (HTML files, image files, query results, etc.) on the World Wide Web. The default port is TCP 80, but other ports can be used as well. It provides a standardized way for computers to communicate with each other. HTTP specification specifies how clients' request data will be constructed and sent to the server, and how the servers respond to these requests.

Basic Features

There are three basic features that make HTTP a simple but powerful protocol:

- **HTTP is connectionless:** The HTTP client, i.e., a browser initiates an HTTP request and after a request is made, the client waits for the response. The server processes the request and sends a response back after which client disconnect the connection. So client and server knows about each other during current request and response only. Further requests are made on new connection like client and server are new to each other.
- **HTTP is media independent:** It means, any type of data can be sent by HTTP as long as both the client and the server know how to handle the data content. It is required for the client as well as the server to specify the content type using appropriate MIME-type.
- **HTTP is stateless:** As mentioned above, HTTP is connectionless and it is a direct result of HTTP being a stateless protocol. The server and client are aware of each other only during a current request. Afterwards, both of them forget about each other. Due to this nature of the protocol, neither the client nor the browser can retain information between different requests across the web pages.

6.http protocol request and response header formats, ?

HTTP messages are how data is exchanged between a server and a client. There are two types of messages: *requests* sent by the client to trigger an action on the server, and *responses*, the answer from the server.

HTTP messages are composed of textual information encoded in ASCII, and span over multiple lines. In HTTP/1.1, and earlier versions of the protocol, these messages were openly sent across the connection. In HTTP/2, the once human-readable message is now divided up into HTTP frames, providing optimization and performance improvements.

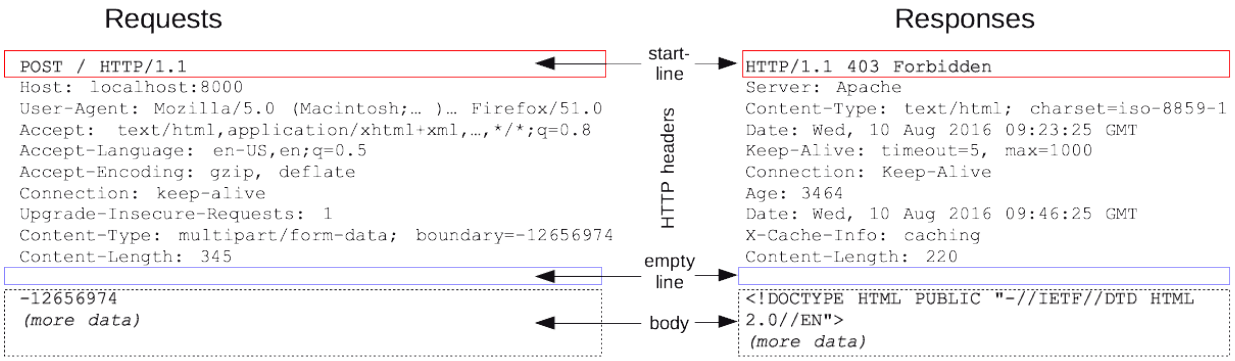
Web developers, or webmasters, rarely craft these textual HTTP messages themselves: software, a Web browser, proxy, or Web server, perform this action. They provide HTTP messages through config files (for proxies or servers), APIs (for browsers), or other interfaces.

The HTTP/2 binary framing mechanism has been designed to not require any alteration of the APIs or config files applied: it is broadly transparent to the user.

HTTP requests, and responses, share similar structure and are composed of:

1. A *start-line* describing the requests to be implemented, or its status of whether successful or a failure. This start-line is always a single line.
2. An optional set of *HTTP headers* specifying the request, or describing the body included in the message.
3. A blank line indicating all meta-information for the request have been sent.
4. An optional *body* containing data associated with the request (like content of an HTML form), or the document associated with a response. The presence of the body and its size is specified by the start-line and HTTP headers.

