# #90DaysOfDevOps Challenge

## Learn, Upskill, Grow with the Community

This repository is a Challenge for the DevOps Community to get stronger in DevOps. This challenge starts on the 1st January 2023 and in the next 90 Days we promise ourselves to become better at DevOps.

The reason for making this Public is so that others can learn from the community and help each other grow.

## Steps:

* Fork[https://github.com/LondheShubham153/90DaysOfDevOps/fork] the Repo.
* Learn Everyday and add your learnings in the day wise folders.
* Check out what others are Learning and help/learn from them.
* Showcase your learnings on LinkedIn

These are our community Links.

* Telegram Channel: https://t.me/trainwithshubham
* Discord Channel: https://discord.gg/hs3Pmc5F
* WhatsApp Group: https://chat.whatsapp.com/FvRlAAZVxUhCUSZ0Y1s7KY
* YouTube Channel: https://www.youtube.com/@TrainWithShubham
* Website: https://www.trainwithshubham.com/
* LinkedIn: https://www.linkedin.com/in/shubhamlondhe1996/

## Events

YouTube Live Announcement: https://youtu.be/rO5Rllir-LM

YouTube Playlist for DevOps: https://youtube.com/playlist?list=PLlfy9GnSVerRqYJgVYO0UiExj5byjrW8u

DevOps Course: https://bit.ly/devops-batch-2

# Introduction - Day 1

This is the day you have to Take this challenge and start your #90DaysOfDevOps with the #TrainWithShubham Community

* Fork this Repo.
* Start with a DevOps Roadmap[https://youtu.be/iOE9NTAG35g]
* Write a LinkedIn post or a small article about your understanding of DevOps
* What is DevOps
* What is Automation, Scaling, Infrastructure
* Why DevOps is Important, etc

## Basic linux commands

### Listing commands

ls option\_flag arguments–> list the sub directories and files avaiable in the present directory

Examples:

* ls -l–> list the files and directories in long list format with extra information
* ls -a–> list all including hidden files and directory
* ls \*.sh –> list all the files having .sh extension.
* ls -i –> list the files and directories with index numbers inodes
* ls -d \*/ –> list only directories.(we can also specify a pattern)

### Directoy commands

* pwd –> print work directory. Gives the present working directory.
* cd path\_to\_directory –> change directory to the provided path
* cd ~ or just cd –> change directory to the home directory
* cd - –> Go to the last working directory.
* cd .. –> change directory to one step back.
* cd ../.. –> Change directory to 2 levels back.
* mkdir directoryName –> to make a directory in a specific location

Examples:

mkdir newFolder # make a new folder 'newFolder'  
  
mkdir .NewFolder # make a hidden directory (also . before a file to make it hidden)  
  
mkdir A B C D #make multiple directories at the same time  
  
mkdir /home/user/Mydirectory # make a new folder in a specific location  
  
mkdir -p A/B/C/D # make a nested directory

# Day 2 Task: Basics linux command

Task: What is the linux command to 1. Check your present working directory. 2. List all the files or directories including hidden files. 3. Create a nested directory A/B/C/D/E

Note: [Check this file for reference](basic_linux_commands.md)

Check the basic\_linux\_commands.md file on the same directory day2

# Day 3 Task: Basic Linux Commands

Task: What is the linux command to

1. To view what’s written in a file.
2. To change the access permissions of files.
3. To check which commands you have run till now.
4. To remove a directory/ Folder.
5. To create a fruits.txt file and to view the content.
6. Add content in devops.txt (One in each line) - Apple, Mango, Banana, Cherry, Kiwi, Orange, Guava.
7. To Show only top three fruits from the file.
8. To Show only bottom three fruits from the file.
9. To create another file Colors.txt and to view the content.
10. Add content in Colors.txt (One in each line) - Red, Pink, White, Black, Blue, Orange, Purple, Grey.
11. To find the difference between fruits.txt and Colors.txt file.

Reference: https://www.linkedin.com/pulse/linux-commands-devops-used-day-to-day-activit-chetan-/

# Day 4 Task: Basic Linux Shell Scripting for DevOps Engineers.

## What is Kernel

The kernel is a computer program that is the core of a computer’s operating system, with complete control over everything in the system.

## What is Shell

A shell is special user program which provide an interface to user to use operating system services. Shell accept human readable commands from user and convert them into something which kernel can understand. It is a command language interpreter that execute commands read from input devices such as keyboards or from files. The shell gets started when the user logs in or start the terminal.

## What is Linux Shell Scripting?

A shell script is a computer program designed to be run by a linux shell, a command-line interpreter. The various dialects of shell scripts are considered to be scripting languages. Typical operations performed by shell scripts include file manipulation, program execution, and printing text.

**Tasks**

* Explain in your own words and examples, what is Shell Scripting for DevOps.
* What is #!/bin/bash? can we write #!/bin/sh as well?
* Write a Shell Script which prints I will complete #90DaysOofDevOps challenge
* Write a Shell Script to take user input, input from arguments and print the variables.
* Write an Example of If else in Shell Scripting by comparing 2 numbers

Was it difficult?

* Post about it on LinkedIn and Let me know :)

Article Reference: [Click here to read basic Linux Shell Scripting](https://devopscube.com/linux-shell-scripting-for-devops/)

YouTube Vedio: [EASIEST Shell Scripting Tutorial for DevOps Engineers](https://www.youtube.com/watch?v=_-D6gkRj7xc&list=PLlfy9GnSVerQr-Se9JRE_tZJk3OUoHCkh&index=3)

**Day 5 Task: Advanced Linux Shell Scripting for DevOps Engineers with User management**

If you noticed that there are total 90 sub directories in the directory ‘2023’ of this repository. What did you think, how did I create 90 directories. Manually one by one or using a script, or a command ?

All 90 directories within seconds using a simple command.

mkdir day{1..90}

### Tasks

1. You have to do the same using Shell Script i.e using either Loops or command with start day and end day variables using arguments -

So Write a bash script createDirectories.sh that when the script is executed with three given arguments (one is directory name and second is start number of directories and third is the end number of directories ) it creates specified number of directories with a dynamic directory name.

Example 1: When the script is executed as

./createDirectories.sh day 1 90

then it creates 90 directories as day1 day2 day3 .... day90

Example 2: When the script is executed as

./createDirectories.sh Movie 20 50 then it creates 50 directories as Movie20 Movie21 Movie23 ...Movie50

Notes: You may need to use loops or commands (or both), based on your preference . [Check out this reference: https://www.geeksforgeeks.org/bash-scripting-for-loop/](https://www.geeksforgeeks.org/bash-scripting-for-loop/)

1. Create a Script to backup all your work done till now.

Backups are an important part of DevOps Engineers day to Day activities The video in References will help you to understand How a DevOps Engineer takes backups (it can feel a bit difficult but keep trying, Nothing is impossible.) Watch [this video](https://youtu.be/aolKiws4Joc)

In case of Doubts, post it in [Discord Channel for #90DaysOfDevOps](https://discord.gg/hs3Pmc5F)

1. Read About Cron and Crontab, to automate the backup Script

Cron is the system’s main scheduler for running jobs or tasks unattended. A command called crontab allows the user to submit, edit or delete entries to cron. A crontab file is a user file that holds the scheduling information.

Watch This video as a Reference to Task 2 and 3 <https://youtu.be/aolKiws4Joc>

1. Read about User Management and Let me know on Linkedin if you’re ready for Day 6.

A user is an entity, in a Linux operating system, that can manipulate files and perform several other operations. Each user is assigned an ID that is unique for each user in the operating system. In this post, we will learn about users and commands which are used to get information about the users. After installation of the operating system, the ID 0 is assigned to the root user and the IDs 1 to 999 (both inclusive) are assigned to the system users and hence the ids for local user begins from 1000 onwards.

1. Create 2 users and just display their Usernames

[Check out this reference: https://www.geeksforgeeks.org/user-management-in-linux/](https://www.geeksforgeeks.org/user-management-in-linux/)

Post your daily work on Linkedin and le [me](https://www.linkedin.com/in/shubhamlondhe1996/) know , writing an article is the best :)

# Day 6 Task: File Permissions and Access Control Lists

### Today is more on Reading, Learning and Implementing File permissions

The concept of Linux File permission and ownership is important in Linux. Here, we will be working on Linux permissions and ownership and will do tasks on both of them. Let us start with the Permissions.

1. Create a simple file and do ls -ltr to see the details of the files [refer to Notes](https://github.com/LondheShubham153/90DaysOfDevOps/tree/master/2023/day6/notes)

Each of the three permissions are assigned to three defined categories of users. The categories are: - owner — The owner of the file or application. - “chown” is used to change the ownership permission of a file or directory. - group — The group that owns the file or application. - “chgrp” is used to change the gropu permission of a file or directory. - others — All users with access to the system. (outised the users are in a group) - “chmod” is used to change the other users permissions of a file or directory.

As a task, change the user permissions of the file and note the changes after `ls -ltr`

1. Write an article about File Permissions based on your understanding from the notes.
2. Read about ACL and try out the commands getfacl and setfacl

In case of any doubts, post it on [Discord Community](https://discord.gg/hs3Pmc5F)

Happy Learning

# Day 7 Task: Understanding package manager and systemctl

### What is a package manager in Linux?

In simpler words, a package manager is a tool that allows users to install, remove, upgrade, configure and manage software packages on an operating system. The package manager can be a graphical application like a software center or a command line tool like apt-get or pacman.

