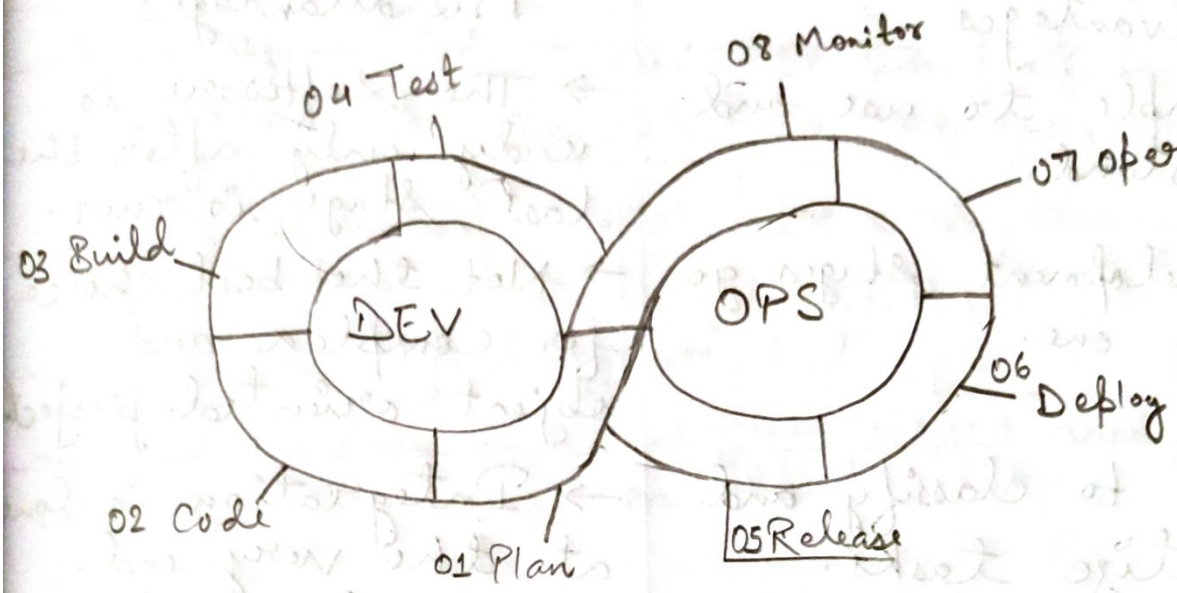
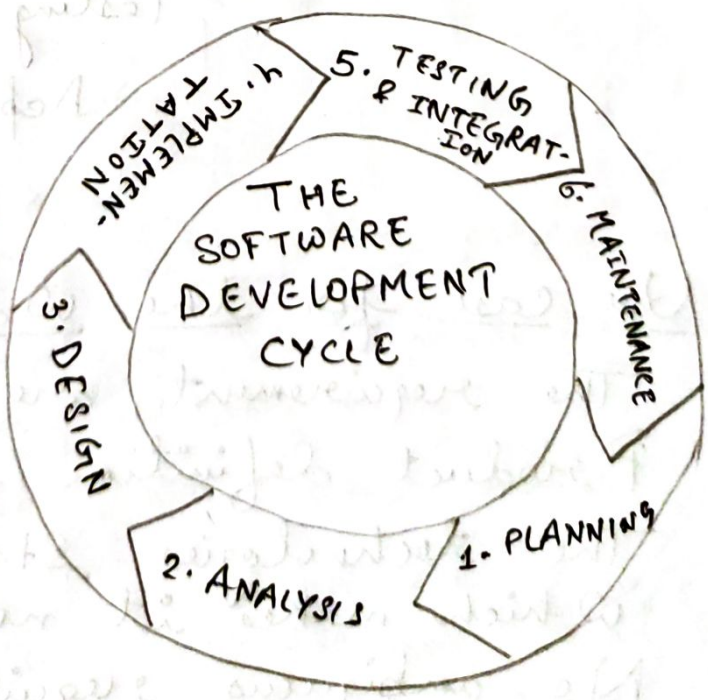
Ch:- 1 Introduction to DevOps



* **DevOps** :- DevOps is short for Development and Operations. It Concentrates on collaboration between developers and other parties, involved in building, deploying, operating and maintaining Software Systems.