The start-line and HTTP headers of the HTTP message are collectively known as the *head* of the requests, whereas its payload is known as the *body*.




```
<html>
<head>
  <title>Echoing HTML Request Parameters</title>
</head>
<body>
  <h3>Choose an author:</h3>
  <form method="get">
    <input type="checkbox" name="author" value="Tan Ah Teck">Tan
    <input type="checkbox" name="author" value="Mohd Ali">Ali
    <input type="checkbox" name="author" value="Kumar">Kumar
    <input type="submit" value="Query">
  </form>

  <%
  String[] authors = request.getParameterValues("author");
  if (authors != null) {
  %>
    <h3>You have selected author(s):</h3>
    <ul>
  <%
      for (int i = 0; i < authors.length; ++i) {
  %>
        <li><%= authors[i] %></li>
  <%
      }
  %>
    </ul>
    <a href="<%= request.getRequestURI() %>">BACK</a>
  <%
  }
  %>
</body>
</html>
```

7. difference between stateless and state (state full) protocols.
Comparisons between Stateless and Stateful Protocol:

Stateless Protocol	Stateful Protocol
Stateless Protocol does not require the server to retain the server information or session details.	Stateful Protocol require server to save the status and session information.
In Stateless Protocol, there is no tight dependency between server and client.	In Stateful protocol, there is tight dependency between server and client
The Stateless protocol design simplify the server design.	The Stateful protocol design makes the design of server very complex and heavy.

Stateless Protocol	Stateful Protocol
Stateless Protocols works better at the time of crash because there is no state that must be restored, a failed server can simply restart after a crash.	Stateful Protocol does not work better at the time of crash because stateful server have to keep the information of the status and session details of the internal states.
Stateless Protocols handle the transaction very fastly.	Stateful Protocols handle the transaction very slowly.
Stateless Protocols are easy to implement in Internet.	Stateful protocols are logically heavy to implement in Internet.

8. list and explain status codes.?

HTTP response status codes indicate whether a specific HTTP request has been successfully completed. Responses are grouped in five classes: informational responses, successful responses, redirects, client errors, and servers errors.

Information responses

`100 Continue`
This interim response indicates that everything so far is OK and that the client should continue with the request or ignore it if it is already finished.

`101 Switching Protocol`
This code is sent in response to an `Upgrade` request header by the client, and indicates the protocol the server is switching to.

`102 Processing` (WebDAV)
This code indicates that the server has received and is processing the request, but no response is available yet.

`103 Early Hints`
This status code is primarily intended to be used with the `Link` header to allow the user agent to start preloading resources while the server is still preparing a response.

Successful responses

`200 OK`
The request has succeeded. The meaning of a success varies depending on the HTTP method:
GET: The resource has been fetched and is transmitted in the message body.
HEAD: The entity headers are in the message body.
PUT or POST: The resource describing the result of the action is transmitted in the message body.
TRACE: The message body contains the request message as received by the server

`201 Created`
The request has succeeded and a new resource has been created as a result of it. This is typically the response sent after a POST request, or after some PUT requests.

`202 Accepted`
The request has been received but not yet acted upon. It is non-committal, meaning that there is no way in HTTP to later send an asynchronous response indicating the outcome of processing the request. It is intended for cases where another process or server handles the request, or for batch processing.

`203 Non-Authoritative Information`

This response code means returned meta-information set is not exact set as available from the origin server, but collected from a local or a third party copy. Except this condition, 200 OK response should be preferred instead of this response.

`204 No Content`

There is no content to send for this request, but the headers may be useful. The user-agent may update its cached headers for this resource with the new ones.

`205 Reset Content`

This response code is sent after accomplishing request to tell user agent reset document view which sent this request.

`206 Partial Content`

This response code is used because of range header sent by the client to separate download into multiple streams.

`207 Multi-Status` (WebDAV)

A Multi-Status response conveys information about multiple resources in situations where multiple status codes might be appropriate.

`208 Multi-Status` (WebDAV)

Used inside a DAV: propstat response element to avoid enumerating the internal members of multiple bindings to the same collection repeatedly.

`226 IM Used` (HTTP Delta encoding)

The server has fulfilled a GET request for the resource, and the response is a representation of the result of one or more instance-manipulations applied to the current instance.

Redirection messages

`300 Multiple Choice`

The request has more than one possible response. The user-agent or user should choose one of them. There is no standardized way of choosingone of the responses.

