**SELF STUDY PROJECT**

1. **SOFTWARE DEVELOPMENT LIFE CYCLE(SDLC)**

The SDLC is the process involved in developing a Software from planning to the finishing state.

There are mainly 5 phases to a life cycle and each phase has Milestones, Deliverables and tasks associated with it.

A framework commonly used by Project Manager is Agile using a Scrum method. This ensures that all tasks, resources and communications are shared and obstacles and risks managed as they arise.

The steps are:

**1.Initiation Phase**

The initiation phase includes gathering information and producing the Project Charter. The Project charter simply contains information about the stakeholders, their level of involvement, and objectives. Previous versions and requirements to be met by the current software to be developed. The main deliverable here is the Project Charter.

**2. Planning phase**

contains the information about how the software to be developed. Information such as scope, budget, time, requirements, stakeholders communication, objectives, resources to be used and all necessary resources needed to carry out the project. RACI logs are created, RAID(Risk) logs and WBS are also created. All these will be included in what is termed as the Project Plan. All these are created setting Deliverables, tasks and milestones

**3. Execution phase**

This is where all the objectives and deliverables highlighted are put together. The requirements collected are now put into practise. This is where the Software product itself is being developed and care must be taken so that all the requirements are met by the development team.

**4. Monitoring and control phase**

This is where to ensure that developers are on task and executing effectively. All issues that arise are discussed and implemented. Risks that may arise managed and dealt with so as not to affect the project development. All logs updated and managed to ensure the project meets all its targets, on time and within budget.

**5. End phase**

The end Phase is when the Software developed is tested by outside testers and in-house testers to ensure that all bugs and issues are fixed for the final roll out. This phase also marks the end of the project by highlighting and reviewing what went well and could have been done well that could help improve the software future development and releases.

1. **LAMP AND LAMP STACK**

LAMP stands for Linux, Apache, MySQL, and PHP. Together, they provide a proven set of software for delivering high-performance web applications. Each component contributes essential capabilities to the stack:

* Linux: The operating system. Linux is a free and open source operating system (OS) that has been around since the mid-1990s. Today, it has an extensive worldwide user base that extends across industries. Linux is popular in part because it offers more flexibility and configuration options than some other operating systems.
* Apache: The web server. The Apache web server processes requests and serves up web assets via HTTP so that the application is accessible to anyone in the public domain over a simple web URL. Developed and maintained by an open community, Apache is a mature, feature-rich server that runs a large share of the websites currently on the internet.
* MySQL: The database. MySQL is an open source relational database management system for storing application data. With My SQL, you can store all your information in a format that is easily queried with the SQL language. SQL is a great choice if you are dealing with a business domain that is well structured, and you want to translate that structure into the backend. MySQL is suitable for running even large and complex sites. PHP: The programming language. The PHP open source scripting language works with Apache to help you create dynamic web pages. You cannot use HTML to perform dynamic processes such as pulling data out of a database. To provide this type of functionality, you simply drop PHP code into the parts of a page that you want to be dynamic.

PHP is designed for efficiency. It makes programming easier—and a bit more fun—by allowing you to write new code, hit refresh, and immediately see the resulting changes without the need for compiling. If you prefer, you can swap out PHP in favor of Perl or the increasingly popular Python language.

1. **Chmod and Cmon COMMANDS**
2. Chmod command is used to change permission for files and folders

9 dashes are used, First 3 for user, second 3 are for group and third 3 are for everyone. Commands abbreviations are: w=write, r=read, x=eXecute,u=user, g=group, o=others, +=add permission, -=remove permission

1. Chown command is used to change users/ or groups ownership of a given file or folder. Syntax= chown username FileorDirectory ( To change username only).

**4.TCP AND UDP**

Both of these are connection protocols. Out of these two, TCP is a connection-oriented one while the UDP is connectionless. A major difference between them is their speed. UDP is much faster than TCP. It has a much more effective protocol because it is simpler and faster. On the other hand, TCP allows retransmission of data packets (lost ones)- something that the UDP doesn’t offer. Another major difference between TCP and UDP is that UDP does not work on end-to-end communications. Conversely, TCP delivers data in the intended order to the server from the user (and vice versa). In fact, UDP also does not check a receiver’s readiness.

Before we discover the difference between TCP and UDP in further detail, let us know a bit more about them.

## What is TCP?

TCP is an abbreviation for Transmission Control Protocol. It is a connection-oriented protocol. It means that after establishing a stable connection, one can easily transmit data in two directions. TCP comes with built-in systems. They help in checking errors and guarantee the delivery of data in the same order of sending. Thus, TCP is the perfect protocol for transferring information like data files, images, web pages, and more.

While this protocol is inherently reliable, the feedback mechanisms working with it also result in a larger overhead. It translates to greater use of the bandwidth available in your network.

## What is UDP?

UDP is an abbreviation for User Datagram Protocol. It is a connectionless, simple Internet protocol that requires no recovery and error-checking services. The use of UDP causes zero overhead for opening, maintaining, or terminating any connection. It sends data continuously to any recipient, irrespective of whether they receive it or not.