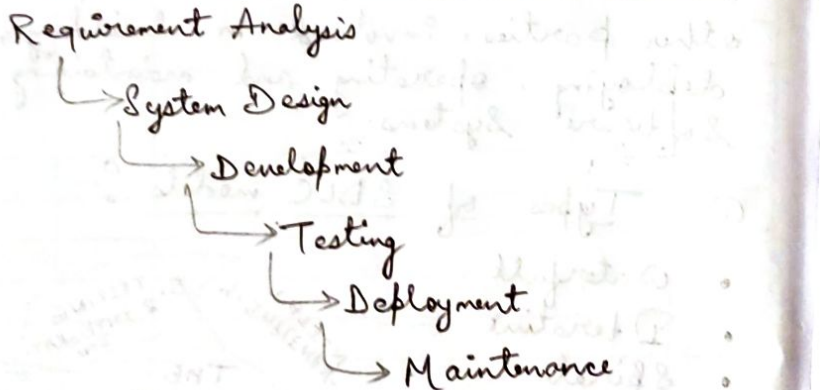
① **Types of SDLC models** :-

- Waterfall
- Iterative
- Spiral
- V-Shaped
- Agile



Waterfall model

Advantages	Disadvantages
→ Simple to use and understand.	→ The software is ready only after the last stage is over.
→ Development stages go one by one.	→ Not the best choice for complex and object oriented projects.
→ Easy to classify and prioritize tasks.	→ Integration is done at the very end, which does not give the option of identifying the problem in advance.

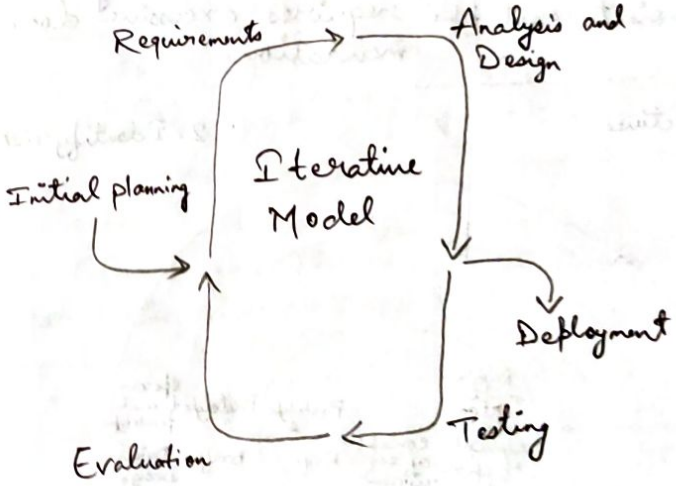


Use case for the Waterfall SDLC model:

- The requirement are precisely documented.
- Product definition is stable.
- The technologies stack is predefined, which makes it not dynamic.
- No ambiguous requirements.
- The project is short.

Iterative Model

Advantages	Disadvantages
→ The parallelled development can be applied	→ Constant management is required
→ The shorter iteration is — the easier testing and debugging stages are.	→ Bad choice for the small projects.
→ Flexibility and readiness to the changes in the requirements.	→ Risks analysis requires involvement of the highly qualified specialists.