`301 Moved Permanently`

This response code means that the URI of the requested resource has been changed permanently. Probably, the new URI would be given in the response.

`302 Found`

This response code means that the URI of requested resource has been changed *temporarily*. New changes in the URI might be made in the future. Therefore, this same URI should be used by the client in future requests.

`303 See Other`

The server sent this response to direct theclient to get the requested resource at another URI with aGET request.

`304 Not Modified`

This is used for caching purposes. It tells the client that the response has not been modified, so the client can continue to use the same cached version of the response.

`305 Use Proxy`

Was defined in a previous version of the HTTP specification to indicate that a requested response must be accessed by a proxy. It has been deprecated due to security concerns regarding in-band configuration of a proxy.

`306 unused`

This response code is no longer used, it is just reserved currently. It was used in a previous version of the HTTP 1.1 specification.

`307 Temporary Redirect`

The server sends this response to direct theclient to get the requested resource at another URI with same method that was used in the prior request. This has the same semantics as the `302 Found` HTTP response code, with the exception that the user agent *must not* change the

HTTP method used: If a `POST` was used in the first request, a `POST` must be used in the second request.

`308 Permanent Redirect`

This means that the resource is now permanently located at another URI, specified by the `Location:` HTTP Response header. This has the same semantics as the `301 Moved Permanently` HTTP response code, with the exception that the user agent *must not* change the HTTP method used: If a `POST` was used in the first request, a `POST` must be used in the second request.

Client error responses

`400 Bad Request`

This response means that server could not understand the request due to invalid syntax.

`401 Unauthorized`

Although the HTTP standard specifies "unauthorized", semantically this response means "unauthenticated". That is, the client must authenticate itself to get the requested response.

`402 Payment Required`

This response code is reserved for future use. Initial aim for creating this code was using it for digital payment systems,however this status code is used very rarely and no standard convention exists.

`403 Forbidden`

The clientdoes not have access rights to the content, i.e. they are unauthorized,so server is rejecting to give proper response. Unlike 401, the client's identity is known to the server.

`404 Not Found`

The server can not find requested resource. In the browser, this means the URL is not recognized. In an API, this can also mean that the endpoint is valid but the resource itself does not exist. Servers may also send this response instead of 403 to hide the existence of a resource from an unauthorized client. This response code is probably themost famous one due to its frequent occurence on theweb.

`405 Method Not Allowed`

The request method is known by the server but has been disabled and cannot be used. For example, an API may forbid DELETE-ing a resource. The two mandatory methods,`GET` and `HEAD`, must never be disabled and should not return this error code.

`406 Not Acceptable`

This response is sent when the web server, after performing server-driven content negotiation, doesn't find any content following the criteria given by the user agent.

`407 Proxy Authentication Required`

This is similar to 401 but authentication is needed to be done by a proxy.

`408 Request Timeout`

This response is sent on an idle connection by some servers, even without any previous request by the client. It means that the server would like to shut down this unused connection. This response is used much more since some browsers, like Chrome, Firefox 27+, or IE9, use HTTP pre-connection mechanisms to speed up surfing. Also note that some servers merely shut down the connection without sending this message.

`409 Conflict`

This response issent when a request conflicts with the current state of the server.

`410 Gone`

This response would be sent when the requested content has been permanently deleted from server, with no forwarding address. Clients are expected to remove their caches and links to the resource. The HTTP specification intends this status code to be used for "limited-time, promotional services". APIs should not feel compelled to indicate resources that have been deleted with this status code.

`411 Length Required`
: Server rejected the request because the `Content-Length` header field is not defined and the server requires it.

`412 Precondition Failed`
: The client has indicated preconditions in its headers which the server does not meet.

`413 Payload Too Large`
: Request entity is larger than limits defined by server; the server might close the connection or return an `Retry-After` header field.

`414 URI Too Long`
: The URI requested by the client is longer than the server is willing to interpret.

`415 Unsupported Media Type`
: The media format of the requested data is not supported by the server, so the server is rejecting the request.

`416 Requested Range Not Satisfiable`
: The range specified by the `Range` header field in the request can't be fulfilled; it's possible that the range is outside the size of the target URI's data.

