**Devops Cource**

**------------------------------------------------------------------------------------------------**

**DEMO**

**Course Details**

4 months

MON-FRI 07:15-08:45 AM IST

Slack community

Life time access to recordings, slack and QA

QA daily 6 hours

Resume sessions, Interview behaviour

ATS resume session

Interview preparation

scenario based, twisted, indirect

weekly quiz

Project based learning from DAY-1

daily notes and architecture diagrams

Multiple Environments

Articles

**Why DevOps is popular? can It be there in next 10 years**

DevOps is popular because it streamlines software development and deployment by integrating development and IT operations, leading to faster, more reliable software releases and improved collaboration. It's a cultural philosophy, practices, and tools that promote automation, continuous improvement, and feedback loops.

**DevOps vs DevSecOps vs AIOps/MLOps**

DevOps (Development + Operations)

**Goal:**

Bridge the gap between software development and IT operations to deliver software faster and more reliably.

**Key Focus Areas:**

Continuous Integration / Continuous Delivery (CI/CD)

Infrastructure as Code (IaC)

Automation of testing and deployment

Monitoring and logging

Collaboration between developers and operations teams

**Benefits:**

Faster release cycles

Improved collaboration

Higher deployment success rate

**DevSecOps (Development + Security + Operations)**

**Goal:**

Integrate security practices into the DevOps pipeline, making security a shared responsibility across the SDLC.

**Key Focus Areas:**

Shift-left security (early integration of security)

Automated security testing (SAST, DAST)

Compliance and governance

Secure CI/CD pipelines

Threat modeling and vulnerability management

**Benefits:**

Early detection of security issues

Reduced risk of breaches

Faster compliance

**Difference from DevOps:**

Security is embedded from the start, not bolted on at the end.

AIOps (Artificial Intelligence for IT Operations)

**Goal:**  
Use AI/ML to enhance IT operations by automating problem detection, response, and optimization.

**Key Focus Areas:**

Anomaly detection

Root cause analysis

Predictive analytics

Noise reduction in alerts

Automated remediation

**Benefits:**

Reduced downtime

Faster incident resolution

Smarter monitoring

**MLOps (Machine Learning Operations)**

**Goal:**  
Operationalize machine learning models to ensure they are deployed, monitored, and maintained effectively.

**Key Focus Areas:**

Model versioning and tracking

Automated training and deployment

Model monitoring and drift detection

Reproducibility and scalability

Collaboration between data scientists and operations

**Benefits:**

Reliable and scalable ML model deployment

Continuous improvement of models

Faster experimentation and feedback cycles

Why Our course is diff

Most courses focus on one area. We cover all major ops disciplines **development, security, automation, AI, and ML in operations.**

Devops is a

Recorded vs Live

Discipline and engaging

Updates

Eco system

Job portal

Who can take this course

IT professionals --> any technology, any experience

NON-IT

Career Gap

College students

Freshers

Placement assistance

**Certifications**

AWS solutions architect associate

Terraform associate

CKA

**AI replace DevOps:**

❌ **AI will not replace DevOps — but it will significantly transform it.**

**Laptop configuration:**

i5/i7 16GB

i3 8GB --> worst case

**CLASS-1**

**https://github.com/daws-84s**

**Today’s topics:**

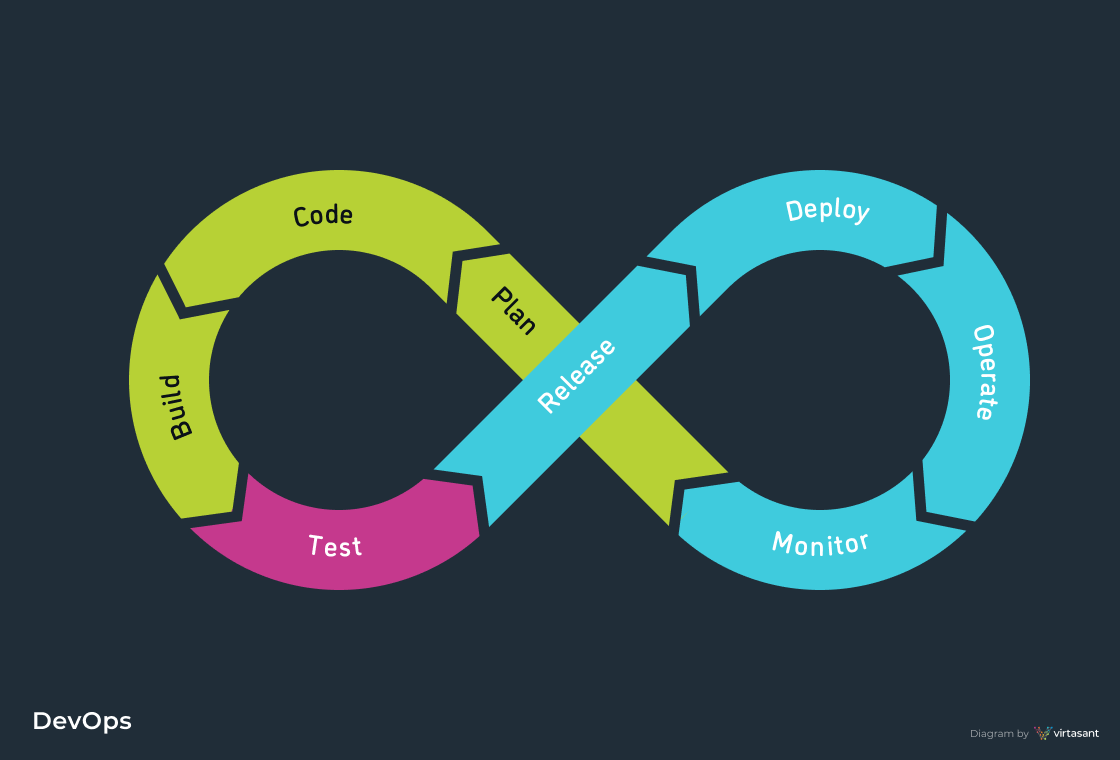
--------------------

What is DevOps?

SDLC

Waterfall vs Agile vs DevOps vs DevSecOps vs AiOps

**What is DevOps?**

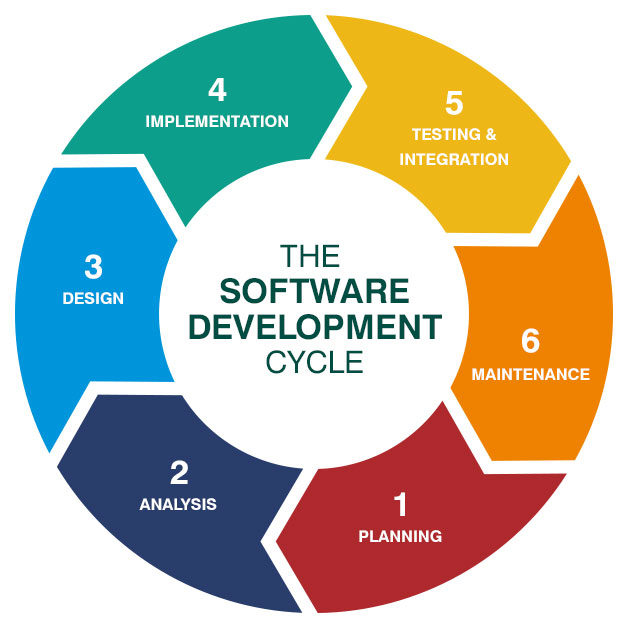


DevOps is the process of building, deploying and testing the code written by developer on the same day instead of doing after complete development.. we can acheive this using continous integration, continous deployment, continous delivery and continous testing.

We can do faster releases with less defect using DevOps... basically automation mindset to get the best results.

**SDLC (**Software Development Life Cycle)

**-------**



DevOps incorporates the Software Development Lifecycle (SDLC) by integrating development and operations to speed up software delivery, emphasizing collaboration, automation, and continuous feedback. It essentially streamlines the traditional SDLC process with DevOps principles, creating a continuous delivery pipeline.