You’ll often find me using the term ‘package’ in tutorials and articles, To understand package manager, you must understand what a package is.

### What is a package?

A package is usually referred to an application but it could be a GUI application, command line tool or a software library (required by other software programs). A package is essentially an archive file containing the binary executable, configuration file and sometimes information about the dependencies.

### Different kinds of package managers

Package Managers differ based on packaging system but same packaging system may have more than one package manager.

For example, RPM has Yum and DNF package managers. For DEB, you have apt-get, aptitude command line based package managers.

### Tasks

1. You have to install docker and jenkins in your system from your terminal using package managers
2. Write a small blog or article to install these tools using package managers on Ubuntu and CentOS

### systemctl and systemd

systemctl is used to examine and control the state of “systemd” system and service manager. systemd is system and service manager for Unix like operating systems(most of the distributions, not all).

### Tasks

1. check the status of docker service in your system (make sure you completed above tasks, else docker won’t be installed)
2. stop the service jenkins and post before and after screenshots
3. read about the commands systemctl vs service

eg. systemctl status docker vs service docker status

For Reference, read [this](https://www.howtogeek.com/devops/how-to-check-if-the-docker-daemon-or-a-container-is-running/#:~:text=Checking%20With%20Systemctl&text=Check%20what%27s%20displayed%20under%20%E2%80%9CActive,running%20sudo%20systemctl%20start%20docker%20.)

#### Post about this and bring your friends to this #90DaysOfDevOps challenge.

# Day 8 Task: Basic Git & GitHub for DevOps Engineers.

## What is Git?

Git is a version control system that allows you to track changes to files and coordinate work on those files among multiple people. It is commonly used for software development, but it can be used to track changes to any set of files.

With Git, you can keep a record of who made changes to what part of a file, and you can revert back to earlier versions of the file if needed. Git also makes it easy to collaborate with others, as you can share changes and merge the changes made by different people into a single version of a file.

## What is Github?

GitHub is a web-based platform that provides hosting for version control using Git. It is a subsidiary of Microsoft, and it offers all of the distributed version control and source code management (SCM) functionality of Git as well as adding its own features. GitHub is a very popular platform for developers to share and collaborate on projects, and it is also used for hosting open-source projects.

## What is Version Control? How many types of version controls we have?

Version control is a system that tracks changes to a file or set of files over time so that you can recall specific versions later. It allows you to revert files back to a previous state, revert the entire project back to a previous state, compare changes over time, see who last modified something that might be causing a problem, who introduced an issue and when, and more.

There are two main types of version control systems: centralized version control systems and distributed version control systems.

1. A centralized version control system (CVCS) uses a central server to store all the versions of a project’s files. Developers “check out” files from the central server, make changes, and then “check in” the updated files. Examples of CVCS include Subversion and Perforce.
2. A distributed version control system (DVCS) allows developers to “clone” an entire repository, including the entire version history of the project. This means that they have a complete local copy of the repository, including all branches and past versions. Developers can work independently and then later merge their changes back into the main repository. Examples of DVCS include Git, Mercurial, and Darcs.

## Why we use distributed version control over centralized version control?

1. Better collaboration: In a DVCS, every developer has a full copy of the repository, including the entire history of all changes. This makes it easier for developers to work together, as they don’t have to constantly communicate with a central server to commit their changes or to see the changes made by others.
2. Improved speed: Because developers have a local copy of the repository, they can commit their changes and perform other version control actions faster, as they don’t have to communicate with a central server.
3. Greater flexibility: With a DVCS, developers can work offline and commit their changes later when they do have an internet connection. They can also choose to share their changes with only a subset of the team, rather than pushing all of their changes to a central server.
4. Enhanced security: In a DVCS, the repository history is stored on multiple servers and computers, which makes it more resistant to data loss. If the central server in a CVCS goes down or the repository becomes corrupted, it can be difficult to recover the lost data.

Overall, the decentralized nature of a DVCS allows for greater collaboration, flexibility, and security, making it a popular choice for many teams.

## Task:

* Install Git on your computer (if it is not already installed). You can download it from the official website at https://git-scm.com/downloads
* Create a free account on GitHub (if you don’t already have one). You can sign up at https://github.com/
* Learn the basics of Git by reading through the [video](https://youtu.be/AT1uxOLsCdk) This will give you an understanding of what Git is, how it works, and how to use it to track changes to files.

## Exercises:

1. Create a new repository on GitHub and clone it to your local machine
2. Make some changes to a file in the repository and commit them to the repository using Git
3. Push the changes back to the repository on GitHub

Reff :- https://youtu.be/AT1uxOLsCdk

Post your daily work on Linkedin and le me know , writing an article is the best :) ;

# Day 9 Task: Deep Dive in Git & GitHub for DevOps Engineers.

## Find the answers by your understandings(Shoulden’t be copied by internet & used hand-made diagrams) of below quistions and Write blog on it.

1. What is Git and why is it important?
2. What is difference Between Main Branch and Master Branch??
3. Can you explain the difference between Git and GitHub?
4. How do you create a new repository on GitHub?
5. What is difference between local & remote repository? How to connect local to remote?

## Tasks

task-1: - Set your user name and email address, which will be associated with your commits.

task-2: - Create a repository named “Devops” on GitHub - Connect your local repository to the repository on GitHub. - Create a new file in Devops/Git/Day-02.txt & add some content to it - Push your local commits to the repository on GitHub

reff :- https://youtu.be/AT1uxOLsCdk

Note: These steps assume that you have already installed Git on your computer and have created a GitHub account. If you need help with these prerequisites, you can refer to the [day-08](https://github.com/LondheShubham153/90DaysOfDevOps/blob/ee7c53f276edb02a85a97282027028295be17c04/2023/day08/tasks.md)

# Day 10 Task: Advance Git & GitHub for DevOps Engineers.

## Git Branching

Use a branch to isolate development work without affecting other branches in the repository. Each repository has one default branch, and can have multiple other branches. You can merge a branch into another branch using a pull request.

Branches allow you to develop features, fix bugs, or safely experiment with new ideas in a contained area of your repository.

## Git Revert and Reset

Two commonly used tools that git users will encounter are those of git reset and git revert . The benefit of both of these commands is that you can use them to remove or edit changes you’ve made in the code in previous commits.

## Git Rebase and Merge

### What Is Git Rebase?

Git rebase is a command that lets users integrate changes from one branch to another, and the logs are modified once the action is complete. Git rebase was developed to overcome merging’s shortcomings, specifically regarding logs.

### What Is Git Merge?

Git merge is a command that allows developers to merge Git branches while the logs of commits on branches remain intact.

The merge wording can be confusing because we have two methods of merging branches, and one of those ways is actually called “merge,” even though both procedures do essentially the same thing.

Refer to this article for a better understanding of Git Rebase and Merge [Read here](https://www.simplilearn.com/git-rebase-vs-merge-article)

## Task 1:

Add a text file called version01.txt inside the Devops/Git/ with “This is first feature of our application” written inside. This should be in a branch coming from master, [hint try git checkout -b dev], swithch to dev branch ( Make sure your commit message will reflect as “Added new feature”). [Hint use your knowledge of creating branches and Git commit command]

* version01.txt should reflect at local repo first followed by Remote repo for review. [Hint use your knowledge of Git push and git pull commands here]

Add new commit in dev branch after adding below mentioned content in Devops/Git/version01.txt: While writing the file make sure you write these lines

* 1st line>> This is the bug fix in development branch
* Commit this with message “ Added feature2 in development branch”
* 2nd line>> This is gadbad code
* Commit this with message “ Added feature3 in development branch
* 3rd line>> This feature will gadbad everything from now.
* Commit with message “ Added feature4 in development branch

Restore the file to a previous version where the content should be “This is the bug fix in development branch” [Hint use git revert or reset according to your knowledge]

## Task 2:

* Demonstrate the concept of branches with 2 or more branches with screenshot.
* add some changes to dev branch and merge that branch in master
* as a practice try git rebase too, see what difference you get.

## Note:

We should learn and follow the [best practices](https://www.flagship.io/git-branching-strategies/) , industry follows for branching.

Simple Reference on branching: [video](https://youtu.be/NzjK9beT_CY)

Advance Reference on branching : [video](https://youtu.be/7xhkEQS3dXw)

You can Post on LinkedIn and let us know what you have learned from this task by #90DaysOfDevOps Challange. Happy Learning :)

# Day 11 Task: Advance Git & GitHub for DevOps Engineers: Part-2

## Git Stash:

Git stash is a command that allows you to temporarily save changes you have made in your working directory, without committing them. This is useful when you need to switch to a different branch to work on something else, but you don’t want to commit the changes you’ve made in your current branch yet.

To use Git stash, you first create a new branch and make some changes to it. Then you can use the command git stash to save those changes. This will remove the changes from your working directory and record them in a new stash. You can apply these changes later. git stash list command shows the list of stashed changes.

You can also use git stash drop to delete a stash and git stash clear to delete all the stashes.