`417 Expectation Failed`
: This response code means the expectation indicated by the `Expect` request header field can't be met by the server.

`418 I'm a teapot`
: The server refuses the attempt to brew coffee with a teapot.

`421 Misdirected Request`
: The request was directed at a server that is not able to produce a response. This can be sent by a server that is not configured to produce responses for the combination of scheme and authority that are included in the request URI.

`422 Unprocessable Entity` (WebDAV)
: The request was well-formed but was unable to be followed due to semantic errors.

`423 Locked` (WebDAV)
: The resource that is being accessed is locked.

`424 Failed Dependency` (WebDAV)
: The request failed due to failure of a previous request.

`425 Too Early`
: Indicates that the server is unwilling to risk processing a request that might be replayed.

`426 Upgrade Required`
: The server refuses to perform the request using the current protocol but might be willing to do so after the client upgrades to a different protocol. The server sends an `Upgrade` header in a 426 response to indicate the required protocol(s).

`428 Precondition Required`
: The origin server requires the request to be conditional. Intended to prevent the 'lost update' problem, where a client GETs a resource's state, modifies it, and PUTs it back to the server, when meanwhile a third party has modified the state on the server, leading to a conflict.

`429 Too Many Requests`
: The user has sent too many requests in a given amount of time ("rate limiting").

`431 Request Header Fields Too Large`
: The server is unwilling to process the request because its header fields are too large. The request MAY be resubmitted after reducing the size of the request header fields.

`451 Unavailable For Legal Reasons`
: The user requests an illegal resource, such as a web page censored by a government.

Server error responses

`500 Internal Server Error`

The server has encountered a situation it doesn't know how to handle.

`501 Not Implemented`

The request method is not supported by the server and cannot be handled. The only methods that servers are required to support (and therefore that must not return this code) are `GET`and `HEAD`.

`502 Bad Gateway`

This error response means that the server, while working as a gateway to get a response needed to handle the request, got an invalid response.

`503 Service Unavailable`

The server is not ready to handle the request. Common causes are a server that is down for maintenance or that is overloaded. Note that together with this response, a user-friendly page explaining the problem should be sent. This responses should be used for temporary conditions and the `Retry-After:` HTTPheader should, if possible, contain the estimated time before the recovery of the service. The webmaster must also take care about the caching-related headers that are sent along with this response, as these temporary condition responses should usually not be cached.

`504 Gateway Timeout`

This error response is given when the server is acting as a gateway and cannot get a response in time.

`505 HTTP Version Not Supported`

The HTTP version used in the request is not supported by the server.

`506 Variant Also Negotiates`

The server has an internal configuration error: the chosen variant resource is configured to engage in transparent content negotiation itself, and is therefore not a proper end point in the negotiation process.

`507 Insufficient Storage` (WebDAV)

The method could not be performed on the resource because the server is unable to store the representation needed to successfully complete the request.

`508 Loop Detected` (WebDAV)

The server detected an infinite loop while processing the request.

`510 Not Extended`

Further extensions to the request are required for the server to fulfill it.

`511 Network Authentication Required`

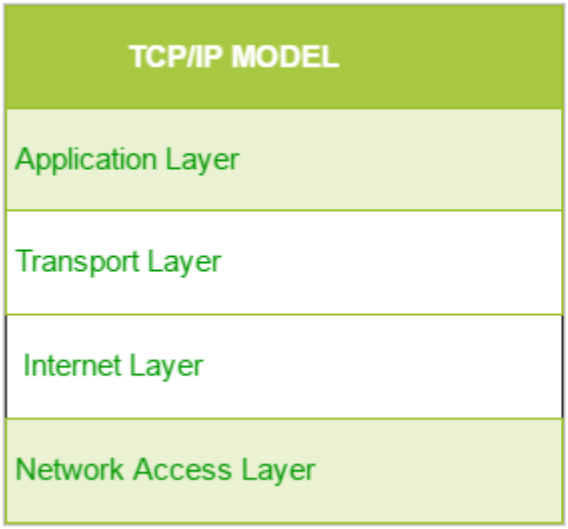
The 511 status code indicates that the client needs to authenticate to gain network access.