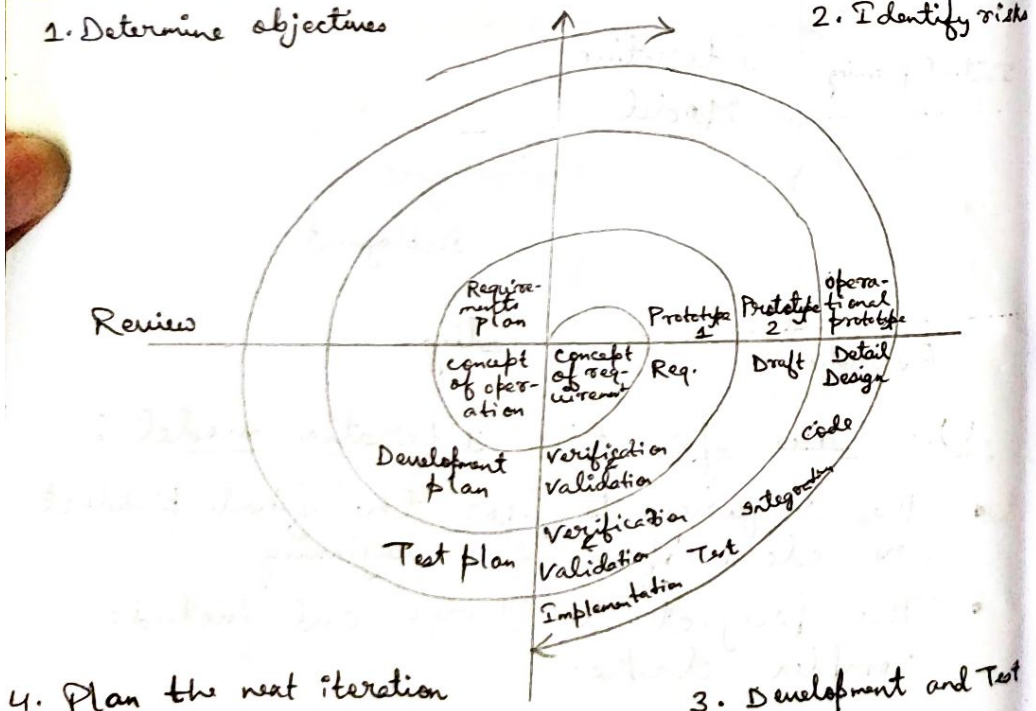


Use Cases for the Iteration model:

- The requirements for the final product are clear from the bginning.
- The project is large and include complex tasks.
- The main tasks is predefined, but the details may change in the process.

Spiral Model

Advantages	Disadvantages
→ The development process is precisely documented.	→ The risk control demands involvement of the highly skilled professionals.
→ The Scalability allows to make changes and add new functionality even at the relatively late stages.	→ Can be ineffective for the small project.
→ The earlier working prototype is done - sooner users can point out the flaws.	→ Big number of the intermediate stages requires excessive documentation

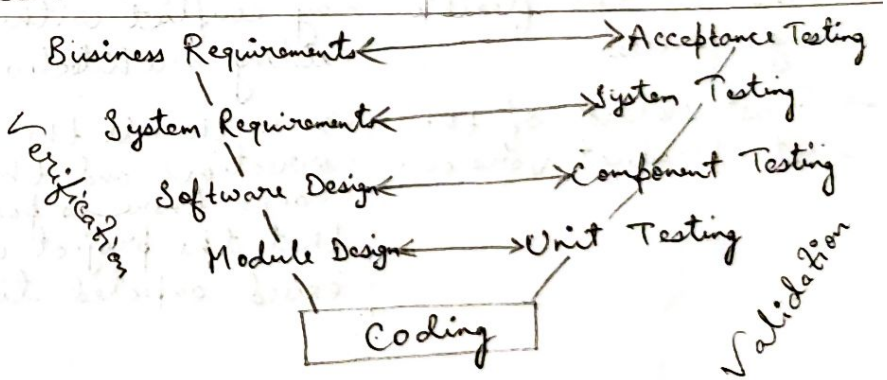


Use cases for the Spiral model

- The Customer isn't sure about the requirements.
- Significant edits are expected during the Software development life cycle.
- Risk management is highly essential for the project.

V-Shaped Model

Advantages	Disadvantages
→ Every Stage of V-Shaped model has strict results, so its easy to control.	→ Lack of the flexibility
→ Testing and verification take place in the early stages.	→ Bad choice for the small projects.
→ Good for the small projects, where requirements are static and clear.	→ Relatively big risks



Use cases for the V-shaped model:

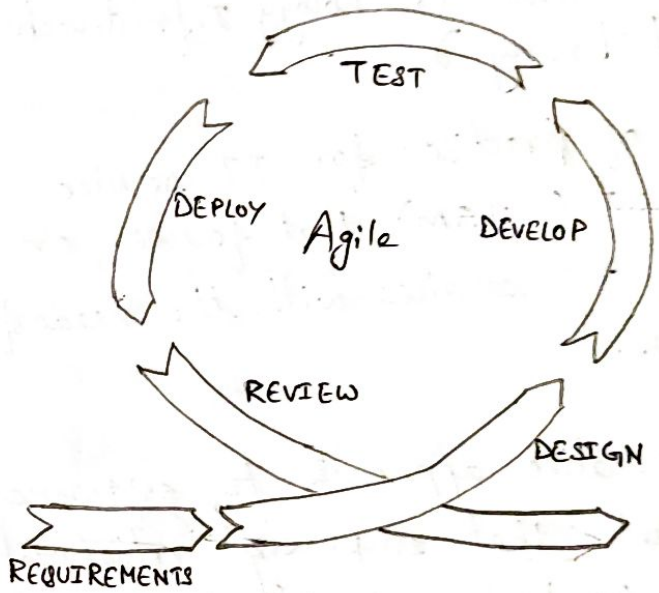
- For the projects where accurate product testing is required.
- For the small and mid-sized projects, where requirements are strictly predefined.
- The engineers of the required qualifications, especially testers, are within easy reach.