## Cherry-pick:

Git cherry-pick is a command that allows you to select specific commits from one branch and apply them to another. This can be useful when you want to selectively apply changes that were made in one branch to another.

To use git cherry-pick, you first create two new branches and make some commits to them. Then you use git cherry-pick command to select the specific commits from one branch and apply them to the other.

## Resolving Conflicts:

Conflicts can occur when you merge or rebase branches that have diverged, and you need to manually resolve the conflicts before git can proceed with the merge/rebase. git status command shows the files that have conflicts, git diff command shows the difference between the conflicting versions and git add command is used to add the resolved files.

### Task-01

* Create a new branch and make some changes to it.
* Use git stash to save the changes without committing them.
* Switch to a different branch, make some changes and commit them.
* Use git stash pop to bring the changes back and apply them on top of the new commits.

### Task-02

* In version01.txt of development branch add below lines after “This is the bug fix in development branch” that you added in Day10 and reverted to this commit.
* Line2>> After bug fixing, this is the new feature with minor alteration”
* Commit this with message “ Added feature2.1 in development branch”
* Line3>> This is the advancement of previous feature
* Commit this with message “ Added feature2.2 in development branch”
* Line4>> Feature 2 is completed and ready for release
* Commit this with message “ Feature2 completed”
* All these commits messages should be reflected in Production branch too which will come out from Master branch (Hint: try rebase).

### Task-03

* In Production branch Cherry pick Commit “Added feature2.2 in development branch” and added below lines in it:
* Line to be added after Line3>> This is the advancement of previous feature
* Line4>>Added few more changes to make it more optimized.
* Commit: Optimized the feature

### Reference [video](https://youtu.be/apGV9Kg7ics)

You can Post on LinkedIn and let us know what you have learned from this task by #90DaysOfDevOps Challange. Happy Learning :)

## Finally!! 🎉

You have completed the Linux & Git-GitHub handson and I hope you have learned something interesting from it.🙌

Now why not make an interesting 😉 assignment, which not only will help you for the future but also for the DevOps Community!

Let’s make a well articulated and documented **“cheat-sheet”** with all the commands you learned so far in Linux, Git-GitHub and brief info about its usage.

Let’s show us your knowledge mixed with your creativity😎

*I have added a* [*cheatsheet*](https://www.sqltutorial.org/wp-content/uploads/2016/04/SQL-Cheat-Sheet-2.png) *for your reference, Make sure every cheatsheet must be UNIQUE*

Post it on Linkedin and Spread the knowledge.😃

**Happy Learning :)**

Hello Dosto 😎

Let’s Start with Basics of Python as this is also important for Devops Engineer to build the logic and Programs.

**What is Python?**

* Python is a Open source, general purpose, high level, and object-oriented programming language.
* It was created by **Guido van Rossum**
* Python consists of vast libraries and various frameworks like Django,Tensorflow, Flask, Pandas, Keras etc.

**How to Install Python?**

You can install Python in your System whether it is window, MacOS, ubuntu, centos etc. Below are the links for the installation: - [Windows Installation](https://www.python.org/downloads/) - Ubuntu: apt-get install python3.6

Task1: 1. Install Python in your respective OS, and check the version. 2. Read about different Data Types in Python.

You can get the complete Playlist [here](https://www.youtube.com/watch?v=abPgj_3hzVY&list=PLlfy9GnSVerS_L5z0COaF7rsbgWmJXTOM)🙌

Don’t forget to share your Journey over linkedin. Let the community know that you have started another chapter of your Journey.

Happy Learning, Ruko Mat Phod do😃

## Day 14 Task: Python Data Types and Data Structures for DevOps

### New day, New Topic…. Let’s learn along 😉

### Data Types

* Data types are the classification or categorization of data items. It represents the kind of value that tells what operations can be performed on a particular data.
* Since everything is an object in Python programming, data types are actually classes and variables are instance (object) of these classes.
* Python has the following data types built-in by default: Numeric(Integer, complex, float), Sequential(string,lists, tuples), Boolean, Set, Dictionaries, etc