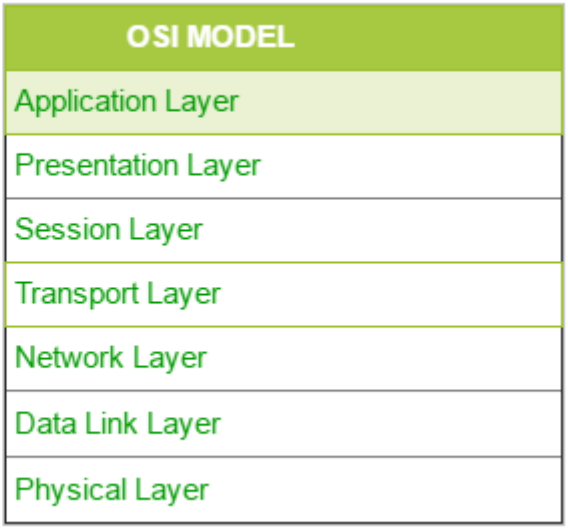
9. TCP /IP protocol ?

The **OSI Model** we just looked at is just a reference/logical model. It was designed to describe the functions of the communication system by dividing the communication procedure into smaller and simpler components. But when we talk about the TCP/IP model, it was designed and developed by Department of Defense (DoD) in 1960s and is based on standard protocols. It stands for Transmission Control Protocol/Internet Protocol. The **TCP/IP model** is a concise version of the OSI model. It contains four layers, unlike seven layers in the OSI model. The layers are:

1. Process/Application Layer
2. Host-to-Host/Transport Layer
3. Internet Layer
4. Network Access/Link Layer

The diagrammatic comparison of the TCP/IP and OSI model is as follows :





Difference between TCP/IP and OSI Model:

TCP/IP	OSI
TCP refers to Transmission Control Protocol.	OSI refers to Open Systems Interconnection.
TCP/IP has 4 layers.	OSI has 7 layers.
TCP/IP is more reliable	OSI is less reliable
TCP/IP does not have very strict boundaries.	OSI has strict boundaries
TCP/IP follow a horizontal approach.	OSI follows a vertical approach.
TCP/IP uses both session and presentation layer in the application layer itself.	OSI uses different session and presentation layers.
TCP/IP developed protocols then model.	OSI developed model then protocol.

The first layer is the Process layer on the behalf of the sender and Network Access layer on the behalf of the receiver. During this article, we will be talking on the behalf of the receiver.

1. Network Access Layer –

This layer corresponds to the combination of Data Link Layer and Physical Layer of the OSI model. It looks out for hardware addressing and the protocols present in this layer allows for the physical transmission of data.
We just talked about ARP being a protocol of Internet layer, but there is a conflict about declaring it as a protocol of Internet Layer or Network access layer. It is described as residing in layer 3, being encapsulated by layer 2 protocols.

2. Internet Layer –

This layer parallels the functions of OSI's Network layer. It defines the protocols which are responsible for logical transmission of data over the entire network. The main protocols residing at this layer are :

1. **IP –** stands for Internet Protocol and it is responsible for delivering packets from the source host to the destination host by looking at the IP addresses in the packet headers. IP has 2 versions:

IPv4 and IPv6. IPv4 is the one that most of the websites are using currently. But IPv6 is growing as the number of IPv4 addresses are limited in number when compared to the number of users.

2. **ICMP –** stands for Internet Control Message Protocol. It is encapsulated within IP datagrams and is responsible for providing hosts with information about network problems.
3. **ARP –** stands for Address Resolution Protocol. Its job is to find the hardware address of a host from a known IP address. ARP has several types: Reverse ARP, Proxy ARP, Gratuitous ARP and Inverse ARP.

3. Host-to-Host Layer –

This layer is analogous to the transport layer of the OSI model. It is responsible for end-to-end communication and error-free delivery of data. It shields the upper-layer applications from the complexities of data. The two main protocols present in this layer are :

1. **Transmission Control Protocol (TCP) –** It is known to provide reliable and error-free communication between end systems. It performs sequencing and segmentation of data. It also has acknowledgment feature and controls the flow of the data through flow control mechanism. It is a very effective protocol but has a lot of overhead due to such features. Increased overhead leads to increased cost.
2. **User Datagram Protocol (UDP) –** On the other hand does not provide any such features. It is the go-to protocol if your application does not require reliable transport as it is very cost-effective. Unlike TCP, which is connection-oriented protocol, UDP is connectionless.