Requirements analysis

Planning

Design --> General requirements to technical requirements

Implementation

Deployment

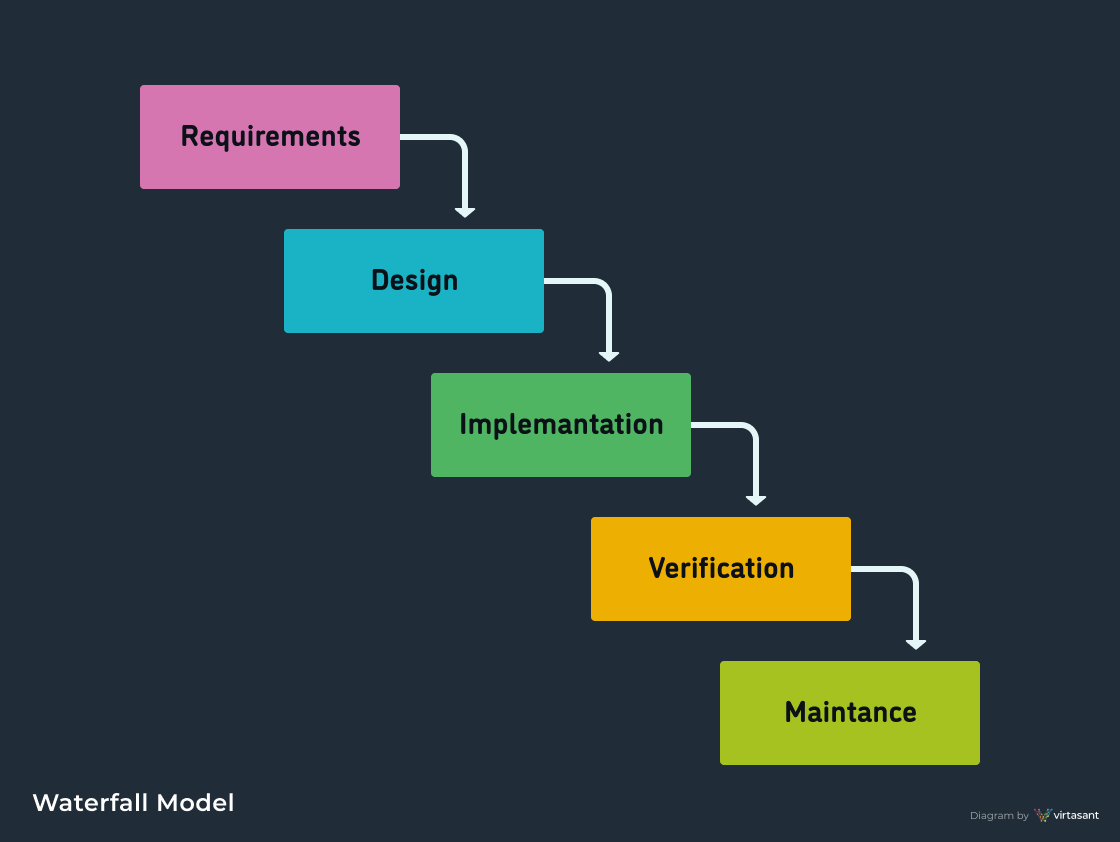
Testing

Maintaince

**Waterfall vs Agile vs DevOps vs DevSecOps vs AiOps**

**Waterfall vs Agile :**

**The First SDLC Methodology - The Waterfall Method - 1970s to 90s**



The Waterfall Method - 1970s to 90s

The first SDLC methodology to take hold in software development was the Waterfall method. Associated with [Winston W. Royce](https://en.wikipedia.org/wiki/Winston_W._Royce" \t "https://www.virtasant.com/blog/_blank), It was first [introduced](https://pragtob.wordpress.com/2012/03/02/why-waterfall-was-a-big-misunderstanding-from-the-beginning-reading-the-original-paper/" \t "https://www.virtasant.com/blog/_blank) in a paper he wrote and used it as an example of what a bad methodology looks like: "I believe in this concept, but the implementation described above is risky and invites failure." Despite his warnings and guidance, the Waterfall methodology quickly became the standard and stayed that way for over 20 years.

Waterfall is broken down into phases, and other modern methodologies can even pull from these phases and utilize them, these phases are:

* Requirement Analysis
* Planning
* Architectural Design
* Software Development
* Testing
* Deployment
* Maintenance

According to the Waterfall method, the software development process goes through all the SDLC phases with no overlapping and consists of a single development cycle. According to the fact that it is a linear-sequential life cycle model, any phase in the development process can begin only if the previous one is complete. Teams are large and everyone on the team (business analysts, architects, developers, tests, operations, etc.) all work within their own silos.

After the entire architecture, data structures, and functional designs are ready, the development team starts coding the software. Only after all code is written can integration and validation start. This means that the code is not tested before the Testing phase and only unit tests are executed during development.

Finally, the software finishes testing and is deployed to production and for the first time, where users are able to take it for a test drive. The Waterfall method can take several months or even years to complete, which means that if it doesn’t meet user expectations, changes are extremely slow and expensive. In many cases, defects never get fixed at all.

Likewise, due to the lack of feedback from customers or other stakeholders during the design and development process, it was quite common for Waterfall teams to build unnecessary or under-used features, leading to wasted time, effort, and money.

As technology leaders of the 1990s began realizing that the Waterfall method had a tendency to produce lengthy and costly business outcomes, they started seeking more flexible alternatives.

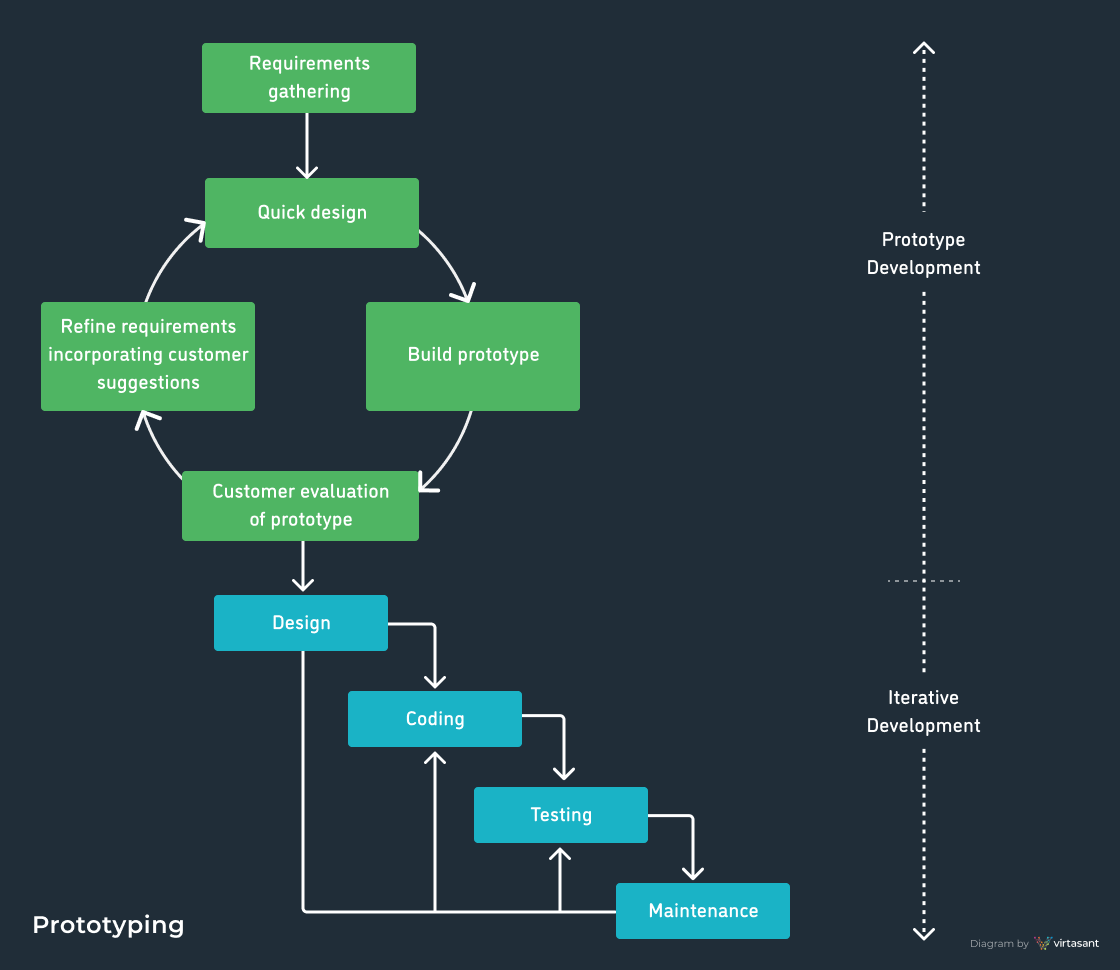
### Alternative Methodologies Come and Golean

Waterfall was showing its age, and it never really worked well, to begin with. As a result, pioneers in software developed novel methodologies aiming to either improve or replace Waterfall.

Methodologies like Prototyping, Iterative, Spiral, V-Shape, came and went, and more modern frameworks like Scrum, XP (Extreme Programming), and Kanban were developed around the same time as the standard we use today, Agile. In fact, a lot of folks that signed the Agile Manifesto were XP creators and users.

Understanding some of these now-outdated models helps us better understand how Waterfall transitioned into Agile:

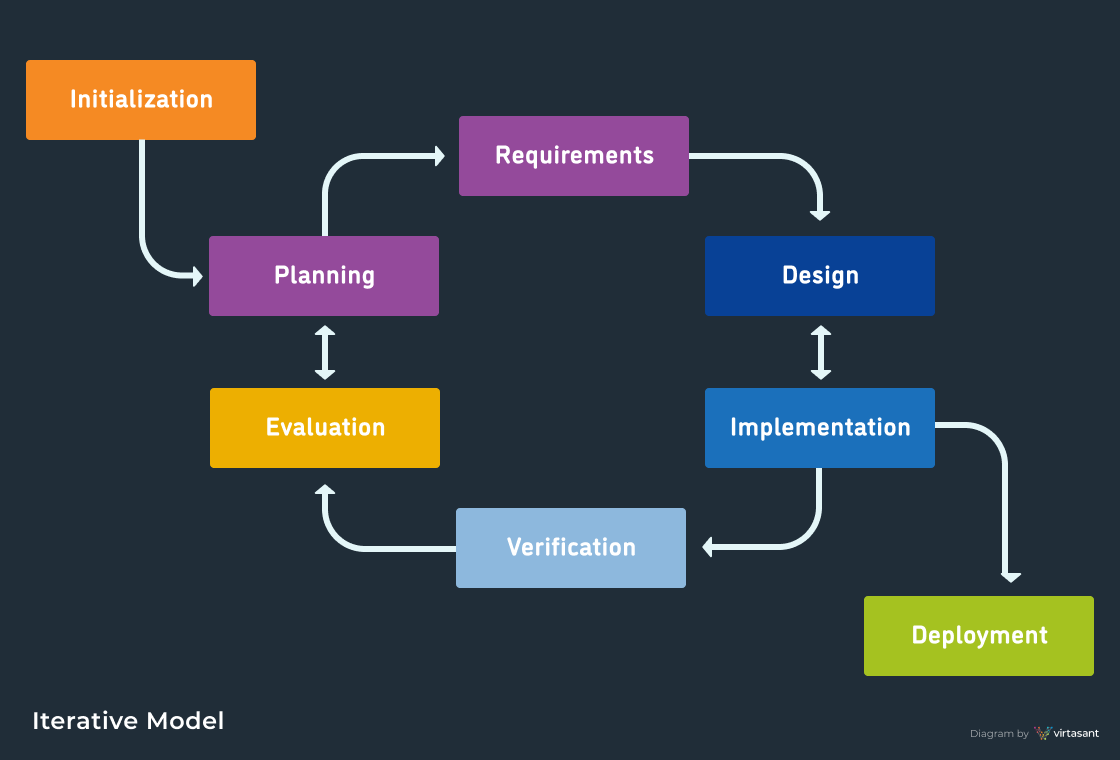
### The Prototyping Model



The Prototyping Model

An outdated methodology that is no longer in active use, it served its purpose as one of the earliest alternatives to Waterfall, dating back to the mid 1970s. The Prototype method revolves around the creation of a low fidelity prototype for the purposes of collecting early feedback from prospective users. From there, prototypes are evolved into final software requirements.