To check what is the data type of the variable used, we can simply write:

```type(your\_variable)```  
  
**Data Structures**  
  
 Data Structures are a way of organizing data so that it can be accessed more efficiently depending upon the situation. Data Structures are fundamentals of any programming language around which a program is built. Python helps to learn the fundamental of these data structures in a simpler way as compared to other programming languages.  
  
- Lists  
Python Lists are just like the arrays, declared in other languages which is an ordered collection of data. It is very flexible as the items in a list do not need to be of the same type  
  
- Tuple  
Python Tuple is a collection of Python objects much like a list but Tuples are immutable in nature i.e. the elements in the tuple cannot be added or removed once created. Just like a List, a Tuple can also contain elements of various types.  
  
- Dictionary  
Python dictionary is like hash tables in any other language with the time complexity of O(1). It is an unordered collection of data values, used to store data values like a map, which, unlike other Data Types that hold only a single value as an element, Dictionary holds the key:value pair. Key-value is provided in the dictionary to make it more optimized  
  
**Tasks**  
1. Give the Difference between List, Tuple and set. Do Handson and put screenshots as per your understanding.  
2. Create below Dictionary and use Dictionary methods to print your favourite tool just by using the keys of the Dictionary.

fav\_tools = { 1:“Linux”, 2:“Git”, 3:“Docker”, 4:“Kubernetes”, 5:“Terraform”, 6:“Ansible”, 7:“Chef” }

3. Create a List of cloud service providers  
eg.

cloud\_providers = [“AWS”,“GCP”,“Azure”]

Write a program to add `Digital Ocean` to the list of cloud\_providers and sort the list in alphabetical order.  
  
[Hint: Use keys to built in functions for Lists]  
  
If you want to deep dive further, Watch [Python](https://youtu.be/abPgj\_3hzVY)  
  
You can share the learning with everyone over linkedin and tag us along 😃

**Day 15 Task: Python Libraries for DevOps**  
  
**Reading JSON and YAML in Python**  
  
- As a DevOps Engineer you should be able to parse files, be it txt, json, yaml, etc.   
- You should know what all libraries one should use in Pythonfor DevOps.  
- Python has numerous libraries like `os`, `sys`, `json`, `yaml` etc that a DevOps Engineer uses in day to day tasks.  
  
  
  
**Tasks**  
1. Create a Dictionary in Python and write it to a json File.  
  
2. Read a json file `services.json` kept in this folder and print the service names of every cloud service provider.

output

aws : ec2 azure : VM gcp : compute engine

3. Read YAML file using python, file `services.yaml` and read the contents to convert yaml to json  
  
Python Project for your practice:  
https://youtube.com/playlist?list=PLlfy9GnSVerSzFmQ8JqP9v0XHHOAeWbjo

**Day 16 Task: Docker for DevOps Engineers**.  
  
 **Docker**  
 Docker is a software platform that allows you to build, test, and deploy applications quickly. Docker packages software into standardized units called containers that have everything the software needs to run including libraries, system tools, code, and runtime. Using Docker, you can quickly deploy and scale applications into any environment and know your code will run.  
  
 **Tasks**  
  
 As you have already installed docker in previous days tasks, now is the time to run Docker commands.  
  
- Use the `docker run` command to start a new container and interact with it through the command line. [Hint: docker run hello-world]  
  
- Use the `docker inspect` command to view detailed information about a container or image.  
  
- Use the `docker port` command to list the port mappings for a container.  
  
- Use the `docker stats` command to view resource usage statistics for one or more containers.  
  
- Use the `docker top` command to view the processes running inside a container.  
  
- Use the `docker save` command to save an image to a tar archive.  
  
- Use the `docker load` command to load an image from a tar archive.  
  
These tasks involve simple operations that can be used to manage images and containers.   
  
For reference you can watch this video:  
https://youtu.be/Tevxhn6Odc8  
  
You can Post on LinkedIn and let us know what you have learned from this task by #90DaysOfDevOps Challange. Happy Learning :)  
  
**Day 17 Task: Docker Project for DevOps Engineers.**  
### You people are doing just amazing in \*\*#90daysofdevops\*\*. Today's challenge is so special Because You are going to do DevOps project today with Docker. Are You Exited 😍  
  
**Dockerfile**  
  
Docker is a tool that makes it easy to run applications in containers. Containers are like small packages that hold everything an application needs to run. To create these containers, developers use something called a Dockerfile.  
  
A Dockerfile is like a set of instructions for making a container. It tells Docker what base image to use, what commands to run, and what files to include. For example, if you were making a container for a website, the Dockerfile might tell Docker to use an official web server image, copy the files for your website into the container, and start the web server when the container starts.  
  
For more about Dockerfile visit [here](https://rushikesh-mashidkar.hashnode.dev/dockerfile-docker-compose-swarm-and-volumes)  
  
**task**:  
  
- Create a Dockerfile for a simple web application (e.g. a Node.js or Python app)  
  
- Build the image using the Dockerfile and run the container  
  
- Verify that the application is working as expected by accessing it in a web browser  
  
- Push the image to a public or private repository (e.g. Docker Hub )  
  
For Refference Project visit [here](https://youtu.be/Tevxhn6Odc8)  
  
If you want to dive further, Watch [bootcamp](https://youtube.com/playlist?list=PLlfy9GnSVerRqYJgVYO0UiExj5byjrW8u)   
  
You can share the learning with everyone over linkedin and tag us along 😃  
  
  
**Day 18 Task: Docker for DevOps Engineers**  
  
Till now you have created Docker file and pushed it to the Repository. Let's move forward and dig more on other Docker concepts.  
Aj thodi padhai krte hai on Docker Compose 😃

**Docker Compose**  
- Docker Compose is a tool that was developed to help define and share multi-container applications.  
- With Compose, we can create a YAML file to define the services and with a single command, can spin everything up or tear it all down.  
- Learn more about docker-compose [visit here](https://tecadmin.net/tutorial/docker/docker-compose/)  
  
## What is YAML?  
- YAML is a data serialization language that is often used for writing configuration files. Depending on whom you ask, YAML stands for yet another markup language or YAML ain’t markup language (a recursive acronym), which emphasizes that YAML is for data, not documents.   
- YAML is a popular programming language because it is human-readable and easy to understand.  
- YAML files use a .yml or .yaml extension.  
- Read more about it [here](https://www.redhat.com/en/topics/automation/what-is-yaml)  
  
**Task-1**  
  
Learn how to use the docker-compose.yml file, to set up the environment, configure the services and links between different containers, and also to use environment variables in the docker-compose.yml file.   
  
[Sample docker-compose.yaml file](https://github.com/LondheShubham153/90DaysOfDevOps/blob/master/2023/day18/docker-compose.yaml)  
  
  
 **Task-2**  
- Pull a pre-existing Docker image from a public repository (e.g. Docker Hub) and run it on your local machine. Run the container as a non-root user (Hint- Use `usermod ` command to give user permission to docker). Make sure you reboot instance after giving permission to user.  
- Inspect the container's running processes and exposed ports using the docker inspect command.  
- Use the docker logs command to view the container's log output.  
- Use the docker stop and docker start commands to stop and start the container.  
- Use the docker rm command to remove the container when you're done.  
  
## How to run Docker commands without sudo?  
- Make sure docker is installed and system is updated (This is already been completed as a part of previous tasks):  
- sudo usermod -a -G docker $USER   
- Reboot the machine.  
  
For reference you can watch this [video](https://youtu.be/Tevxhn6Odc8)  
  
You can Post on LinkedIn and let us know what you have learned from this task by #90DaysOfDevOps Challange. Happy Learning :)

**Day 19 Task: Docker for DevOps Engineers**  
  
\*\*Till now you have learned how to create docker-compose.yml file and pushed it to the Repository. Let's move forward and dig more on other Docker-compose.yml concepts.\*\*  
\*\*Aaj thodi padhai krte hai on Docker Volume & Docker Network\*\* 😃  
  
**Docker-Volume**  
Docker allows you to create something called volumes. Volumes are like separate storage areas that can be accessed by containers. They allow you to store data, like a database, outside the container, so it doesn't get deleted when the container is deleted.  
You can also mount from the same volume and create more containers having same data.  
[reference](https://docs.docker.com/storage/volumes/)  
  
**Docker Network**  
Docker allows you to create virtual spaces called networks, where you can connect multiple containers (small packages that hold all the necessary files for a specific application to run) together. This way, the containers can communicate with each other and with the host machine (the computer on which the Docker is installed).  
When we run a container, it has its own storage space that is only accessible by that specific container. If we want to share that storage space with other containers, we can't do that. [reference](https://docs.docker.com/network/)  
  
 **Task-1**  
- Create a multi-container docker-compose file which will bring \*UP\* and bring \*DOWN\* containers in a single shot ( Example - Create application and database container )  
  
\*hints:\*  
- Use the `docker-compose up` command with the `-d` flag to start a multi-container application in detached mode.  
- Use the `docker-compose scale` command to increase or decrease the number of replicas for a specific service. You can also add [`replicas`](https://stackoverflow.com/questions/63408708/how-to-scale-from-within-docker-compose-file) in deployment file for \*auto-scaling\*.  
- Use the `docker-compose ps` command to view the status of all containers, and `docker-compose logs` to view the logs of a specific service.  
- Use the `docker-compose down` command to stop and remove all containers, networks, and volumes associated with the application  
  
**Task-2**  
- Learn how to use Docker Volumes and Named Volumes to share files and directories between multiple containers.  
- Create two or more containers that read and write data to the same volume using the `docker run --mount` command.  
- Verify that the data is the same in all containers by using the docker exec command to run commands inside each container.  
- Use the docker volume ls command to list all volumes and docker volume rm command to remove the volume when you're done.  
  
## You can use this task as \*Project\* to add in your resume.  
  
You can Post on LinkedIn and let us know what you have learned from this task by #90DaysOfDevOps Challange. Happy Learning :)  
  
 **Finally!! 🎉**  
You have completed✅ the Docker handson and I hope you have learned something interesting from it.🙌  
  
Now it's time to take your Docker skills to the next level by creating a comprehensive cheat-sheet of all the commands you've learned so far. This cheat-sheet should include commands for both Docker and Docker-Compose, as well as brief explanations of their usage.  
This cheat-sheet will not only help you in the future but also contribute to the DevOps community by providing a useful resource for others.😊🙌  
  
  
So, put your knowledge and creativity to the test and create a cheat-sheet that truly stands out! 🚀  
  
\*I have added a [cheatsheet](https://cdn.hashnode.com/res/hashnode/image/upload/v1670863735841/r6xdXpsap.png?auto=compress,format&format=webp) for your reference, Make sure every cheatsheet must be UNIQUE\*  
  
Post it on Linkedin and Spread the knowledge.😃   
  
\*\*Happy Learning :)\*\*   
**Day 21 Task: Docker Important interview Questions**.  
  
 **Docker Interview**  
 Docker is a good topic to ask in DevOps Engineer Interviews, mostly for freshers.  
 One must surely try these questions in order to be better in Docker  
   
## Questions  
  
  
- What is the Difference between an Image, Container and Engine?  
- What is the Difference between the Docker command COPY vs ADD?  
- What is the Difference between the Docker command CMD vs RUN?  
- How Will you reduce the size of the Docker image?  
- Why and when to use Docker?  
- Explain the Docker components and how they interact with each other.  
- Explain the terminology: Docker Compose, Docker File, Docker Image, Docker Container?  
- In what real scenarios have you used Docker?  
- Docker vs Hypervisor?  
- What are the advantages and disadvantages of using docker?  
- What is a Docker namespace?  
- What is a Docker registry?  
- What is an entry point?  
- How to implement CI/CD in Docker?  
- Will data on the container be lost when the docker container exits?  
- What is a Docker swarm?  
- What are the docker commands for the following:  
 - view running containers  
 - command to run the container under a specific name  
 - command to export a docker  
 - command to import an already existing docker image  
 - commands to delete a container  
 - command to remove all stopped containers, unused networks, build caches, and dangling images?  
- What are the common docker practices to reduce the size of Docker Image?  
  
  
These questions will help you in your next DevOps Interview.  
\*Write a Blog and share it on LinkedIn.\*  
  
\*\*Happy Learning :)\*\*

**Day-22 : Getting Started with Jenkins 😃**  
  
\*\*Linux, Git, Git-Hub, Docker finish ho chuka hai to chaliye seekhte hai inko deploy krne ke lye CI-CD tool:\*\*  
  
## What is Jenkins?  
- Jenkins is an open source continuous integration-continuous delivery and deployment (CI/CD) automation software DevOps tool written in the Java programming language. It is used to implement CI/CD workflows, called pipelines.  
  
- Jenkins is a tool that is used for automation, and it is an open-source server that allows all the developers to build, test and deploy software. It works or runs on java as it is written in java. By using Jenkins we can make a continuous integration of projects(jobs) or end-to-endpoint automation.  
  
- Jenkins achieves Continuous Integration with the help of plugins. Plugins allow the integration of Various DevOps stages. If you want to integrate a particular tool, you need to install the plugins for that tool. For example Git, Maven 2 project, Amazon EC2, HTML publisher etc.  
  
\*\*Let us do discuss the necessity of this tool before going ahead to the procedural part for installation:\*\*   
- Nowadays, humans are becoming lazy😴 day by day so even having digital screens and just one click button in front of us then also need some automation.   
  
- Here, I’m referring to that part of automation where we need not have to look upon a process(here called a job) for completion and after it doing another job. For that, we have Jenkins with us.  
  
Note: By now Jenkins should be installed on your machine(as it was a part of previous tasks, if not follow [Installation Guide](https://youtu.be/OkVtBKqMt7I))  
  
   
## Tasks:   
  
\*\*1. What you understood in Jenkin, write a small article in your own words (Don't copy from Internet Directly)\*\*  
  
\*\*2.Create a freestyle pipeline to print "Hello World!!\*\*  
Hint: Use link for [Article](https://www.geeksforgeeks.org/what-is-jenkins)  
  
Don't forget to post your progress on Linkedin. Till then Happy learning :)   
  
**Day 23 Task: Jenkins Freestyle Project for DevOps Engineers**.  
  
 The Community is absolutely crushing it in the #90daysofdevops journey. Today's challenge is particularly exciting as it entails creating a Jenkins Freestyle Project, an opportunity for DevOps engineers to showcase their skills and push their limits. Who's ready to dive in and make it happen? 😍  
  
## What is CI/CD?  
- CI or Continuous Integration is the practice of automating the integration of code changes from multiple developers into a single codebase. It is a software development practice where the developers commit their work frequently into the central code repository (Github or Stash). Then there are automated tools that build the newly committed code and do a code review, etc as required upon integration.  
 The key goals of Continuous Integration are to find and address bugs quicker, make the process of integrating code across a team of developers easier, improve software quality and reduce the time it takes to release new feature updates.   
  
  
- CD or Continuous Delivery is carried out after Continuous Integration to make sure that we can release new changes to our customers quickly in an error-free way. This includes running integration and regression tests in the staging area (similar to the production environment) so that the final release is not broken in production. It ensures to automate the release process so that we have a release-ready product at all times and we can deploy our application at any point in time.   
  
## What Is a Build Job?  
A Jenkins build job contains the configuration for automating a specific task or step in the application building process. These tasks include gathering dependencies, compiling, archiving, or transforming code, and testing and deploying code in different environments.  
  
Jenkins supports several types of build jobs, such as freestyle projects, pipelines, multi-configuration projects, folders, multibranch pipelines, and organization folders.  
  
## What is Freestyle Projects ?? 🤔  
A freestyle project in Jenkins is a type of project that allows you to build, test, and deploy software using a variety of different options and configurations. Here are a few tasks that you could complete when working with a freestyle project in Jenkins:  
  
  
**Task-01**  
- create a agent for your app. ( which you deployed from docker in earlier task)  
- Create a new Jenkins freestyle project for your app.  
- In the "Build" section of the project, add a build step to run the "docker build" command to build the image for the container.  
- Add a second step to run the "docker run" command to start a container using the image created in step 3.  
  
  
**Task-02**  
- Create Jenkins project to run "docker-compose up -d" command to start the multiple containers defined in the compose file (Hint- use day-19 Application & Database docker-compose file)  
- Set up a cleanup step in the Jenkins project to run "docker-compose down" command to stop and remove the containers defined in the compose file.  
  
For Refference jenkins Freestyle Project visit [here](https://youtu.be/wwNWgG5htxs)  
  
You can Post on LinkedIn and let us know what you have learned from this task by #90DaysOfDevOps Challange.  
  
Happy Learning:)  
  
  
**Day 24 Task: Complete Jenkins CI/CD Project**   
  
   
  
Let's make a beautiful CI/CD Pipeline for your Node JS Application 😍  
  
   
  
## Did you finish Day 23?  
  
- Day 23 was all about Jenkins CI/CD, make sure you have done it and understood the concepts. As today You will be doing one Project End to End and adding it to your resume :)  
   
- As you have worked with Docker and Docker compose, it will be good to use it in a live project.  
   
  
**Task-01**  
  
- Fork [this](https://github.com/LondheShubham153/node-todo-cicd.git) repository:  
- Create a connection to your Jenkins job and your GitHub Repository via GitHub Integration.   
- Read About [GitHub WebHooks](https://betterprogramming.pub/how-too-add-github-webhook-to-a-jenkins-pipeline-62b0be84e006) and make sure you have CICD setup   
- Refer [this](https://youtu.be/nplH3BzKHPk) video for the entire project  
  
**Task-02**  
 - In the Execute shell run the application using Docker compose  
 - You will have to make a Docker Compose file for this Project (Can be a good open source contribution)  
 - Run the project and give yourself a treat:)  
  
For Reference and entire hands-on Project visit [here](https://youtu.be/nplH3BzKHPk)  
  
   
  
You can Post on LinkedIn and let us know what you have learned from this task by #90DaysOfDevOps Challange.  
  
   
  
Happy Learning:)  
  
  
   
  
 **Day 25 Task: Complete Jenkins CI/CD Project - Continued with Documentation**  
  
   
  
   
  
I can imagine catching up will be tough so take a small breather today and complete the Jenkins CI/CD project from Day 24 and add a documentation.  
  
   
  
   
  
## Did you finish Day 24?  
  
   
  
- Day 24 will give you an End to End project and adding it to your resume will be a cherry on the top.  
  
- take more time, finish the project, add a Documentation, add it to your Resume and post about it today.  
  
   
  
**Task-01**  
  
   
  
- Document the process from cloning the repository to adding webhooks, and Deployment, etc. as a README , go through [this example](https://github.com/LondheShubham153/fynd-my-movie/blob/master/README.md)  
  
   
  
- A well written readme file will help others to understand your project and you will understand how to use the project again without any problems.  
  
   
   
  
**Task-02**  
  
   
  
- Also it's important to keep smaller goals, as its a small task, think of a small Goal you can accomplish.  
  
   
  
- Write about it using [this template](https://www.linkedin.com/posts/shubhamlondhe1996\_taking-resolutions-and-having-goals-for-an-activity-7023858409762373632-s2J8?utm\_source=share&utm\_medium=member\_desktop)  
  
   
  
- Have small goals and strategies to achieve them, also have a small reward for yourself.  
  
   
  
For Reference and entire hands-on Project visit [here](https://youtu.be/nplH3BzKHPk)  
  
   
You can Post on LinkedIn and let us know what you have learned from this task by #90DaysOfDevOps Challange.  
  
  
Happy Learning:)  
  
**Day 26 Task: Jenkins Declarative Pipeline**  
  
   
One of the most important parts of your DevOps and CICD journey is a Declarative Pipeline Syntax of Jenkins  
   
  
## Some terms for your Knowledge  
  
\*\*What is Pipeline -\*\* A pipeline is a collection of steps or jobs interlinked in a sequence.  
  
\*\*Declarative:\*\* Declarative is a more recent and advanced implementation of a pipeline as a code.  
  
\*\*Scripted:\*\* Scripted was the first and most traditional implementation of the pipeline as a code in Jenkins. It was designed as a general-purpose DSL (Domain Specific Language) built with Groovy.  
  
# Why you should have a Pipeline  
  
The definition of a Jenkins Pipeline is written into a text file (called a [`Jenkinsfile`](https://www.jenkins.io/doc/book/pipeline/jenkinsfile)) which in turn can be committed to a project’s source control repository.   
This is the foundation of "Pipeline-as-code"; treating the CD pipeline as a part of the application to be versioned and reviewed like any other code.  
  
\*\*Creating a `Jenkinsfile` and committing it to source control provides a number of immediate benefits:\*\*  
  
- Automatically creates a Pipeline build process for all branches and pull requests.  
   
- Code review/iteration on the Pipeline (along with the remaining source code).  
  
  
# Pipeline syntax  
  