4. Process Layer –

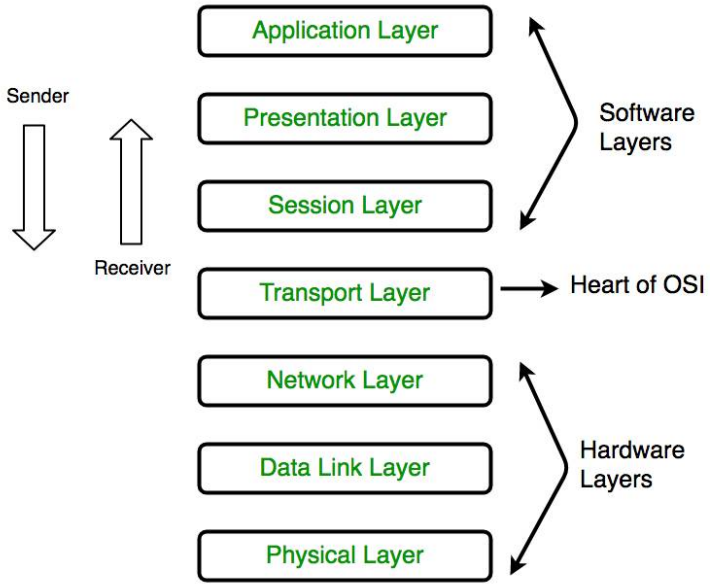
This layer performs the functions of top three layers of the OSI model: Application, Presentation and Session Layer. It is responsible for node-to-node communication and controls user-interface specifications. Some of the protocols present in this layer are: HTTP, HTTPS, FTP, TFTP, Telnet, SSH, SMTP, SNMP, NTP, DNS, DHCP, NFS, X Window, LPD. Have a look at Protocols in Application Layer for some information about these protocols. Protocols other than those present in the linked article are :

1. **HTTP and HTTPS –** HTTP stands for Hypertext transfer protocol. It is used by the World Wide Web to manage communications between web browsers and servers. HTTPS stands for HTTP-Secure. It is a combination of HTTP with SSL(Secure Socket Layer). It is efficient in cases where the browser need to fill out forms, sign in, authenticate and carry out bank transactions.
2. **SSH –** SSH stands for Secure Shell. It is a terminal emulations software similar to Telnet. The reason SSH is more preferred is because of its ability to maintain the encrypted connection. It sets up a secure session over a TCP/IP connection.
3. **NTP –** NTP stands for Network Time Protocol. It is used to synchronize the clocks on our computer to one standard time source. It is very useful in situations like bank transactions. Assume the following situation without the presence of NTP. Suppose you carry out a transaction, where your computer

reads the time at 2:30 PM while the server records it at 2:28 PM. The server can crash very badly if it's out of sync.

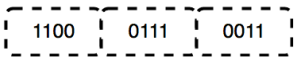
10. OSI Protocol ?

OSI stands for **Open Systems Interconnection**. It has been developed by ISO – '**International Organization of Standardization**', in the year 1974. It is a 7 layer architecture with each layer having specific functionality to perform. All these 7 layers work collaboratively to transmit the data from one person to another across the globe.



1. Physical Layer (Layer 1) :

The lowest layer of the OSI reference model is the physical layer. It is responsible for the actual physical connection between the devices. The physical layer contains information in the form of **bits.** It is responsible for the actual physical connection between the devices. When receiving data, this layer will get the signal received and convert it into 0s and 1s and send them to the Data Link layer, which will put the frame back together.