Agile Model

Advantages	Disadvantages
→ Corrections of functional requirements are implemented into the development process to provide the competitiveness.	→ Difficulties with measuring the final cost because of permanent changes.
→ Project is divided by short and transparent iterations.	→ The team should be highly professional and client oriented.
→ Risks are minimized thanks to the flexible change process.	→ New requirements may conflict with the existing architecture.
→ Fast release of the first product version.	→ With all the corrections and the changes there is possibility that the project will exceed expected time.

Use cases for the Agile model:

- The users need change dynamically.
- Less price for the changes implemented because of the numerous iterations.
- It requires only initial planning to start the project.



21/03/024

Lean: Focuses on minimizing waste and maximizing value in the software development process.

ITIL: (Information Technology Infrastructure Library)

⇒ A set of practices for IT service management (ITSM) that focuses on aligning IT services with the needs of business.

Agile: An iterative approach to software development that emphasizes collaboration, customer feedback and the ability to respond to change.

Q. Why DevOps ?

DevOps aims to improve the collaboration b/w Development and Operations teams, automate manual processes, increase the speed of software delivery and ensure reliability and quality of software releases.

History of DevOps :-

1. **Origin :**

→ DevOps emerged around 2007 - 2008 in response to concerns raised by the software development and IT operations communities.

2. **Silos and dysfunction :**

→ Traditionally, Development and Operations team worked in separate Silos ( ).

→ Developers focused on writing code, while operations team handled deployment and support.

→ These separation led to competing objectives, disjointed efforts and different metrics for success.

→ Physical separation, such as working on different floors or in separate buildings, further worsen the issue.

3. **Call for change :**

→ Frustrated by failed releases, long hours, customer dissatisfaction, both communities realised a need for better approach.

→ They began to question if there was a way to bridge the gap b/w development & operations.

4. The DevOps Conversation :-

→ Visionaries like ~~Patrick~~ Patrick Debois, Gene Kim & John willis started discussing these challenges.

→ Initially, these conversations took place in online forums and local meetups.

→ Eventually, the DevOps movement gained momentum and became a major theme in the software industry.

DevOps Stakeholders

Dev includes all people involved in developing software products and services.

- product owner
- customers
- ~~Dest~~ Designer
- ~~Product manager~~ Kuchu
- Project manager
- QA professional
- Testers & analyst

developer

Ops (operations) : Includes all people involved in delivering and managing software products and services, such as -

- System engineer
- DB administrators
- Network Engineers
- IT Operations Engineers.
- System administrator
- Support Professionals.

DevOps Goals

1. Faster delivery :-

→ The primary goal of DevOps is to increase the speed of software delivery without compromising quality.

→ This is achieved through automation, collaboration and CI/CD practices.

2. Scalability :-

→ It focuses on making the software delivery process more scalable.

→ This involves designing systems that can easily scale up or down based on requirements

3. Improved Collaboration :-

DevOps promotes collaboration between development, operations and other stakeholders involved in the software delivery process.

4. Increased Efficiency :-

By automating repetative tasks and streamlining processes, DevOps aims to increase efficiency in the software development lifecycle.

5. Enhanced Security :-

DevOps emphasizes the importance of security throughout the software delivery process. By integrating security practices early in the development lifecycle and automating security checks, DevOps aims to create more secure software products.

6. provides faster feedback.

This includes gathering feedback from users, monitoring sys performance & analysing metrics to identify areas for improvement.

Important terminologies :

- artifact
- Build agent
- commit
- configuration mgmt.
- Containerization
- CI/CD
- Continuous Delivery
- Continuous Testing
- Deployment Pipeline
- IAS
- Microservices
- Pair programming
- Production
- Staging
- rollback
- Rolling Update
- source control
- Test automation
- Virtual Machine
- Bastion host
- Backup
- Build
- Bare-metal
- cluster
- cron job
- IAC ode (Infrastructure as Code)
- Orchestration
- Source Code mgmt
- Software configuration mgmt.