````groovy  
pipeline {  
 agent any   
 stages {  
 stage('Build') {   
 steps {  
 //   
 }  
 }  
 stage('Test') {   
 steps {  
 //   
 }  
 }  
 stage('Deploy') {   
 steps {  
 //   
 }  
 }  
 }  
}

### Task-01

* Create a New Job, this time select Pipeline instead of Freestyle Project.
* Follow the Official Jenkins [Hello world example](https://www.jenkins.io/doc/pipeline/tour/hello-world/)
* Complete the example using the Declarative pipeline
* In case of any issues feel free to post on any Groups, [Discord](https://discord.gg/Q6ntmMtH) or [Telegram](https://t.me/trainwithshubham)

You can post your progress on LinkedIn and let us know what you have learned from this task by #90DaysOfDevOps Challenge.

Happy Learning:)

# Day 27 Task: Jenkins Declarative Pipeline with Docker

Day 26 was all about a Declarative pipeline, now its time to level up things, let’s integrate Docker and your Jenkins declarative pipeline

## Use your Docker Build and Run Knowledge

**docker build -** you can use sh 'docker build . -t <tag>' in your pipeline stage block to run the docker build command. (Make sure you have docker installed with correct permissions.

**docker run:** you can use sh 'docker run -d <image>' in your pipeline stage block to build the container.

**How will the stages look**

stages {  
 stage('Build') {  
 steps {  
 sh 'docker build -t trainwithshubham/django-app:latest'  
 }  
 }  
 }

# Task-01

* Create a docker-integrated Jenkins declarative pipeline
* Use the above-given syntax using sh inside the stage block
* You will face errors in case of running a job twice, as the docker container will be already created, so for that do task 2

# Task-02

* Create a docker-integrated Jenkins declarative pipeline using the docker groovy syntax inside the stage block.
* You won’t face errors, you can Follow [this documentation](https://tempora-mutantur.github.io/jenkins.io/github_pages_test/doc/book/pipeline/docker/)
* Complete your previous projects using this Declarative pipeline approach
* In case of any issues feel free to post on any Groups, [Discord](https://discord.gg/Q6ntmMtH) or [Telegram](https://t.me/trainwithshubham)

Are you enjoying the #90DaysOfDevOps Challenge? Let me know how are feeling after 4 weeks of DevOps Learnings,

Happy Learning:)

# Day 28 Task: Jenkins Agents

### Jenkins Master (Server)

Jenkins’s server or master node holds all key configurations. Jenkins master server is like a control server that orchestrates all the workflow defined in the pipelines. For example, scheduling a job, monitoring the jobs, etc.