The functions of the physical layer are :

1. **Bit synchronization:** The physical layer provides the synchronization of the bits by providing a clock. This clock controls both sender and receiver thus providing synchronization at bit level.
2. **Bit rate control:** The Physical layer also defines the transmission rate i.e. the number of bits sent per second.
3. **Physical topologies:** Physical layer specifies the way in which the different, devices/nodes are arranged in a network i.e. bus, star or mesh topolgy.
4. **Transmission mode:** Physical layer also defines the way in which the data flows between the two connected devices. The various transmission modes possible are: Simplex, half-duplex and full-duplex.

* Hub, Repeater, Modem, Cables are Physical Layer devices.
** Network Layer, Data Link Layer and Physical Layer are also known as **Lower Layers** or **Hardware Layers**.

2. Data Link Layer (DLL) (Layer 2) :

The data link layer is responsible for the node to node delivery of the message. The main function of this layer is to make sure data transfer is error free from one node to another, over the physical layer. When a packet arrives in a network, it is the responsibility of DLL to transmit it to the Host using its MAC address.
Data Link Layer is divided into two sub layers :

1. Logical Link Control (LLC)
2. Media Access Control (MAC)

The packet received from Network layer is further divided into frames depending on the frame size of NIC(Network Interface Card). DLL also encapsulates Sender and Receiver's MAC address in the header.

The Receiver's MAC address is obtained by placing an ARP(Address Resolution Protocol) request onto the wire asking "Who has that IP address?" and the destination host will reply with its MAC address.

The functions of the data Link layer are :

1. **Framing:** Framing is a function of the data link layer. It provides a way for a sender to transmit a set of bits that are meaningful to the receiver. This can be accomplished by attaching special bit patterns to the beginning and end of the frame.
2. **Physical addressing:** After creating frames, Data link layer adds physical addresses (MAC address) of sender and/or receiver in the header of each frame.
3. **Error control:** Data link layer provides the mechanism of error control in which it detects and retransmits damaged or lost frames.
4. **Flow Control:** The data rate must be constant on both sides else the data may get corrupted thus , flow control coordinates that amount of data that can be sent before receiving acknowledgement.

5. **Access control:** When a single communication channel is shared by multiple devices, MAC sub-layer of data link layer helps to determine which device has control over the channel at a given time.

** Packet in Data Link layer is referred as* **Frame**.
*** Data Link layer is handled by the NIC (Network Interface Card) and device drivers of host machines.*
**** Switch & Bridge are Data Link Layer devices.*

3. Network Layer (Layer 3) :

Network layer works for the transmission of data from one host to the other located in different networks. It also takes care of packet routing i.e. selection of the shortest path to transmit the packet, from the number of routes available. The sender & receiver's IP address are placed in the header by network layer.
The functions of the Network layer are :

1. **Routing:** The network layer protocols determine which route is suitable from source to destination. This function of network layer is known as routing.
2. **Logical Addressing:** In order to identify each device on internetwork uniquely, network layer defines an addressing scheme. The sender & receiver's IP address are placed in the header by network layer. Such an address distinguishes each device uniquely and universally.

* *Segment* in Network layer is referred as **Packet**.

** Network layer is implemented by networking devices such as routers.

4. Transport Layer (Layer 4) :

Transport layer provides services to application layer and takes services from network layer. The data in the transport layer is referred to as *Segments*. It is responsible for the End to End delivery of the complete message. Transport layer also provides the acknowledgment of the successful data transmission and re-transmits the data if an error is found.
• At sender's side:
Transport layer receives the formatted data from the upper layers, performs **Segmentation** and also implements **Flow & Error control** to ensure proper data transmission. It also adds Source and Destination port number in its header and forwards the segmented data to the Network Layer.
Note: The sender need to know the port number associated with the receiver's application.
Generally, this destination port number is configured, either by default or manually. For example, when a web application makes a request to a web server, it typically uses port number 80, because this is the default port assigned to web applications. Many applications have default port assigned.
• At receiver's side:
Transport Layer reads the port number from its header and forwards the Data which it has received to the respective application. It also performs sequencing and reassembling of the segmented data.

The functions of the transport layer are :

1. **Segmentation and Reassembly:** This layer accepts the message from the (session) layer , breaks the message into smaller units . Each of the segment produced has a header associated with it. The transport layer at the destination station reassembles the message.
2. **Service Point Addressing:** In order to deliver the message to correct process, transport layer header includes a type of address called service point address or port address. Thus by specifying this address, transport layer makes sure that the message is delivered to the correct process.