22.03.2024

DevOps Perspective

DevOps is an IT mindset that encourages communication, collaboration, integration and automation among Software Developers and IT operations in order to improve the speed and quality of delivering Software.

* DevOps and Agile

* Satisfy customer through early and continuous delivery of valuable Software.
* Deliver working Software frequently with a preference for the shorter time-scale.
* Business people and ~~dev~~ Developers must work together daily throughout the project.
* Replace non-human steps using tools.
* Improve the collaboration between all teams.
* Automate to create a potentially shippable increment.

* ~~DevOps~~ Tools in various Stages of DevOps

(1) Plan :- No tools given.
(2) Code :- JIRA, git
(3) Build :- Maven
(4) Test :- Selenium, JUnit
(5) Release :- Jenkins
(6) Deploy :- Docker, AWS
(7) Operate :- Chef, Ansible, Kubernetes
(8) Monitor :- DataDog, Splunk, Nagios.

* Configuration management

A process of establishing and maintaining consistent settings of a System.
These solutions also include SysAdmin tools for IT infrastructure automation (e.g. chef, Puppet, etc.)

* CONTINUOUS INTEGRATION ~~AND DEPLOYMENT~~

A Software development process where a branch of source code is rebuilt every time code is committed to the source control system. The process is often extended to include, deployment, installation and testing of applications in production environments.

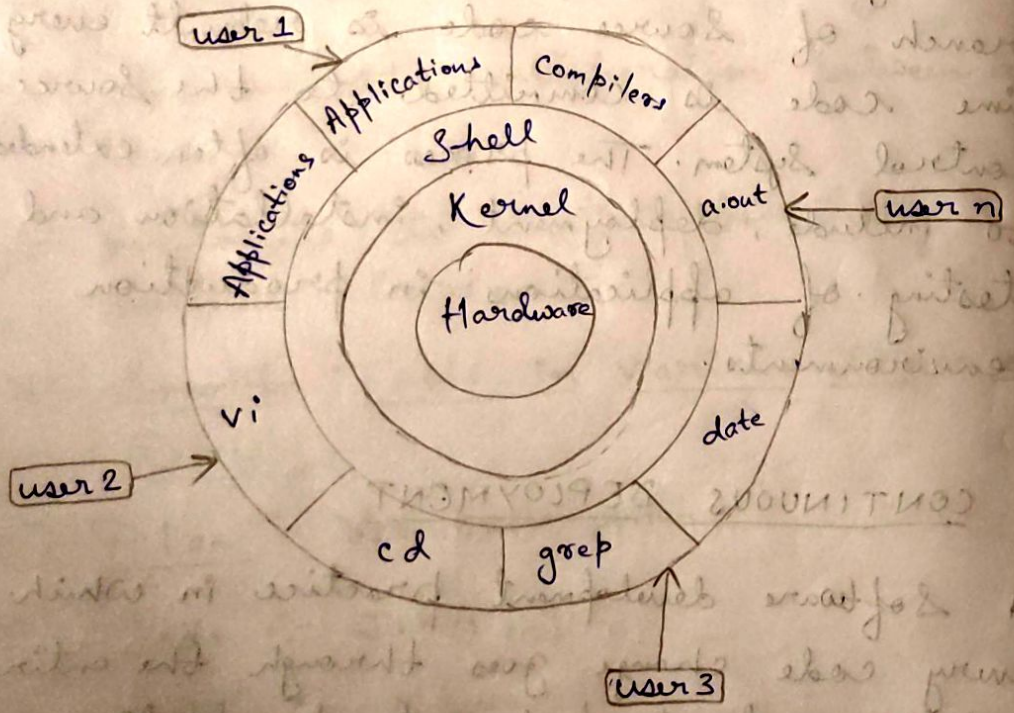
* CONTINUOUS DEPLOYMENT

A Software development practice in which every code change goes through the entire pipeline and is put into production automatically, resulting in many production deployments every day. It does everything that continuous delivery does, but the process is fully automated, and there's no human intervention at all.