### Jenkins Agent

An agent is typically a machine or container that connects to a Jenkins master and this agent that actually execute all the steps mentioned in a Job. When you create a Jenkins job, you have to assign an agent to it. Every agent has a label as a unique identifier.

When you trigger a Jenkins job from the master, the actual execution happens on the agent node that is configured in the job.

A single, monolithic Jenkins installation can work great for a small team with a relatively small number of projects. As your needs grow, however, it often becomes necessary to scale up. Jenkins provides a way to do this called “master to agent connection.” Instead of serving the Jenkins UI and running build jobs all on a single system, you can provide Jenkins with agents to handle the execution of jobs while the master serves the Jenkins UI and acts as a control node.

### Pre-requisites

Let’s say we’re starting with a fresh Ubuntu 22.04 Linux installation. To get an agent working make sure you install Java ( same version as jenkins master server ) and Docker on it.

Note:- While creating an agent, be sure to separate rights, permissions, and ownership for jenkins users.

### Task-01

* Create an agent by setting up a node on Jenkins
* Create a new AWS EC2 Instance and connect it to master(Where Jenkins is installed)
* The connection of master and agent requires SSH and the public-private key pair exchange.
* Verify its status under “Nodes” section.
* You can follow [this article](https://www.linkedin.com/posts/chetanrakhra_devops-project-share-activity-7017885886461698048-os5f?utm_source=share&utm_medium=member_android) for the same

### Task-02

* Run your previous Jobs (which you built on Day 26, and Day 27) on the new agent
* Use labels for the agent, your master server should trigger builds for the agent server.
* In case of any issues feel free to post on any Groups, [Discord](https://discord.gg/Q6ntmMtH) or [Telegram](https://t.me/trainwithshubham)

Are you enjoying the #90DaysOfDevOps Challenge?

Let me know how are feeling after 4 weeks of DevOps Learning.

Happy Learning:)

# Day 29 Task: Jenkins Important interview Questions.

## Jenkins Interview

Here are some Jenkins-specific questions related to Docker that one can use during a DevOps Engineer interview:

## Questions

1. What’s the difference between continuous integration, continuous delivery, and continuous deployment?
2. Benefits of CI/CD
3. What is meant by CI-CD?
4. What is Jenkins Pipeline?
5. How do you configure the job in Jenkins?
6. Where do you find errors in Jenkins?
7. In Jenkins how can you find log files?
8. Jenkins workflow and write a script for this workflow?
9. How to create continuous deployment in Jenkins?
10. How build job in Jenkins?
11. Why we use pipeline in Jenkins?
12. Is Only Jenkins enough for automation?
13. How will you handle secrets?
14. Explain diff stages in CI-CD setup
15. Name some of the plugins in Jenkin?

These questions will help you in your next DevOps Interview. Write a Blog and share it on LinkedIn.

*Happy Learning :)*

# Day 30 Task: Kubernetes Architecture

## Kubernetes Overview

With the widespread adoption of [containers](https://cloud.google.com/containers) among organizations, Kubernetes, the container-centric management software, has become a standard to deploy and operate containerized applications and is one of the most important parts of DevOps.

Originally developed at Google and released as open-source in 2014. Kubernetes builds on 15 years of running Google’s containerized workloads and the valuable contributions from the open-source community. Inspired by Google’s internal cluster management system, [Borg](https://research.google.com/pubs/pub43438.html),

## Tasks

1. What is Kubernetes? Write in your own words and why do we call it k8s?
2. What are the benefits of using k8s?
3. Explain the architecture of Kubernetes, refer to [this video](https://youtu.be/FqfoDUhzyDo)
4. What is Control Plane?
5. Write the difference between kubectl and kubelets.
6. Explain the role of the API server.

Kubernetes architecture is important, so make sure you spend a day understanding it. [This video](https://youtu.be/FqfoDUhzyDo) will surely help you.

*Happy Learning :)*

# Day 31 Task: Launching your First Kubernetes Cluster with Nginx running

### Awesome! You learned the architecture of one of the top most important tool “Kubernetes” in your previous task.

## What about doing some hands-on now?

Let’s read about minikube and implement *k8s* in our local machine

1. **What is minikube?**

*Ans*:- Minikube is a tool which quickly sets up a local Kubernetes cluster on macOS, Linux, and Windows. It can deploy as a VM, a container, or on bare-metal.

Minikube is a pared-down version of Kubernetes that gives you all the benefits of Kubernetes with a lot less effort.

This makes it an interesting option for users who are new to containers, and also for projects in the world of edge computing and the Internet of Things.

1. **Features of minikube**

*Ans* :-

1. Supports the latest Kubernetes release (+6 previous minor versions)
2. Cross-platform (Linux, macOS, Windows)
3. Deploy as a VM, a container, or on bare-metal
4. Multiple container runtimes (CRI-O, containerd, docker)
5. Direct API endpoint for blazing fast image load and build
6. Advanced features such as LoadBalancer, filesystem mounts, FeatureGates, and network policy
7. Addons for easily installed Kubernetes applications
8. Supports common CI environments

## Task-01:

## Install minikube on your local

For installation, you can Visit [this page](https://minikube.sigs.k8s.io/docs/start/).

If you want to try an alternative way, you can check [this](https://k8s-docs.netlify.app/en/docs/tasks/tools/install-minikube/).

## Let’s understand the concept pod

*Ans:-*

Pods are the smallest deployable units of computing that you can create and manage in Kubernetes.

A Pod (as in a pod of whales or pea pod) is a group of one or more containers, with shared storage and network resources, and a specification for how to run the containers. A Pod’s contents are always co-located and co-scheduled, and run in a shared context. A Pod models an application-specific “logical host”: it contains one or more application containers which are relatively tightly coupled.

You can read more about pod from [here](https://kubernetes.io/docs/concepts/workloads/pods/) .

## Task-02:

## Create your first pod on Kubernetes through minikube.

We are suggesting you make an nginx pod, but you can always show your creativity and do it on your own.

**Having an issue? Don’t worry, adding a sample yaml file for pod creation, you can always refer that.**

*Happy Learning :)*

# Day 32 Task: Launching your Kubernetes Cluster with Deployment

### Congratulation ! on your learning on K8s on Day-31

## What is Deployment in k8s

A Deployment provides a configuration for updates for Pods and ReplicaSets.

You describe a desired state in a Deployment, and the Deployment Controller changes the actual state to the desired state at a controlled rate. You can define Deployments to create new replicas for scaling, or to remove existing Deployments and adopt all their resources with new Deployments.

## Today’s task let’s keep it very simple.

## Task-1:

**Create one Deployment file to deploy a sample todo-app on K8s using “Auto-healing” and “Auto-Scaling” feature**

* add a deployment.yml file (sample is kept in the folder for your reference)
* apply the deployment to your k8s (minikube) cluster by command kubectl apply -f deployment.yml

Let’s make your resume shine with one more project ;)

**Having an issue? Don’t worry, adding a sample deployment file , you can always refer that or wathch** [**this video**](https://youtu.be/ONrbWFJXLLk)

Happy Learning :)

# Day 33 Task: Working with Namespaces and Services in Kubernetes

### Congrats🎊🎉 on updating your Deployment yesterday💥🙌

## What are Namespaces and Services in k8s

In Kubernetes, Namespaces are used to create isolated environments for resources. Each Namespace is like a separate cluster within the same physical cluster. Services are used to expose your Pods and Deployments to the network. Read more about Namespace [Here](https://kubernetes.io/docs/concepts/workloads/pods/user-namespaces/)

## Today’s task:

## Task 1:

* Create a Namespace for your Deployment
* Use the command kubectl create namespace <namespace-name> to create a Namespace
* Update the deployment.yml file to include the Namespace
* Apply the updated deployment using the command: kubectl apply -f deployment.yml -n <namespace-name>
* Verify that the Namespace has been created by checking the status of the Namespaces in your cluster.

## Task 2:

* Read about Services, Load Balancing, and Networking in Kubernetes. Refer official documentation of kubernetes [Link](https://kubernetes.io/docs/concepts/services-networking/)

Need help with Namespaces? Check out this [video](https://youtu.be/K3jNo4z5Jx8) for assistance.

Keep growing your Kubernetes knowledge💥🙌

Happy Learning! :)

# Day 34 Task: Working with Services in Kubernetes

### Congratulation🎊 on your learning on Deployments in K8s on Day-33

## What are Services in K8s

In Kubernetes, Services are objects that provide stable network identities to Pods and abstract away the details of Pod IP addresses. Services allow Pods to receive traffic from other Pods, Services, and external clients.

## Task-1:

* Create a Service for your todo-app Deployment from Day-32
* Create a Service definition for your todo-app Deployment in a YAML file.
* Apply the Service definition to your K8s (minikube) cluster using the kubectl apply -f service.yml -n <namespace-name> command.
* Verify that the Service is working by accessing the todo-app using the Service’s IP and Port in your Namespace.

## Task-2:

* Create a ClusterIP Service for accessing the todo-app from within the cluster
* Create a ClusterIP Service definition for your todo-app Deployment in a YAML file.
* Apply the ClusterIP Service definition to your K8s (minikube) cluster using the kubectl apply -f cluster-ip-service.yml -n <namespace-name> command.
* Verify that the ClusterIP Service is working by accessing the todo-app from another Pod in the cluster in your Namespace.