#### The Iterative Model

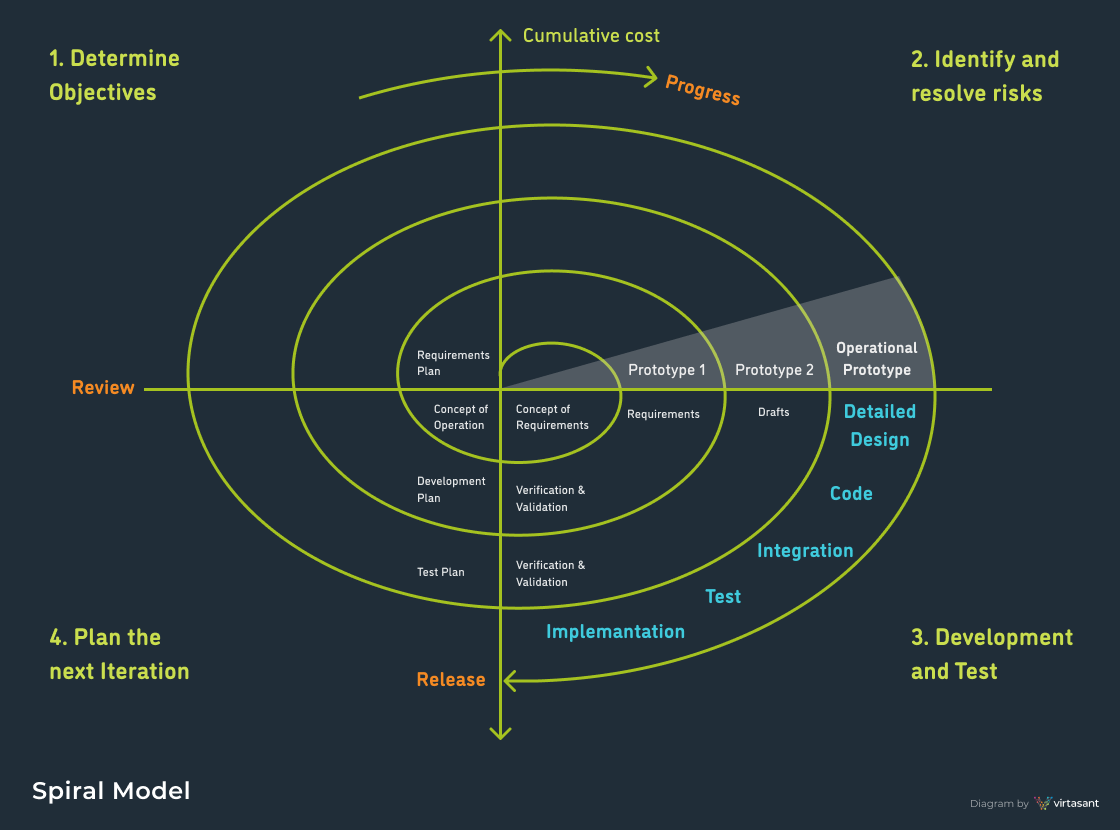


The Iterative Model

The Iterative methodology was an early precursor to Agile. It emphasized iterative and incremental action. Its earliest reported use was as part of NASA’s Project Mercury in the early 1960s.

With the Iterative Model, only the major requirements are known from the beginning. Based on these, the development team creates a quick and cheap first version of the software. Then, as additional requirements are identified, additional iterations of the software are designed and built.  Each iteration goes through all the phases of the SDLC and these cycles are repeated until completion. It was common for the team to work on several SDLC phases at the same time.

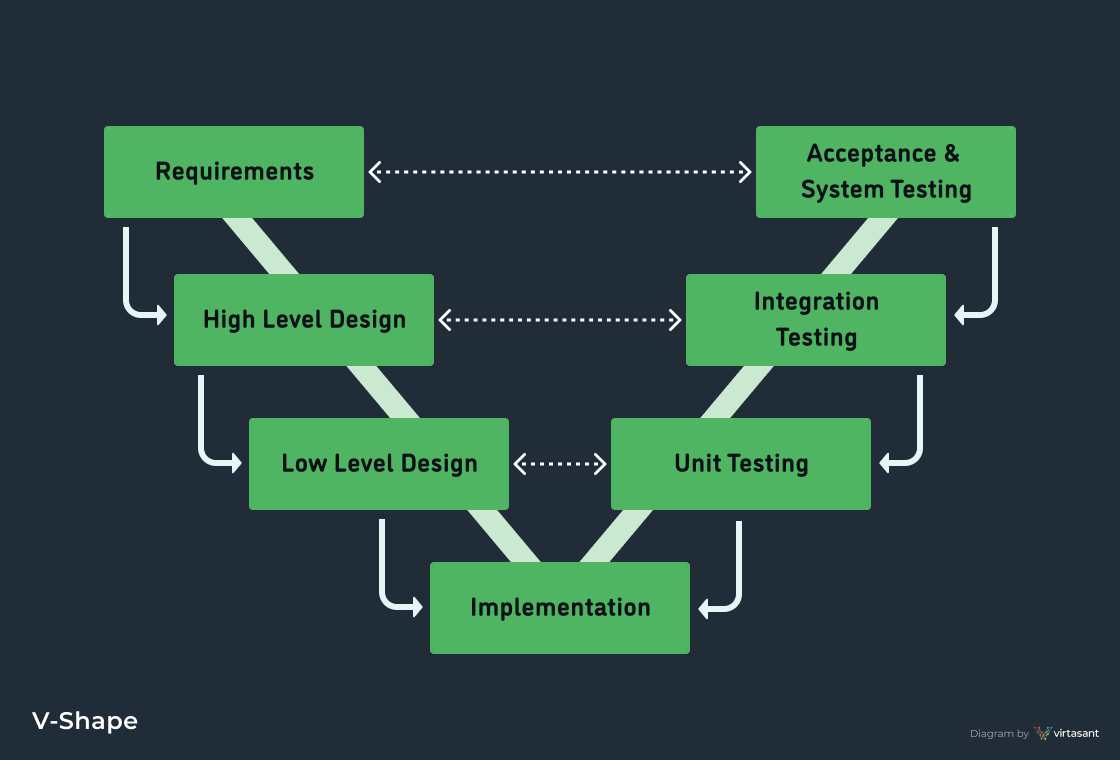
#### Spiral Model



Spiral Model

The Spiral Method is described by Barry Boehm in his 1986 paper “A Spiral Model of Software Development and Enhancement.” The Spiral Model boils down to a meta-model, which evaluates the specific risk profile of the project before recommending an approach that blends aspects of the other popular methodologies of the day, including Iterative and Waterfall. As such, it rejects a one size fits all approach to process model adoption.

#### V-Shape Model

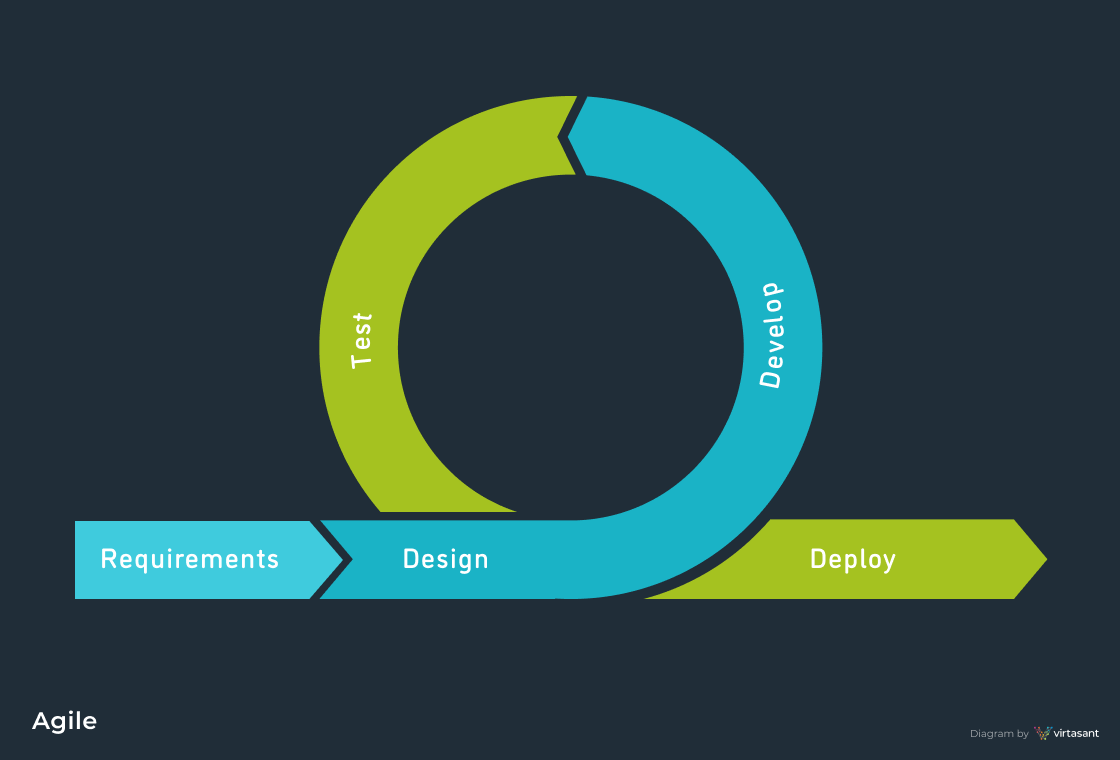


V-Shape Model

The V-Shape model is named after its two key concepts: Validation and Verification. In the Verification Phases, requirements and designs are created. Each Validation Phase has a corresponding Verification Phase, where testing and user acceptance occurs. These two phases are linked together by the Implementation (or coding) phase.

### The New Millenium: Agile Takes Over

With no single methodology presenting a suitable alternative to Waterfall, which was woefully too slow and risky, [17 pioneers in software engineering gathered](https://agilemanifesto.org/history.html) to create the [Agile “Software Development” Manifesto](https://agilemanifesto.org/) on February 11th, 2001.



Agile

Agile is the mainstream methodology used in modern software development, and expands its influence beyond coding into many aspects of product development, from ideation to customer experience.

The Agile methodology breaks a project down into multiple cycles, each passing through some or all of the SDLC phases. The focus is on people and how they work together to get the project done. Agile calls for continuous collaboration between team members and stakeholders with regular cycles of feedback and iteration.