UDP isn’t an ideal protocol for viewing a webpage, sending an email, or downloading any file. But one can largely prefer it in cases of real-time communications such as broadcasts or multi-task network transmission.

## Difference Between TCP and UDP

| **Parameter** | **TCP** | **UDP** |
| --- | --- | --- |
| Definition | TCP is a communication-based protocol. One can use it for the transmission of data over the network between systems. The data transmission occurs in the form of packets.  TCP includes error-checking techniques, guarantees data delivery, and maintains the order of data and information packets. | UDP is similar to the TCP protocol. But it does not guarantee data recovery and error-checking services.  If a user deploys this protocol, the data will get continuously sent, irrespective of any issues with the receiver. |
| Design | This protocol is connection-oriented. | This protocol is connectionless. |
| Transmission of Data | Data transmission in TCP occurs in a particular sequence. It means that the data packets arrive in the intended order at the receiver’s end. | Sequencing of data does not occur in the case of UDP. It means that a user can implement ordering only by managing it by the application layer. |
| Speed | TCP is comparatively slower than UDP. | UDP is faster as compared to TCP. |
| Efficiency | TCP is less efficient as compared to UDP. | UDP is more efficient as compared to TCP. |
| Retransmission | It is possible to retransmit data in TCP- just in case any packet is lost in the way, and a user needs to resend it. | It is not possible to retransmit data packets in UDP in the same way TCP does. |
| Status of Connection | TCP requires a very established connection for data transmission. One needs to close the connection once the transmission of data is complete. | UCP is a connectionless protocol. So it doesn’t require overhead to open, maintain, or terminate a connection. |
| Guarantee of Delivery | TCP guarantees data delivery to the destination receiver/router. | UDP does not offer any guarantee regarding data delivery to the destination receiver/router. |
| Sequencing of Data | TCP is capable of sequencing data. It rearranges the data packets in a specific order. | UDP is incapable of sequencing data. It has no fixed order, and all the packets remain independent of each other. |
| Size of Header | The size of a Header in TCP is 20 bytes. | The size of a Header in UDP is 8 bytes. |
| Checking of Errors | It offers an extensive acknowledgment of data and error checking. | It follows basic mechanisms of data checking like checksums. |
| Broadcasting | TCP does not support broadcasting. | UDP supports broadcasting. |
| Transferring Method | TCP reads data using the byte system. Every message transmits to the segment boundaries. | UDP packets have defined boundaries. It sends every packet individually and checks for the integrity of data on its arrival. |
| Reliability | TCP guarantees data delivery to the destination route and offers support for error checking. Thus, it is more reliable as compared to the UDP protocol. | UDP offers support for only basic error checking using the checksum data blocks. It also doesn’t guarantee data delivery to the destination as compared to that of TCP. |
| Optimal Use | Mostly HTTP, HTTPS, POP, SMTP, FTP, etc., utilize the TCP protocol. | Mostly DNS, VoIP, media streaming, video conferencing systems, etc., utilize the UDP protocol. |
| Weight | The TCP protocol is heavy-weight. It needs a total of three data packets for the setting up of a socket connection prior to sending any user data. | The UDP protocol is lightweight. No ordering of messages, tracking connections, etc., are present. |
| Acknowledgment segments | TCP has Acknowledgement segments. | UDP does not have any Acknowledgement segments. |
| Handshake Protocol | TCP uses a handshake protocol for establishing connections like SYN-ACK, SYN, ACK, etc. | UDP uses no handshake protocol since it is connectionless. |

| Pros | * TCP lets users set up/establish a connection between various types of computers. * It offers support for many routing protocols. * It does not depend on the operating system for operation. * Internetworking is possible between organizations using the TCP protocol. * The TCP/IP model has a client-server architecture that is highly scalable. * TCP supports various routing protocols. | * UDP never restricts the users to a communication model that is connection-based. * The startup latency in any distributed application with UDP remains low. * It also offers multicast and broadcast transmission. * The recipient of UDP packets can unmanage them. It also includes the block boundaries. * UDP can also make up for data loss. * It offers small transactions like the DNS lookup. * UDP is bandwidth-intensive. Thus, it endures packet loss. |
| --- | --- | --- |
| Cons | * A user cannot ask for multicast or broadcast transmission with TCP. * It has no block boundaries. So, a user needs to create this on their own. * It offers various features that a user may not require. As a result, it wastes time, effort, or bandwidth of the operating network. * The transport layer of the TCP model offers zero guarantee of delivery of packets. * It is not very easy to replace a protocol in TCP/IP. * TCP does not offer a clear separation of interfaces, services, and protocols. | * A packet may not reach its intended receiver or may reach twice iN a UDP protocol. * It has no flow control and congestion control. Thus, the implementation of this protocol is a user application’s job. * The routers stay quite careless with the UDP protocol. In case of a collision, they never retransmit. * UDP suffers from packet loss, while UDP doesn’t. |