## Task-3:

* Create a LoadBalancer Service for accessing the todo-app from outside the cluster
* Create a LoadBalancer Service definition for your todo-app Deployment in a YAML file.
* Apply the LoadBalancer Service definition to your K8s (minikube) cluster using the kubectl apply -f load-balancer-service.yml -n <namespace-name> command.
* Verify that the LoadBalancer Service is working by accessing the todo-app from outside the cluster in your Namespace.

Struggling with Services? Take a look at this video for a step-by-step [guide](https://youtu.be/OJths_RojFA).

Need help with Services in Kubernetes? Check out the Kubernetes [documentation](https://kubernetes.io/docs/concepts/services-networking/service/) for assistance.

Happy Learning :)

# Day 35: Mastering ConfigMaps and Secrets in Kubernetes🔒🔑🛡️

### 👏🎉 Yay! Yesterday we conquered Namespaces and Services 💪💻🔗🚀

## What are ConfigMaps and Secrets in k8s

In Kubernetes, ConfigMaps and Secrets are used to store configuration data and secrets, respectively. ConfigMaps store configuration data as key-value pairs, while Secrets store sensitive data in an encrypted form.

* *Example :- Imagine you’re in charge of a big spaceship (Kubernetes cluster) with lots of different parts (containers) that need information to function properly. ConfigMaps are like a file cabinet where you store all the information each part needs in simple, labeled folders (key-value pairs). Secrets, on the other hand, are like a safe where you keep the important, sensitive information that shouldn’t be accessible to just anyone (encrypted data). So, using ConfigMaps and Secrets, you can ensure each part of your spaceship (Kubernetes cluster) has the information it needs to work properly and keep sensitive information secure! 🚀*
* Read more about [ConfigMap](https://kubernetes.io/docs/concepts/configuration/configmap/) & [Secret](https://kubernetes.io/docs/concepts/configuration/secret/). ## Today’s task: ## Task 1:
* Create a ConfigMap for your Deployment
* Create a ConfigMap for your Deployment using a file or the command line
* Update the deployment.yml file to include the ConfigMap
* Apply the updated deployment using the command: kubectl apply -f deployment.yml -n <namespace-name>
* Verify that the ConfigMap has been created by checking the status of the ConfigMaps in your Namespace.

## Task 2:

* Create a Secret for your Deployment
* Create a Secret for your Deployment using a file or the command line
* Update the deployment.yml file to include the Secret
* Apply the updated deployment using the command: kubectl apply -f deployment.yml -n <namespace-name>
* Verify that the Secret has been created by checking the status of the Secrets in your Namespace.

Need help with ConfigMaps and Secrets? Check out this [video](https://youtu.be/FAnQTgr04mU) for assistance.

Keep learning and expanding your knowledge of Kubernetes💥🙌

# Day 36 Task: Managing Persistent Volumes in Your Deployment 💥

🙌 Kudos to you for conquering ConfigMaps and Secrets in Kubernetes yesterday.

🔥 You’re on fire! 🔥

## What are Persistent Volumes in k8s

In Kubernetes, a Persistent Volume (PV) is a piece of storage in the cluster that has been provisioned by an administrator. A Persistent Volume Claim (PVC) is a request for storage by a user. The PVC references the PV, and the PV is bound to a specific node. Read official documentation of [Persistent Volumes](https://kubernetes.io/docs/concepts/storage/persistent-volumes/).

⏰ Wait, wait, wait! 📣 Attention all #90daysofDevOps Challengers. 💪

Before diving into today’s task, don’t forget to share your thoughts on the #90daysofDevOps challenge 💪 Fill out our feedback form (https://lnkd.in/gcgvrq8b) to help us improve and provide the best experience 🌟 Your participation and support is greatly appreciated 🙏 Let’s continue to grow together 🌱

## Today’s tasks:

### Task 1:

Add a Persistent Volume to your Deployment todo app.

* Create a Persistent Volume using a file on your node. [Template](https://github.com/LondheShubham153/90DaysOfDevOps/blob/94e3970819e097a5b8edea40fe565d583419f912/2023/day36/pv.yml)
* Create a Persistent Volume Claim that references the Persistent Volume. [Template](https://github.com/LondheShubham153/90DaysOfDevOps/blob/94e3970819e097a5b8edea40fe565d583419f912/2023/day36/pvc.yml)
* Update your deployment.yml file to include the Persistent Volume Claim. After Applying pv.yml pvc.yml your deployment file look like this [Template](https://github.com/LondheShubham153/90DaysOfDevOps/blob/94e3970819e097a5b8edea40fe565d583419f912/2023/day36/Deployment.yml)
* Apply the updated deployment using the command: kubectl apply -f deployment.yml
* Verify that the Persistent Volume has been added to your Deployment by checking the status of the Pods and Persistent Volumes in your cluster. Use this commands kubectl get pods ,

kubectl get pv

⚠️ Don’t forget: To apply changes or create files in your Kubernetes deployments, each file must be applied separately. ⚠️

### Task 2:

Accessing data in the Persistent Volume,

* Connect to a Pod in your Deployment using command : `kubectl exec -it – /bin/bash

`

* Verify that you can access the data stored in the Persistent Volume from within the Pod

Need help with Persistent Volumes? Check out this [video](https://youtu.be/U0_N3v7vJys) for assistance.

Keep up the excellent work🙌💥

Happy Learning :)

# Day 37 Task: Kubernetes Important interview Questions.

## Questions

1.What is Kubernetes and why it is important?

2.What is difference between docker swarm and kubernetes?

3.How does Kubernetes handle network communication between containers?

4.How does Kubernetes handle scaling of applications?

5.What is a Kubernetes Deployment and how does it differ from a ReplicaSet?

6.Can you explain the concept of rolling updates in Kubernetes?

7.How does Kubernetes handle network security and access control?

8.Can you give an example of how Kubernetes can be used to deploy a highly available application?

9.What is namespace is kubernetes? Which namespace any pod takes if we don’t specify any namespace?

10.How ingress helps in kubernetes?

11.Explain different types of services in kubernetes?

12.Can you explain the concept of self-healing in Kubernetes and give examples of how it works?

13.How does Kubernetes handle storage management for containers?

14.How does the NodePort service work?

15.What is a multinode cluster and single-node cluster in Kubernetes?

16.Difference between create and apply in kubernetes?

## These questions will help you in your next DevOps Interview.

*Write a Blog and share it on LinkedIn.*

***Happy Learning :)***

# Day 38 Getting Started with AWS Basics☁



AWS

Congratulations!!!! you have come so far. Don’t let your excuses break your consistency. Let’s begin our new Journey with Cloud☁. By this time you have created multiple EC2 instances, if not let’s begin the journey: ## AWS: Amazon Web Services is one of the most popular Cloud Provider that has free tier too for students and Cloud enthutiasts for their Handson while learning (Create your free account today to explore more on it).

Read from [here](https://aws.amazon.com/what-is-aws/)

## IAM:

AWS Identity and Access Management (IAM) is a web service that helps you securely control access to AWS resources. With IAM, you can centrally manage permissions that control which AWS resources users can access. You use IAM to control who is authenticated (signed in) and authorized (has permissions) to use resources. Read from [here](https://docs.aws.amazon.com/IAM/latest/UserGuide/introduction.html)

Get to know IAM more deeply [Click Here!!](https://www.youtube.com/watch?v=ORB4eY8EydA)

### Task1:

Create an IAM user with username of your own wish and grant EC2 Access. Launch your Linux instance through the IAM user that you created now and install jenkins and docker on your machine via single Shell Script.

### Task2:

In this task you need to prepare a devops team of avengers. Create 3 IAM users of avengers and assign them in devops groups with IAM policy.

Post your progress on Linkedin. Till then Happy Learning :)

# Day 39 AWS and IAM Basics☁



AWS

By this time you have created multiple EC2 instances, and post installation manually installed applications like Jenkins, docker etc. Now let’s switch to little automation part. Sounds interesting??🤯

## AWS:

Amazon Web Services is one of the most popular Cloud Provider that has free tier too for students and Cloud enthutiasts for their Handson while learning (Create your free account today to explore more on it).

Read from [here](https://aws.amazon.com/what-is-aws/)

## User Data in AWS:

* When you launch an instance in Amazon EC2, you have the option of passing user data to the instance that can be used to perform common automated configuration tasks and even run scripts after the instance starts. You can pass two types of user data to Amazon EC2: shell scripts and cloud-init directives.
* You can also pass this data into the launch instance wizard as plain text, as a file (this is useful for launching instances using the command line tools), or as base64-encoded text (for API calls).
* This will save time and manual effort everytime you launch an instance and want to install any application on it like apache, docker, Jenkins etc

Read more from [here](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/user-data.html)

## IAM:

AWS Identity and Access Management (IAM) is a web service that helps you securely control access to AWS resources. With IAM, you can centrally manage permissions that control which AWS resources users can access. You use IAM to control who is authenticated (signed in) and authorized (has permissions) to use resources. Read from [here](https://docs.aws.amazon.com/IAM/latest/UserGuide/introduction.html)

Get to know IAM more deeply🏊[Click Here!!](https://www.youtube.com/watch?v=ORB4eY8EydA)

### Task1:

* Launch EC2 instance with already installed Jenkins on it. Once server shows up in console, hit the IP address in browser and you Jenkins page should be visible.
* Take screenshot of Userdata and Jenkins page, this will verify the task completion.