The services provided by transport layer :

1. **Connection Oriented Service:** It is a three-phase process which include
   – Connection Establishment
   – Data Transfer
   – Termination / disconnection
   In this type of transmission, the receiving device sends an acknowledgment, back to the source after a packet or group of packet is received. This type of transmission is reliable and secure.
2. **Connection less service:** It is a one phase process and includes Data Transfer. In this type of transmission, the receiver does not acknowledge receipt of a packet. This approach allows for much faster communication between devices. Connection oriented Service is more reliable than connection less Service.

** Data in the Transport Layer is called as* **Segments**.
*** Transport layer is operated by the Operating System. It is a part of the OS and communicates with the Application Layer by making system calls.*
Transport Layer is called as **Heart of OSI** *model.*

5. Session Layer (Layer 5) :

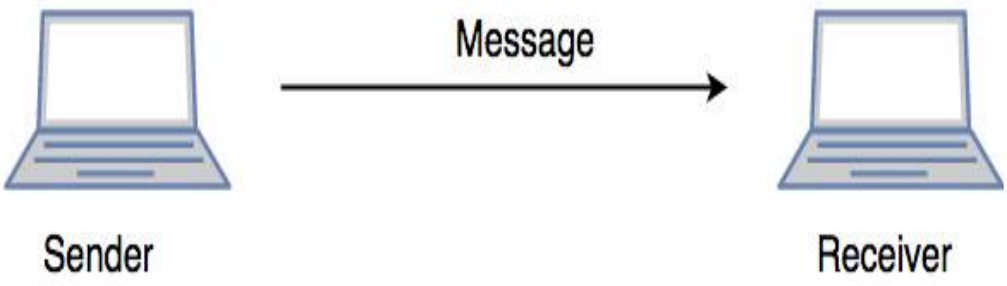
This layer is responsible for establishment of connection, maintenance of sessions, authentication and also ensures security.
The functions of the session layer are :

1. **Session establishment, maintenance and termination:** The layer allows the two processes to establish, use and terminate a connection.
2. **Synchronization :** This layer allows a process to add checkpoints which are considered as synchronization points into the data. These synchronization point help to identify the error so that the data is re-synchronized properly, and ends of the messages are not cut prematurely and data loss is avoided.
3. **Dialog Controller :** The session layer allows two systems to start communication with each other in half-duplex or full-duplex.

***All the below 3 layers(including Session Layer) are integrated as a single layer in TCP/IP model as "Application Layer".*
***Implementation of these 3 layers is done by the network application itself. These are also known as* **Upper Layers** *or* **Software Layers**.

SCENARIO:
Let's consider a scenario where a user wants to send a message through some Messenger

application running in his browser. The “Messenger” here acts as the application layer which provides the user with an interface to create the data. This message or so-called Data is compressed, encrypted (if any secure data) and converted into bits (0’s and 1’s) so that it can be transmitted.



6. Presentation Layer (Layer 6) :

Presentation layer is also called the **Translation layer**.The data from the application layer is extracted here and manipulated as per the required format to transmit over the network.
The functions of the presentation layer are :

1. **Translation :** For example, ASCII to EBCDIC.
2. **Encryption/ Decryption :** Data encryption translates the data into another form or code. The encrypted data is known as the cipher text and the decrypted data is known as plain text. A key value is used for encrypting as well as decrypting data.
3. **Compression:** Reduces the number of bits that need to be transmitted on the network.

7. Application Layer (Layer 7) :

At the very top of the OSI Reference Model stack of layers, we find Application layer which is implemented by the network applications. These applications produce the data, which has to be transferred over the network. This layer also serves as a window for the application services to access the network and for displaying the received information to the user.
Ex: Application – Browsers, Skype Messenger etc.
***Application Layer is also called as Desktop Layer.*

The functions of the Application layer are :

1. Network Virtual Terminal
2. FTAM-File transfer access and management
3. Mail Services
4. Directory Services

OSI model acts as a reference model and is not implemented in Internet because of its late invention. Current model being used is the TCP/IP model.