#### The Agile Manifesto’s 4 Core Values

1. Individuals and interactions over processes and tools
2. Working software over comprehensive documentation
3. Customer collaboration over contract negotiation
4. Responding to change over following a plan

#### Agile Roles

Agile Roles assign responsibilities to members of the team. They are different than positions as a single person can take on multiple Agile roles depending on the scope of the project. Conversely, multiple people can share the same role.  
Here are some of the roles you could see in an Agile project:

* **Product Owner** - The Product Owner, also known as the “voice of the customer”, defines the product vision based on all insights, feedback, and ideas gathered. He or she is the owner of the product requirements and works closely with the development team to communicate the vision by documenting it in short narratives called User Stories. User Stories typically include a name, description, reference to any external documents, and an explanation of how to test the implementation. Product Owners often maintain a backlog of User Stories if there are too many to be executed concurrently.
* **Scrum Master**- Similar to a project manager, this role is all about making sure the team is following Agile principles, values, and processes.
* **Team Member** - All members of the development team have different skills and collaborate together to build functional software. Teams can include QA engineers, business analysts, designers, database engineers, and more depending on the project scope.

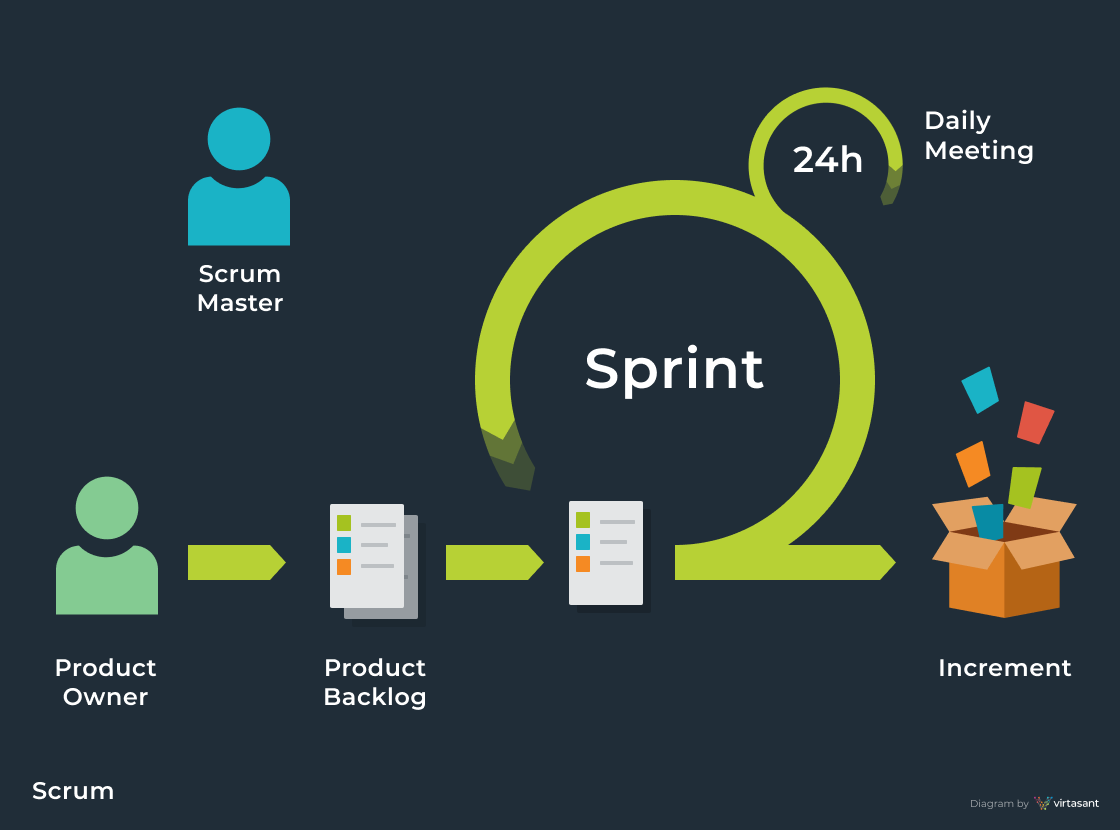
#### Advantages of Agile Methodology

* Deliver software well-tailored to an ever-growing understanding of customer demands
* Software is deployed more quickly and improved more regularly
* Better code hygiene including style, readability, and structuring
* Flexible and adaptable process enables pivots or changes mid-project
* Doesn’t require a complete list of requirements upfront
* Makes room to act on organizational learning as the project progresses
* Transparency and continuous communication with involved stakeholders

#### Agile Frameworks

Organizations can choose to adopt a single Agile framework or they can combine elements of multiple frameworks to suit the needs of the project and characteristics of the team. The most popular Agile frameworks are:

### Scrum



Scrum

Scrum is a very popular Agile framework characterized by continuous collaboration, frequent deliveries, and special development cycles called ‘Sprints’. Scrum revolves around the following checkpoints:

* **Planning meetings,** in which the team identifies and discusses the Sprint priorities.
* **Commitment meetings,** in which the team reviews the backlog of user stories to determine how much effort it involves and how much work can be done during the upcoming Sprint.
* **Daily standup meetings**, which are notably short meetings that ensure everyone is aligned. In this regard, each team member communicates updates on story status, blockers, or concerns.
* **Demo meetings**, which the team attends at the end of each Sprint to show the functionalities implemented during the current sprint to the Product Owner.
* **Retrospective meetings**, which are also hosted at the end of each Sprint to discuss lessons learned, what went well, and what needs improvement.

Scrum introduces the Scrum Master role to the Agile method. The Scrum Master’s job is to manage and improve processes, help the team stay authentic to Agile values, and focus on maximizing productivity. A good Scrum Master ensures that the process and progress are transparent to all stakeholders.

### Kanban



Kanban

Kanban is a scheduling system framework for the Agile-eque Lean methodology. It doesn’t have its roots in software development, but synergizes very well with Agile and has become a staple of Agile teams.

Kanban got its start in lean manufacturing, where Toyota applied the same “just in time” principles that supermarkets use to manage inventory stock levels based on customer demand. Kanban, meaning signboard in Japanese, uses cards to track and support the production system by visually showing the steps within the process and how long each step is taking using cards.

Kanban has a host of benefits when applied to Agile. You can limit WIP, focus on cycle time, and utilize just-in-time practices.

Kanban is sometimes compared to Scrum, which are similar in some ways, but are distinct frameworks:

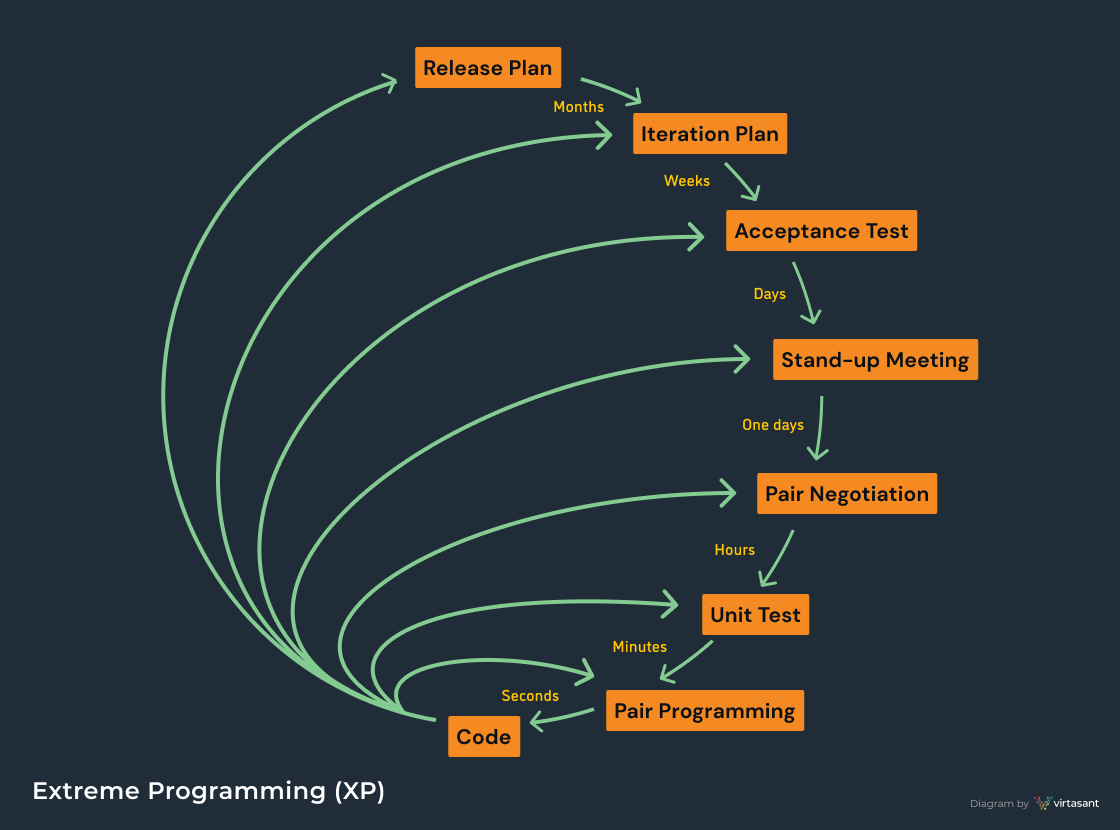
* Scrum utilizes fixed length Sprints cycles while Kanban is about continuous flow
* Scrum is role focused, while Kanban doesn’t utilize roles
* Scrum measures velocity, while Kanban focuses on cycle time

In the Kanban framework, the team creates a visual representation of their tasks and statuses by using sticky notes on a physical whiteboard or by using a dedicated software application. Tasks are moved through predefined stages such as To-Do, In Progress, In Review, or Complete.