LINUX OS INTRODUCTION

⇒ Linux is one of the most popular OS.
⇒ It is open-source as its source code is freely available.
⇒ It is UNIX like OS based on the Linux Kernel, first released in 1991 by Linus Torwalds

ARCHITECTURE



The architecture of a Linux System Consist of the following layers :

① Hardware Layer :- It consists of all peripheral devices (RAM, HDD, CPU, etc).

② Kernel :- It is the core component of Operating System, interacts directly with hardware, provides low level services to upper layer components.

③ Shell :- An Interface to Kernel, hiding complexity of Kernel's functions from users. The Shell takes commands from the user and executes Kernel's functions.

④ Utilities :- Utility program that provide the user most of the functionalities of an OS.
e.g., antivirus Software, file explorer, backup Software, etc.

23/03/024

~~Importants~~

Importance of Linux in DevOps :-

*

→ Linux offers the DevOps team the ~~flexibility~~ flexibility and scalability needed to create a dynamic development process.

→ Majority of enterprises are today running development projects which already have Linux supporting their operations.

→ Linux is used for tasks like managing servers, configuring network settings, deploying and monitoring apps and automating tasks through scripting.

→ Learning and mastering Linux can significantly enhance a DevOps Engineer's ability to manage infrastructure, automate processes & deploy apps efficiently.

Linux Basic Commands

*

① sudo → execute a command with superuser privileges

② cp ⟹ copy files and directories. (cp source destination)

③ free ⇒ display memory usage

④ uptime ⇒ show how long the system is running

⑤ kill ⇒ terminate process.

⑥ top ⇒ display real time system sy summary and process list.

⑦ w ⇒ show who is logged on & what they are doing.

⑧ mv ⇒ move or rename files or directories. (mv source destination)

⑨ tar ⇒ archive files.

⑩ users ⇒ display users currently logged in.

⑪ cat ⟹ concatenate and display file contents.

⑫ grep ⇒ search for patterns within files.

⑬ who ⇒ show who is logged ~~in~~ on.

⑭ cd ⇒ change directory.

⑮ pwd ⇒ print working directory.

⑯ whoami ⇒ display current user.

⑰ tail ⇒ display end of a file (tail filename.ext)

⑱ sort ⇒ sort lines of text files.

⑲ ls ⇒ list directory contents.

⑳ head ⇒ display the beginning of a file.

(21) ssh ⇒ securely connect to a remote machine.

(22) crontab ⇒ schedule commands to run periodically.

(23) lsof ⇒ list open files.

(24) last ⇒ show listing of last logged in users.

(25) mkdir ⇒ create directories

(26) ps ⇒ display current processes.

(27) netstat ⇒ network statistics.

Linux Administration

① Service management.

Overseeing DNS, Apache, MySQL, PHP services

② Data Management

regular backups of ~~db~~ and dB maintenance

③ Troubleshooting

analysing error logs and resolving issues for web hosting and n/w services.

~~④ Communication~~

④ Tool Development

creating & improving Linux tools for users.

⑤ Problem Solving

service issues like disaster recovery, login problems, etc.

⑥ System Security

Installing security systems & collaborating on h/w needs.

⑦ Server maintenance

conducting server troubleshooting as needed.

Environment Variable

खुद से लिखो

Networking in Linux

It involves configuring and managing n/w connections and services on Linux based systems. It is a critical skill for system administrators and includes tasks such as —

1) Configuring network interfaces: ~~ifc~~ ~~ifg~~ using
   → tools like ifconfig, ip, Network Manager. ~~etc~~
2) Managing network services
   → services like DNS & DHCP.
3) Implementing security measures :-
   → using Firewalls, VPNs & other security tools to protect the network.
4) Monitoring and Troubleshooting:
   → utilizing commands like netstat, ping, etc. to monitor network performance.

Commands related to Networking

⇒ ip : shows/manipulates routing, devices, policy routing and tunnels.

⇒ ping : sends ICMP (Internet control message protocol) ~~Ech~~ ECHO_REQUEST packets to network host.

⇒ netstat : shows statistics related to network like ~~interne~~ routing tables, network connections, etc.

⇒ url : Transfer data to or from server.

⇒ route : shows/manipulates the IP routing table.

⇒ nslookup : Queries Internet Domain Name Servers.

⇒ hostname : sets or displays the system's host name.

⇒ ifconfig ⇒ deprecated in many modern Linux distros.