DevOps Terms You Should Know

1) Devops :

Defination : A combination of "Development" and "operation", Devops is a set of practises that automate and integrate the process between Software Development and IT ~~terms~~ teams.

Why it matters : It helps teams build test and release software faster and more reliably.

2) CICD (Continous Integration / Continous Delivery / Continous Deployment)

Defination :

Continous Integration (CI) : Developers regularly merge code changes into shared repository, where the code is automatically tested.

continous Delivery (CD) : After code is tested, it is automatically prepared for release to production.

Continous Deployment (CD): The code is automatically deployed to production without manual intervention.

Why it matters : CI/CD automates and speeds up the software release process, ensuring quick delivery with fewer bugs.

3. Version Control:

Defination : A system that tracks and manages changes to code over time (eg. Git).

Why it matters: It allows developer collaborate on projects, see code history and revert to pervious version if needed.

4. Infrastructure as code (IaC)

Defination: Managing and provisioning IT infrastructure (like servers, networks) using code, rather than manual process.

Why it matters: It makes infrastructure easier to manage, scale and reproduce.

5. Containers

Defination: A lightweight, portable way to package software and its dependencies (eg. Docker) so that it runs consistently accross different environments.

Why it matters: Containers ensure that an application works the same way in development, testing and production environment.

6. Docker

Defination: A tool that allow you to create, deploy and run application in containers.

Why it matters: Docker simplifies deploying application in different environments by packaging them with everything they need to run.

7. Kubernetes

Defination: An open-source platform that automates the deployment, scaling and management of containerized application.

Why it matters: Kubernetes helps manage complex applications by orchestrating how containers run and interact with each other.

8. Microservices

Defination: A way of designing software where application is broken into smaller independent services that can be developed and scaled individually.

Why it matters: Microservices make it easier to scale and maintain large applications.

9. Orchestration

Definition: Automating the arrangement, coordination and management of computer systems, middleware and services (eg. kubernetes orchestrates Container)

Why it matters: It helps manage complex applications made moving parts, such as container and services.

Continous Deployment (CD): The code is automatically deployed to production without manual intervention.

Why it matters : CI/CD automates and speeds up the software release process, ensuring quick delivery with fewer bugs.

3. Version Control:

Defination : A system that tracks and manages changes to code over time (eg. Git).

Why it matters: It allows developer collaborate on projects, see code history and revert to pervious version if needed.

4. Infrastructure as code (IaC)

Defination: Managing and provisioning IT infrastructure (like servers, networks) using code, rather than manual process.

Why it matters: It makes infrastructure easier to manage, scale and reproduce.

13. Cloud Computing

Definition : Delivering computing services like servers, storage, databases and software over the internet. (eg. AWS, Google cloud, Azure).

Why its matters : The cloud offers flexibility and scalability, enabling teams to deploy applications globally without managing physical server.

14. Agile

Definition : A project management approach that emphasizes flexibility, collaboration, and customer feedback.

Why it matters : Agile speeds up the development process by delivering software in small frequent updates.

15. SRE (Site Reliability Engineer)

Definition: A discipline that applies software engineering practices to IT operation to ensure reliability and uptime.

Why it matters: SRE helps maintain highly reliable and scale systems.

16. Load Balancing

Definition: Distributing incoming network traffic across multiple servers to ensure no single server get overwhelmed.

Why it matters: Load Balancing ensure high availability and ~~smoth~~ smooth performance for applications.

17. Artifact

Definition: Any file or package that is created as part of the software development process (eg. binaries, libraries).

Why it matters: Artifacts are used in different stages of the CI/CD pipeline and help ensure consistency.

18. Rollback

Definition: Reverting an application to a previous version if something goes wrong during a deployment.

Why its matters: Rollbacks help maitain system stability by quickly undoing problematic changes.

19. Git

Definition : A version control system that tracks changes to files and allows multiple developers to work on the same project.

Why it matters : Git enables collaboration, code history tracking and rollback to pervious version.

20. Jenkins

Definition : An open-source automation tool used to build test and deploy software automatically (a CI/CD tool).

Why it matters : Jenkins helps automate the software development process, making it faster and more efficient.

By understanding this terms, you will have a strong foundation to navigate Devops concepts and tools.

— Faizan Hasan Shaikh.