A few examples of popular Kanban productivity apps:

* [Trello](https://trello.com/en-US" \t "https://www.virtasant.com/blog/_blank)
* [Jira](https://www.atlassian.com/software/jira" \t "https://www.virtasant.com/blog/_blank)
* [Proofhub](https://www.proofhub.com/" \t "https://www.virtasant.com/blog/_blank)
* [Zoho Projects](https://www.zoho.com/projects/" \t "https://www.virtasant.com/blog/_blank)
* [ZenHubs](https://www.zenhub.com/" \t "https://www.virtasant.com/blog/_blank)

### Extreme Programming



Extreme Programming

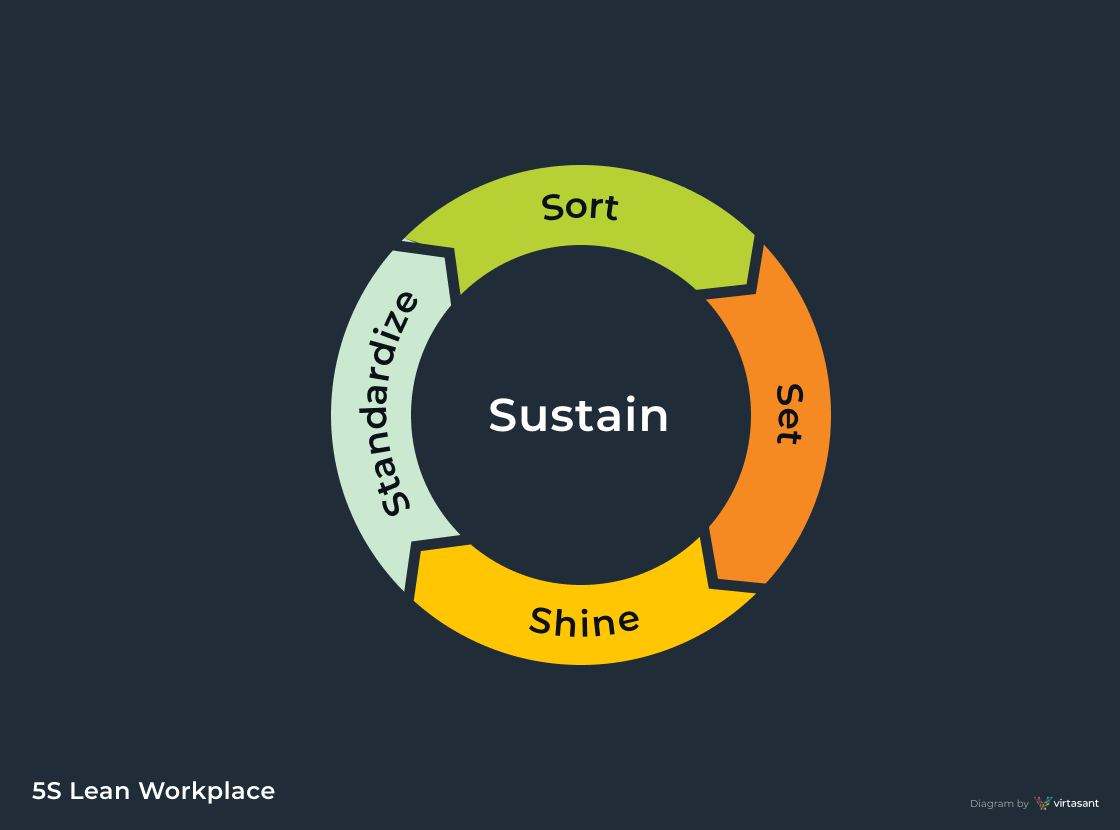
Extreme Programming (XP) is an Agile framework focused on project flexibility and writing high quality, well-tested code. The [official Extreme Programming website](http://www.extremeprogramming.org/when.html" \t "https://www.virtasant.com/blog/_blank) states that XP improves a software project in 5 key ways:

* Communication
* Simplicity
* Feedback
* Respect
* Courage

Extreme Programming is best known for the following:

* **Pair programming**is a technique where two programmers share the same workstation and create software together. One acts as the driver and the other one as the navigator, then they switch roles. When paired, code review can take place instantly, and defects are more likely to be identified and corrected immediately. Pair programming encourages mentorship, knowledge sharing, and learning. And while it may take more time to produce new code when two developers work on the same task, the resulting code is higher quality with less defects.
* **Unit and functional testing** are emphasized in XP. Tests are to be comprehensive and automated, reducing technical debt and ensuring code can confidently be validated and re-used.
* **Continuous communication** between programmers and stakeholders to gather and act upon their input, feedback, and change requests. XP requires an “extended development team” that may include business managers, customers, and other key stakeholders.

### Lean



Lean Worplace

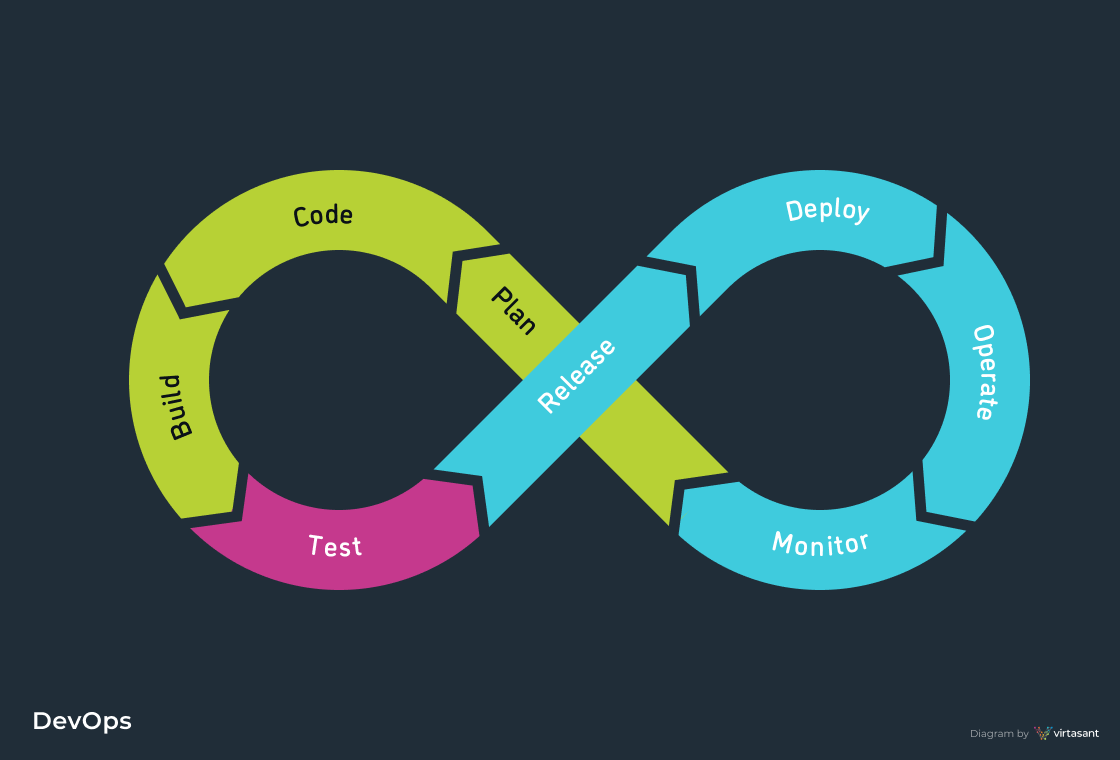
Lean isn’t a software development methodology. [Lean’s origins](https://en.wikipedia.org/wiki/Lean_manufacturing) go back to a manufacturing production method invented in the 1930s, officially given a name in the 80s, and more-formally defined in the 90s. Lean is a system that focuses on making more with less. Many have more-recently discovered that Lean works extremely well with software development, especially Agile.

While Agile focuses on delivering continuous value, the goal of Lean is to increase the speed and decrease the cost of product development. With Lean, the highest risks are wasted time and effort. Lean discourages multitasking and encourages team members to focus on what’s important in the present moment. By doing this, the waste associated with unnecessary documentation, meetings, or planning are eliminated.

Lean focuses on the following “just in time” principles:

* Eliminating waste in cost, scope, and scheduling
* Amplifying learning
* Taking decisions as late as possible
* Fast delivery
* Empowering the team
* Building integrity
* Optimizing the entire project

### DevOps



DevOps

DevOps is not technically an SDLC methodology but it does share the goal of maximizing software project success and includes Agile-inspired concepts.

On [Wikipedia](https://en.wikipedia.org/wiki/DevOps), DevOps is defined as “a set of practices that combines software development and IT operations. It aims to shorten the systems development life cycle and provide continuous delivery with high software quality. DevOps is complementary with Agile software development; several DevOps aspects came from Agile methodology.”

DevOps, just like Lean, can work alongside Agile to create an infrastructure that eliminates the barriers slowing development and delivery of the final software product. DevOps brings deployment and operation of the software fully into the Agile development process in the same way Agile brought testing and business analysis into software development. Ultimately, the team is empowered to be self-sufficient and take ownership of software development, shipping, and support. They use Continuous Delivery (CD) for frequent releases and to maintain a well-tested and high-quality codebase.

#### History of DevOps

The DevOps movement started around 2008. The constant pressure to make rapid changes plus the emergence of a new wave of infrastructure automation allowed non-specialists to enter the space and highlighted the need for cross-functional collaboration.

New expectations around delivering more-regular software changes were a big motivation for creating DevOps. Desktop applications were being replaced by web and mobile applications, and instead of delivering physical media (CDs or DVDs), companies began providing Software as a Service (SaaS) over the web. As the industry’s challenges evolved, DevOps offered a solution.

#### Advantages of DevOps

* Software development teams are self-sufficient; shipping and maintaining software without depending on the IT or technical operations teams.
* The deployment process is automated and optimized. A junior developer can learn to safely deploy, with less effort.
* Teams implement [Continuous Integration](https://aws.amazon.com/devops/continuous-integration" \t "https://www.virtasant.com/blog/_blank)/ [Continuous Delivery](https://aws.amazon.com/devops/continuous-delivery" \t "https://www.virtasant.com/blog/_blank)(CI/CD).
* Using the right tools, engineers save time on deployment so they can focus on coding.
* [Feedback loops](https://medium.com/@antweiss/understanding-feedback-loops-in-devops-e93b92b74bd1" \t "https://www.virtasant.com/blog/_blank) integrated throughout the entire process.

**Waterfall vs Agile vs DevOps vs DevSecOps vs AiOps**

Waterfall, Agile, DevOps, DevSecOps, and AIOps represent different approaches to software development and delivery, each with its own strengths and weaknesses. Waterfall is a sequential, rigid model, while Agile is an iterative, flexible one. DevOps emphasizes collaboration between development and operations teams, DevSecOps integrates security throughout the development lifecycle, and AIOps uses AI to optimize operations.

SMS --> School management system

Examination system

Stakeholders

=============

Parents

Students

Teachers

50 years back

Gurukuls --> waterfall(legacy model)

------------

year end exam

pass percentage --> 40%

DAY-1

Students --> Not serious

Teachers --> Not serious

Parents --> Yes

More exams --> process change

UNIT-TEST - I, II, III, Quarterly, IV, Half yearly, grand test

DAY-30

=======

students --> No

teachers --> serious

parents --> serious

pass/fail

25, 30, 35, 50, 70

90%

slip test --> daily test

========================

99%

students --> serious

teachers --> serious

parents --> serious

clients

developers

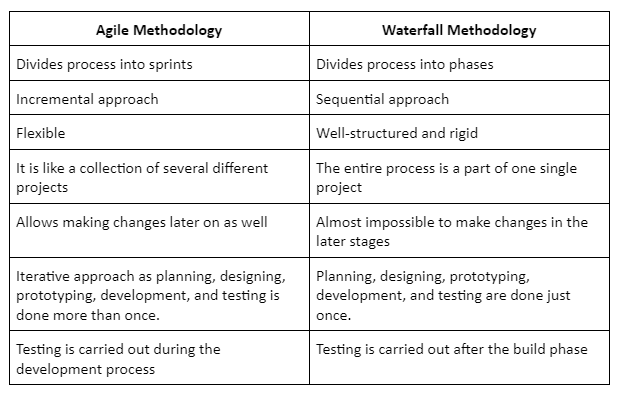
testers

operations

end users

## Agile vs Waterfall – Quick Comparison

Here’s a quick comparison of the **Waterfall and Agile methodologies** of project management.This detailed **project management methodologies comparison**will help you understand the **difference between Waterfall and Agile methodology** and which one is the most suitable project management methodology for you.



Based on this objective **project management methodologies comparison**, the difference between **Waterfall and Agile** is now clear and you can select the right methodology based on your needs and not just on popular buzz.

waterfall

===========

once requirements are fixed we can't change

2 years project to go for PROD

last 1 year

last 1 month deploy --> 100 defects

20 invalid defects -->

process change --> Agile

item --> 1 test --> 2 defects

1000 test --> 3 defects

Agile

===========

Final product --> multiple modules --> sprints

login and signup --> sprint-1 --> 30 days

orders

menu

payment

shipping

cart

reviews

employees

sprint-1

----------

clients UAT(user acceptance testing)

daily standups -->

95% --> 5 defects, 1 invalid defect

sprint-2 --> orders + 4 defects

15 days 15 days testing and deployment

Process change --> Agile with DevOps

=========================

Signup and Login

==================

DAY-1 --> enter first name, enter last name, enter DOB

deploy on the same day, test on the same day

10 defects --> 1 invalid defect

DevOps is the process of building, deploying and testing the code written by developer on the same day instead of doing after complete development.. we can acheive this using continous integration, continous deployment, continous delivery and continous testing.

We can do faster releases with less defect using DevOps... basically automation mindset to get the best results.

DEV, QA, SIT, UAT, PERF, PRE-PROD, PROD

min 2, max any number

Linux --> Linus torvalds

=================

Linux, a popular open-source operating system kernel, was developed by Linus Torvalds in 1991. Inspired by the UNIX operating system and MINIX, Torvalds aimed to create a free and flexible system for personal computers. Linux was initially distributed as source code, and later as bootable disk images, which sparked the creation of numerous Linux distributions to simplify the installation process.

Linux is a free, open-source operating system kernel, the core of a Linux-based operating system. It's a foundational component that manages hardware and software interaction. Linux is typically packaged with other software and libraries into what's called a "Linux distribution" or "distro," creating a complete, functional operating system.

UNIX --> Hardware and Software(Unix OS) together

CPU, RAM, Hard Disk, OS(Software)

Laptop --> IBM BIOS

Linux --> from the sratch using C language based on unix principles. He invented git to store this

windows

-----------

not open source

costly

not secure must install anti virus

slow

too many graphics

frequent restarts

tough to update/upgrade

Linux --> 9MB

Secure

high speed --> mostly text

no need of restarts

install or update packages is easy

free, community support

low resources

While the statement "90% of people are using Linux" is not entirely accurate, Linux is indeed widely used, particularly in specific sectors like cloud computing, server infrastructure, and embedded systems. In the public cloud, for example, Linux powers around 90% of the workload. Additionally, Linux dominates the supercomputer market, with 99% of these machines running on it.

AWS Accounts

================

Debit/Credit cards

International usage should be ON

free trail

**Class-2:**

**Today’s topics:**

1. Client server

2. Firewall

3. Authentication Method

4. Keys generation

5 lunch server

6 connecting servers

International usage enable

Credit/debit

Names and address should match bank records

What is computer?

==================

IP enabled device

In Linux, "computer" refers to the physical machine, or hardware, that runs the operating system. Linux itself is a software operating system (OS) that manages the hardware and software on a computer. The kernel, which is a core part of Linux, acts as the intermediary between the hardware and other software, allowing them to interact.

laptop == mobile == server == chip == tv ==> computer

server ==> only to host applications

RAM, OS, HD, Processor --> IP enabled device

Client Server architecture

Client-server architecture is a network model where clients request services from one or more servers. The server then processes these requests and provides the requested resources or services back to the client. This architecture is fundamental to many modern applications, including web browsing, email, and online banking.

Networking between computers

1. Network

2. Facebook application

serve --> lawyer serves to us, we are his client

ex : server : TCS

client – HDFC,SBI etc…..

Facebook is in Linux server, if problem comes we need to login to server..

https://www.joindevops.com

Windows:

Protocol: https

Port: 443

IP: DNS IP

Username and password --> authentication

Linux:

SSH --> secure shell

Port: 22

IP, Username and password/private-key

Firewall --> checks inbound and outbound traffic

inbound --> incoming traffic --> port number 22, 103.149.59.114

outbound --> outgoing traffic

authentication

==================

1. what you know --> username password

2. what you have --> keys, tokens, authentication, rsa token

3. what you are --> fingerprint, retina, palm, face

Server == box == node

lock = public

key = private

key pairs --> public key and private key

1. create key pair

2. create firewall

3. import public key to aws account

4. create server and attach public key

5. attach firewall to server

Git Bash --> Mini Linux in windows

SSH client

browser --> http/https client

Linux Server --> SSH Server

git bash == putty == super putty == mobaxterm == mac terminal = windows cmd

ssh clients

pwd --> present working directory

User Directory

C:\Users\siva --> windows format

/c/Users/siva --> linux format

windows --> not case sensitive Siva == siva == Siva 🡪 All are same

Linux --> case sensitive --> Siva != siva no space in linux

/c/devops/daws-84s == C:\devops\daws-84s

ssh-keygen -f <file-name>

ssh-keygen -f daws-84s

ssh-rsa long-random-code

Region --> us-east-1

Anywhere : 0.0.0.0/0 (add rule in aws)

98.81.70.98, 22, SSH, ec2-user, private-key

Ip,port,ptotocol and key

ssh -i <private-key> ec2-user@98.81.70.98

OS --> Redhat, Ubuntu, centos, fedore, suse, oracle linux, amazon linux, debian, rocky linux, kali, solaris

Linux is os or not --> Linux is kernel

OS --> to interact with hardware

Kernal --> heart and brain of OS

Kernel + UI == OS

Redhat --> kernel + user interface ==> Redhat OS

Debian --> Linux kernel + UI ==> Debian

99.9% same all Linux distros/flavours

Redhat --> Open source --> code is free

Redhat enterprise RHEL --> support

RHEL = AWS Linux = Centos = Fedora = Rocky Linux = AlmaLinux

command <options> <inputs>

uname – linux(kernel)

uname –a = all

uname --help 🡪 get to know all the information

uname –n 🡪 node name

uname –p \_\_. Processor

/home/ec2-user --> linux home directory

/c/users/siva --> windows home directory

cd --> change directory

cd .. --> one step back

ls --> list subdirectories

ls -l --> long list



**CLASS-3:**

**Today’s Topics:**

Linux commands

Vim editor

File permissions

/c/devops/daws-84s/daws-84s --> full path - give any folder

ssh -i daws-84s ec2-user@I.P

$ --> normal user

# --> root/admin/super user

/root --> root user home directory

command <options> <inputs>

/ --> root directory

ls -l --> long listing format in alphabetical order

ls -lr --> long listing format in reverse alphabetical order

ls -lt --> latest files on top

ls -ltr --> latest at bottom

ls -la --> all files including hidden files and folders

touch <file-name> --> creates empty file

touch devops.txt

cat > <file-name> --> type text, enter and ctrl+d

previous content will be replaced

cat >> <file-name> --> appends text to previous content

> --> usually called as redirection

mkdir <name> --> creates directory

rmdir --> remove empty directory

rm -f --> forcefully removes file

rm -rf --> recursively forcefully delete the files and folders inside too

CRUD --> create read update delete

cp <source> <destination> --> copy files/folders

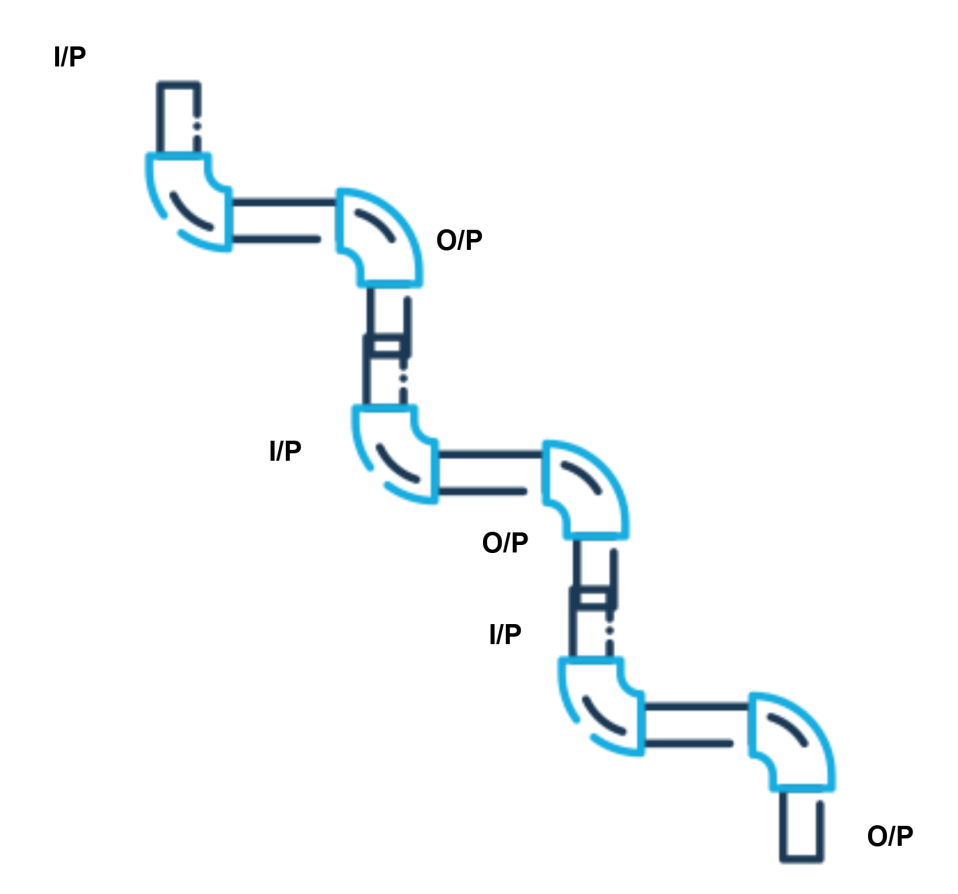
mv <source> <destination> --> cut and paste

wget <URL> --> downloads the file

wget <https://raw.githubusercontent.com/daws-84s/notes/refs/heads/main/session-02.txt>

curl <URL> --> shows on the screen

curl <https://raw.githubusercontent.com/daws-84s/notes/refs/heads/main/session-02.txt>



cat <file-name> | grep <word-to-search>

grep <word-to-search> <file>

echo https://raw.githubusercontent.com/daws-84s/notes/refs/heads/main/session-02.txt | cut -d "/" -f1

https:

[ec2-user@ip-172-31-18-60 ~]$ echo https://raw.githubusercontent.com/daws-84s/notes/refs/heads/main/session-02.txt | cut -d "/" -f9

session-02.txt

https://raw.githubusercontent.com/daws-84s/notes/refs/heads/main/session-02.txt

echo https://raw.githubusercontent.com/daws-84s/notes/refs/heads/main/session-02.txt | cut -d "/" -f1

awk command

------------

echo https://raw.githubusercontent.com/daws-84s/notes/refs/heads/main/session-02.txt | awk -F "/" '{print $1F}'

echo https://raw.githubusercontent.com/daws-84s/notes/refs/heads/main/session-02.txt | awk -F "/" '{print $NF}'

session-02.txt

log files --> tail -f <log-file>

find <where to search> -name <file-name>

vim --> visually improved

1. exit
2. uname
3. uname -a
4. uname --help
5. pwd
6. mkdir --help
7. cd ..
8. pwd
9. ls
10. ls --help
11. cd --help
12. ls --help
13. touch devops.txt
14. ls -l
15. ls -lrth
16. cat >devops.txt
17. ls -lrt
18. cat devops.txt
19. cat >devops.txt
20. ls -lrt
21. cat devops.txt
22. wget https://raw.githubusercontent.com/daws-84s/notes/refs/heads/main/session-02.txt
23. curl https://raw.githubusercontent.com/daws-84s/notes/refs/heads/main/session-02.txt~
24. curl https://raw.githubusercontent.com/daws-84s/notes/refs/heads/main/session-02.txt
25. echo "hello world"
26. echo https://raw.githubusercontent.com/daws-84s/notes/refs/heads/main/session-02.txt | cut -d "/" -f1
27. echo https://raw.githubusercontent.com/daws-84s/notes/refs/heads/main/session-02.txt | cut -d "/" -f9
28. echo https://raw.githubusercontent.com/daws-84s/notes/refs/heads/main/session-02.txt | cut -d "/" -f1-8
29. echo https://raw.githubusercontent.com/daws-84s/notes/refs/heads/main/session-02.txt | cut -d "/" -f1-2
30. echo https://raw.githubusercontent.com/daws-84s/notes/refs/heads/main/session-02.txt | awk -F "/" '{print $NF}'
31. awk -F ":" '{print $1F}' users
32. cp /etc/passwd users
33. ls -l
34. cat users
35. awk -F ":" '{print $1F}' users
36. awk -F ":" '{print $1F,$3F}' users
37. cd /etc/passwd
38. cat users | head
39. cat users | tail
40. cat users | head -n6
41. cat users | head -n2
42. find / -name "users"
43. sudo find / -name "users"
44. find . -name "users"
45. find . -name "\*se\*"

**CLASS - 04:**

**Todays topics:**

**Vim editor**

**User management**

vim editor

CAT >

CAT >>

In Linux, Vim stands for "Vi IMproved" and is a powerful, highly configurable, and widely used text editor. It's an enhanced version of the original Vi editor, known for its modal editing system and efficient keyboard-based operation. Vim is primarily used for editing text files, including code, scripts, and configuration files.

Connect the server

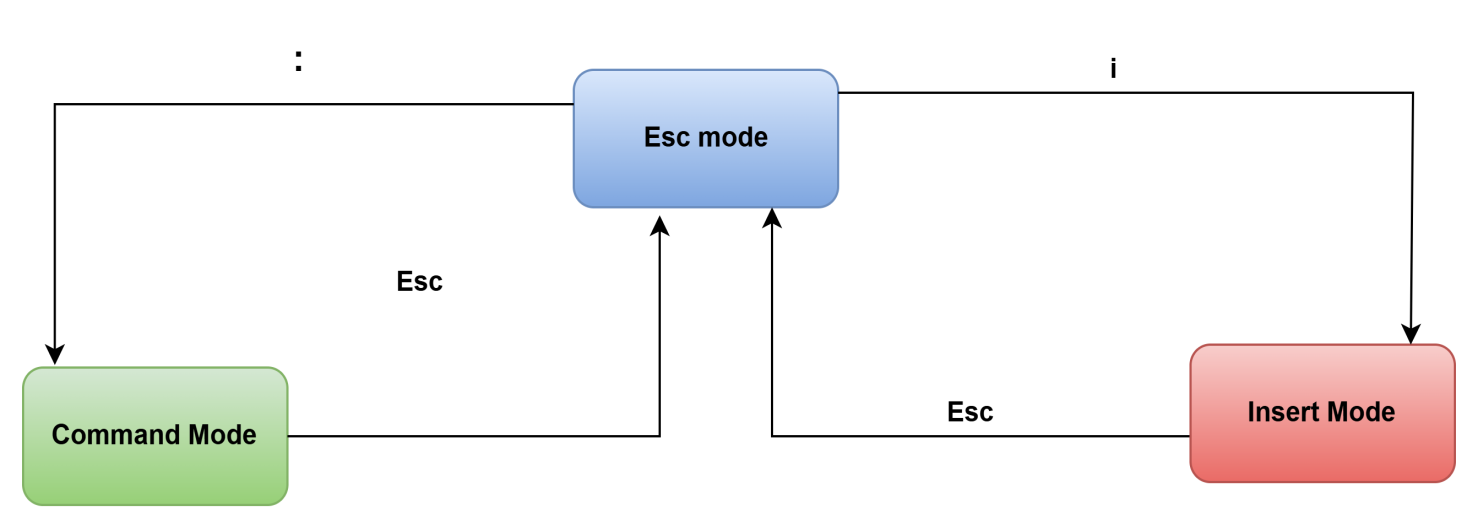
ssh -i daws-84s [ec2-user@I.p](mailto:ec2-user@I.p)

ls -l = long list

vim <file-name>

3 MODEs

Esc, Colon, insert MODEs



command MODE

==============

:q -> quit the file

:wq --> write and quit

:wq! --> force write and quit

:q! --> exit without saving

cp /etc/passwd users --> we have user information

:/<word-search> --> search for the word from top

:?<word-search> --> search for the word from bottom

:/ - forword search

?/ - backword search

:noh --> no highlight

:set nu --> set numbers

:set nonu --> set no numbers

:27 d --> delete 27th line

:%d --> total content will be deleted

:3s/sbin/SBIN --> in 3rd line first occurence of sbin will be replaced with SBIN

:3s/sbin/SBIN/g --> all occurence in that line

:%s/sbin/SBIN/g --> all occurences in the file

Esc Mode

===============

u --> undo

yy --> copy the line where you are

p --> paste

dd --> cut the line

gg --> takes to top of the file

shift+g --> takes us to bottom of the file

Vim commands (if you need any vim commands check in google)

Linux Administration

=====================

A Linux administrator is a technical professional who manages and maintains Linux-based systems, including servers and desktops. Their primary responsibilities involve installation, configuration, security, and troubleshooting of Linux systems. They also handle tasks like user management, software updates, backups, and system monitoring.

User management

**User Account Management**

| **Command** | **Description** |
| --- | --- |
| useradd <username> | Adds a new user. |
| adduser <username> | More user-friendly wrapper around useradd (depends on distro). |
| userdel <username> | Deletes a user. |
| usermod <options> <username> | Modifies an existing user. |
| passwd <username> | Sets or changes the password for a user. |

### ****Group Management****

| **Command** | **Description** |
| --- | --- |
| groupadd <groupname> | Creates a new group. |
| groupdel <groupname> | Deletes a group. |
| groupmod <options> <groupname> | Modifies an existing group. |
| usermod -aG <group> <user> | Adds a user to a group. (-aG = append to group) |
| groups <username> | Lists groups a user belongs to. |

### ****Viewing User & Group Info****

| **Command** | **Description** |
| --- | --- |
| id <username> | Shows UID, GID, and groups of a user. |
| who | Shows who is logged in. |
| whoami | Shows the current logged-in username. |
| getent passwd | Lists all users from /etc/passwd. |
| cat /etc/passwd | View user account information. |
| cat /etc/group | View group information. |

**Other Useful User Management Operations**

| **Command** | **Description** |
| --- | --- |
| chage <username> | Manage password expiry info. |
| sudo | Run commands as another user (typically root). |
| su <user> | Switch to another user. |

CRUD --> **C**reate – **R**ead – **U**pdate – **D**elete

Sudo su - now I’m in administrator

useradd <user-name> --> create user

id <user-name> --> displays the user info

group --> list of similar users

devops team have 20 members

create devops group, add team members to the group

in linux when you create user, by default group also will be created with same name..

**primary group and secondary group**

### ****Primary Group****

1. The main group associated with a user.
2. It's assigned automatically when the user is created.
3. Used by default when the user creates new files (sets group ownership).
4. Typically has the same name as the username.

#### Example:

id shanthi# Output: uid=1001(shanthi) gid=1001(shanthi) groups=1001(shanthi)

Here, gid=1001(shanthi) is the **primary group**.

### ****Secondary Group(s)****

1. Additional groups the user belongs to.
2. Used to grant access to shared resources like files, directories, or devices.

#### Example with secondary group:

sudo usermod -aG developers shanthi

Now:

id shanthi# Output: uid=1001(shanthi) gid=1001(shanthi) groups=1001(shanthi),1002(developers)

Here, developers is a **secondary group**.

### Commands Summary:

| **Task** | **Command** |
| --- | --- |
| View user's groups | id <username> or groups <username> |
| Add to secondary group | sudo usermod -aG <group> <user> |
| Change primary group | sudo usermod -g <newgroup> <user> |

In linux every user should have one primary group and at least one secondary group

cat /etc/passwd - user information

cat /etc/group - group information

1 primary group, atleast one secondary group

groupadd devops --> devops group will be created

usermod -g devops suresh

usermod -aG testers suresh

Userdel suresh --> deleted user name suresh

cd /etc/group - group was not deleted, you should definitely delete the group too.

groupdel suresh

cd /home/suresh - switch to suresh directory

cd - changed the root directory

How to remove user from secondary group in linux

gpasswd -d suresh testers

gpasswd -d suresh devops

usermod -g suresh suresh

How was it created by default

Now also delete by default

The commands used may differ across projects, as they depend on the project's environment, platform, and configuration needs.

userdel suresh

cat /etc/group

I’m adding user again password must be set

useradd suresh - user created

passwd suresh - suresh@123

In linux,we access should be done with default keys only. Do not access with passwords. So that we need to make changes to the configuration.

vim /etc/ssh/sshd\_config

Now, I will open vim /etc/ssh/sshd\_config -- configuration file

In this file,check password - it will be somewhere

:?password - search this

Password authentication yes- made the config changes

sshd -t - for testing

Checks the syntax and validity of the SSH daemon (sshd) configuration file (usually /etc/ssh/sshd\_config), without starting or restarting the service.

**Purpose:**

1. Ensures there are no syntax errors in your SSH configuration before applying changes.
2. Prevents you from accidentally breaking SSH access (especially important on remote servers like AWS EC2).

Systemctl restart sshd

Restarts the SSH daemon (sshd) service using systemctl, which is the service manager on most modern Linux systems (like CentOS, Ubuntu, RHEL, Fedora).

When to Use:

After editing the SSH configuration file (/etc/ssh/sshd\_config)

To apply changes like:

1. Port number
2. Authentication methods
3. PermitRootLogin settings, etc.

Take new server

Give ssh suresh@pub ip

Password:suresh@123

In certain Dockerfiles, creating users is necessary as part of the image setup, depending on the project’s security or application requirements.

passwd <user-name>

**Class 5:**

**Today’s topics:**

1. **Permissions**
2. **ownership**
3. **Sudo access to user**
4. **Key based authentication**
5. **Package management**
6. **Service management**

permissions

==============

In Linux, **permissions** control who can **read**, **write**, or **execute** files and directories. They're essential for **security** and **user access management**.

### ****Types of Permissions:****

| **Permission** | **Symbol** | **Description** |
| --- | --- | --- |
| **Read** | r | View contents of a file or list files in a directory. |
| **Write** | w | Modify a file or create/delete files in a directory. |
| **Execute** | x | Run a file as a program or enter a directory. |

### ****Permission Categories:****

| **Category** | **Applies To** |
| --- | --- |
| **Owner (user)** | The user who owns the file. |
| **Group** | Users who are part of the file’s group. |
| **Others** | All other users. |

### ****Example Output (from**** ls -l****):****

-rwxr-xr-- 1 alice devs 1042 May 3 10:00 script.sh

-rwxr-xr-- → permissions:

Owner (alice): rwx = read, write, execute

Group (devs): r-x = read, execute

Others: r-- = read-only

### ****Managing Permissions:****

| **Task** | **Command** |
| --- | --- |
| Change permissions | chmod |
| Change owner | chown |
| Change group | chgrp |

#### Examples:

chmod 755 file.sh # rwxr-xr-xchown alice:devs file.sh

R -> 4

W -> 2

X -> 1

* rw- r-- r--

file/ user/ group others

owner

directory u g o

ec2-user ec2-user

user group

1 useradd shanthi

2 sudo rm -f /etc/passwd.lock

3 useradd shanthi

4 sudo whoami

5 sudo su -

6 sudo su-

7 sudo su -

8 id

9 id username

10 sudo

11 ls -l'

12 ls -l

13 touch devops.txt

14 chmod ugo+w devops.txt

15 ls -l

16 chmod o-w devops.txt

17 ls -l

18 chmod ugo+rwx devops.txt

19 ls -l

20 chmod 700 devops.txt

21 ls -l

22 chmod 754 devops.txt

23 ls -l

24 cd /tmp

25 ls -l

26 touch aws.txt

27 ls -l

28 sudo useradd shanthi

29 chmod shanthi aws.txt

30 sudo chmod shanthi aws.txt

31 sudo chown shanthi aws.txt

32 ls -l

33 sudo chown shanthi:shanthi aws.txt

34 ls -l

35 cd

36 ls -la

37 cd .ssh

38 ls -l

39 cat authorized\_keys

40 systemctl status sshd

chmod ugo+w devops.txt

chmod 700 devops.txt

only owner/root user can change the permissions

chown <user>:<group> devops.txt

file ownership can only be modified by root user

/etc/ssh/sshd\_config

how can you give key based access to linux user?

ssh-keygen -f sivakumar

65,536 ports 0-65,535

1. create user

2. sivakumar can send his public key to admin user

3. /home/sivakumar admin creates .ssh in /home/sivakumar folder

4. sivakumar is the only owner to this folder... 700

5. create a file called authorized\_keys with max access 600

6. admin keeps sivakumar public key here.

7. now sivakumar should be able to login

port is like flat number

computer system using the **TCP/IP protocol** supports **65,536 ports**, numbered from **0 to 65,535**.

| **Port Range** | **Purpose** |
| --- | --- |
| **0 – 1023** | **Well-known ports** (reserved for system services like HTTP, SSH, DNS, etc.) |
| **1024 – 49151** | **Registered ports** (used by user or vendor software) |
| **49152 – 65535** | **Dynamic / Private ports** (used for temporary communication, e.g., ephemeral ports) |

ssh -i <key> ec2-user@IP

1 cd /tmp

2 ls -l

3 touch aws.txt

4 ls -l

5 sudo useradd shanthi

6 sudo chown aws.txt

7 sudo chown shanthi aws.txt

8 sudo chown shanthi:shanthi aws.txt

9 cd

10 ls -la

11 cd .ssh

12 ls -l

13 cat authorized\_keys

14 systemctl status sshd

15 sudo su -

16 cd

17 ls -lrt

18 ls

19 whoami

20 cd

21 sudo su

22 sudo su

23 cd

24 sudo

25 sudo su -

26 sudo su -

27 useraddkranthikumar

28 useradd sivakumar

29 cd ../sivakumar/

30 whoami

31 sudo su

32 ls -lrt

33 useradd sivakumar

34 cd ../sivakumar/

35 cd ../ssh

36 cd

37 ls -la

38 cd .ssh

39 ls -la

40 cat authorized\_keys

41 ls -l

42 cd

43 sudo su

44 useradd sivakumar

45 cd ../sivakumar

46 cd /sivakumar

47 ls -lrt

48 cd ../sivakumar/

49 is sivakumar

50 id sivakumar

51 cd ../

52 ls -lrt

53 cd

54 ls -lrt

55 sudo su

56 cd

57 sudo su

58 useradd sivakumar

59 cd ../sivakumar/

60 pwd

61 mkdir .ssh

62 chmod 700 -R .ssh/

63 ls -l

64 ls -la

65 cd .ssh/

66 touch authorized\_keys

67 ls -l

68 cat authorized\_keys

69 cat authorized\_keys

70 vim authorized\_keys

71 cat authorized\_keys

72 chmod 600 authorized\_keys

73 ls -l

74 cd ../

75 ls -la

76 chown sivakumar:sivakumar -R .ssh/

77 ls -la

78 cd .ssh/

79 ls -la

80 cd

81 vim /etc/sudoers

82 usermod -aG wheel sivakumar

83 id sivakumar

84 cd

85 Sudo su

86 vim /etc/sudoers

/etc/sudoers --> you can do changes to provide root access

or add user to wheel group without password

**package management**

====================

package manager connects to internet windows website. downloads them and install them

apt-get

yum/dnf

1. Whoami - root user
2. dnf install git
3. dnf remove git -f
4. dnf remove git -y
5. dnf list installed
6. dnf list available
7. dnf list available | grep python

dnf install <package-name>

/etc/yum.repos.d

service management

===================

Service management in Linux involves controlling background processes, known as services or daemons, which are typically managed using **init systems** such as **Systemd**, **SysVinit**, or **Upstart** (with **Systemd** being the most common on modern distributions).

systemctl start <service-name>

systemctl stop <service-name>

systemctl status <service-name>

systemctl restart <service-name>

systemctl enable <service-name>

systemctl disable <service-name>

install nginx and start it

protocol, port, ip, username and password

http 80 <IP>

1. sudo su -
2. dnf install nginx -y
3. systemctl start nginx
4. \systemctl status nginx
5. dnf install nginx -y
6. systemctl start nginx
7. systemctl status nginx
8. systemctl stop nginx
9. systemctl status nginx
10. systemctl stop nginx
11. systemctl status nginx
12. systemctl stop nginx
13. systemctl restart nginx
14. systemctl enable nginx
15. systemctl disable nginx
16. systemctl start nginx
17. **Which is utility is used to create public and private key pairs for ssh authentication?**

ssh-keygen

1. **Where is the private key stored by default?**

~/.ssh/id\_rsa

1. **When using vi or vim to edit a file,how do you exit without saving?**

:q!

1. **How to delete a whole line in vim?**

dd

1. **Which command-line tool is commonly used for downloading files in lunux?**

wget

1. **How can you display the first few lines of a file in linux?**

head

1. **Which port is normally required to be open for secure remote interactive shell access to linux system?**

22

1. **Which protocol is commonly used for secure remote access to servers?**

ssh

1. **What is the home folder for a user joindevops?**

/home/joindevops