### Task2:

* Read more on IAM Roles and explain the IAM Users, Groups and Roles in your own terms.
* Create three Roles named: DevOps-User, Test-User and Admin.

Post your progress on Linkedin. Till then Happy Learning :)

# Day 40 AWS EC2 Automation ☁



AWS

I hope your journey with AWS cloud and automation is going well ### 😍

## Automation in EC2:

Amazon EC2 or Amazon Elastic Compute Cloud can give you secure, reliable, high-performance, and cost-effective computing infrastructure to meet demanding business needs.

Also, if you know a few things, you can automate many things.

Read from [here](https://aws.amazon.com/ec2/)

## Launch template in AWS EC2:

* You can make a launch template with the configuration information you need to start an instance. You can save launch parameters in launch templates so you don’t have to type them in every time you start a new instance.
* For example, a launch template can have the AMI ID, instance type, and network settings that you usually use to launch instances.
* You can tell the Amazon EC2 console to use a certain launch template when you start an instance.

Read more from [here](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-launch-templates.html)

## Instance Types:

Amazon EC2 has a large number of instance types that are optimised for different uses. The different combinations of CPU, memory, storage and networking capacity in instance types give you the freedom to choose the right mix of resources for your apps. Each instance type comes with one or more instance sizes, so you can adjust your resources to meet the needs of the workload you want to run.

Read from [here](https://aws.amazon.com/ec2/instance-types/?trk=32f4fbd0-ffda-4695-a60c-8857fab7d0dd&sc_channel=ps&s_kwcid=AL!4422!3!536392685920!e!!g!!ec2%20instance%20types&ef_id=CjwKCAiA0JKfBhBIEiwAPhZXD_O1-3qZkRa-KScynbwjvHd3l4UHSTfKuigd5ZPukXoDXu-v3MtC7hoCafEQAvD_BwE:G:s&s_kwcid=AL!4422!3!536392685920!e!!g!!ec2%20instance%20types)

## AMI:

An Amazon Machine Image (AMI) is an image that AWS supports and keeps up to date. It contains the information needed to start an instance. When you launch an instance, you must choose an AMI. When you need multiple instances with the same configuration, you can launch them from a single AMI.

### Task1:

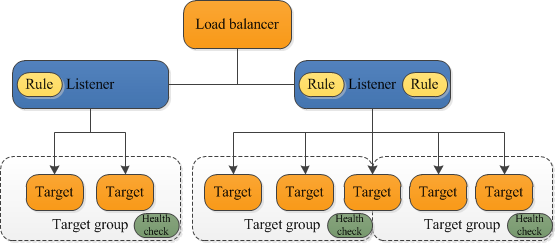
* Create a launch template with Amazon Linux 2 AMI and t2.micro instance type with Jenkins and Docker setup (You can use the Day 39 User data script for installing the required tools.
* Create 3 Instances using Launch Template, there must be an option that shows number of instances to be launched ,can you find it? :)
* You can go one step ahead and create an auto-scaling group, sounds tough?

Check [this](https://docs.aws.amazon.com/autoscaling/ec2/userguide/create-launch-template.html#create-launch-template-for-auto-scaling) out

Post your progress on Linkedin.

Happy Learning :)

# Day 41: Setting up an Application Load Balancer with AWS EC2 🚀 ☁



LB2

### Hi, I hope you had a great day yesterday learning about the launch template and instances in EC2. Today, we are going to dive into one of the most important concepts in EC2: Load Balancing.

## What is Load Balancing?

Load balancing is the distribution of workloads across multiple servers to ensure consistent and optimal resource utilization. It is an essential aspect of any large-scale and scalable computing system, as it helps you to improve the reliability and performance of your applications.

## Elastic Load Balancing:

**Elastic Load Balancing (ELB)** is a service provided by Amazon Web Services (AWS) that automatically distributes incoming traffic across multiple EC2 instances. ELB provides three types of load balancers:

Read more from [here](https://docs.aws.amazon.com/elasticloadbalancing/latest/userguide/what-is-load-balancing.html)

1. **Application Load Balancer (ALB)** - *operates at layer 7 of the OSI model and is ideal for applications that require advanced routing and microservices.*

* Read more from [here](https://docs.aws.amazon.com/elasticloadbalancing/latest/application/introduction.html)

1. **Network Load Balancer (NLB)** - *operates at layer 4 of the OSI model and is ideal for applications that require high throughput and low latency.*

* Read more from [here](https://docs.aws.amazon.com/elasticloadbalancing/latest/network/introduction.html)

1. **Classic Load Balancer (CLB)** - *operates at layer 4 of the OSI model and is ideal for applications that require basic load balancing features.*

* Read more [here](https://docs.aws.amazon.com/elasticloadbalancing/latest/classic/introduction.html)

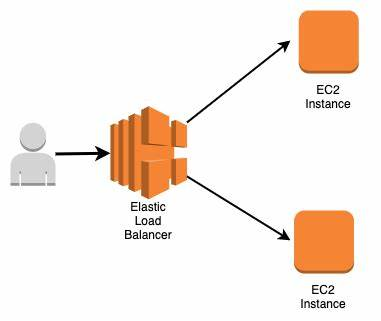
## 🎯 Today’s Tasks:

### Task 1:

* launch 2 EC2 instances with an Ubuntu AMI and use User Data to install the Apache Web Server.
* Modify the index.html file to include your name so that when your Apache server is hosted, it will display your name also do it for 2nd instance which include ” TrainWithShubham Community is Super Aweasome :) “.
* Copy the public IP address of your EC2 instances.
* Open a web browser and paste the public IP address into the address bar.
* You should see a webpage displaying information about your PHP installation.

### Task 2:

* Create an Application Load Balancer (ALB) in EC2 using the AWS Management Console.
* Add EC2 instances which you launch in task-1 to the ALB as target groups.
* Verify that the ALB is working properly by checking the health status of the target instances and testing the load balancing capabilities.



LoadBalancer

Need help with task? Check out this [Blog for assistance](https://rushikesh-mashidkar.hashnode.dev/create-an-application-load-balancer-elastic-load-balancing-using-aws-ec2-instance).

Don’t forget to share your progress on LinkedIn and have a great day🙌💥

Happy Learning! 😃

# Day 42: IAM Programmatic access and AWS CLI 🚀 ☁

Today is more of a reading excercise and getting some programmatic access for your AWS account

## IAM Programmatic access

In order to access your AWS account from a terminal or system, you can use AWS Access keys and AWS Secret Access keys Watch [this video](https://youtu.be/XYKqL5GFI-I) for more details.

## AWS CLI

The AWS Command Line Interface (AWS CLI) is a unified tool to manage your AWS services. With just one tool to download and configure, you can control multiple AWS services from the command line and automate them through scripts.

The AWS CLI v2 offers several new features including improved installers, new configuration options such as AWS IAM Identity Center (successor to AWS SSO), and various interactive features.

## Task-01

* Create AWS\_ACCESS\_KEY\_ID and AWS\_SECRET\_ACCESS\_KEY from AWS Console.

## Task-02

* Setup and install AWS CLI and configure your account credentials

Let me know if you have any issues while doing the task.

Happy Learning :)

# Day 43: S3 Programmatic access with AWS-CLI 💻 📁

Hi, I hope you had a great day yesterday. Today as part of the #90DaysofDevOps Challenge we will be exploring most commonly used service in AWS i.e S3.



s3

## S3

Amazon Simple Storage Service (Amazon S3) is an object storage service that provides a secure and scalable way to store and access data on the cloud. It is designed for storing any kind of data, such as text files, images, videos, backups, and more. Read more [here](https://docs.aws.amazon.com/AmazonS3/latest/userguide/Welcome.html) ## Task-01 - Launch an EC2 instance using the AWS Management Console and connect to it using Secure Shell (SSH). - Create an S3 bucket and upload a file to it using the AWS Management Console. - Access the file from the EC2 instance using the AWS Command Line Interface (AWS CLI).

Read more about S3 using aws-cli [here](https://docs.aws.amazon.com/cli/latest/reference/s3/index.html)

## Task-02

* Create a snapshot of the EC2 instance and use it to launch a new EC2 instance.
* Download a file from the S3 bucket using the AWS CLI.
* Verify that the contents of the file are the same on both EC2 instances.

Added Some Useful commands to complete the task. [Click here for commands](https://github.com/LondheShubham153/90DaysOfDevOps/blob/833a67ac4ec17b992934cd6878875dccc4274f56/2023/day43/aws-cli.md)

Let me know if you have any questions or face any issues while doing the tasks.🚀

Happy Learning :)

# Day 44: Relational Database Service in AWS

Amazon Relational Database Service (Amazon RDS) is a collection of managed services that makes it simple to set up, operate, and scale databases in the cloud

## Task-01

* Create a Free tier RDS instance of MySQL
* Create an EC2 instance
* Create an IAM role with RDS access
* Assign the role to EC2 so that your EC2 Instance can connect with RDS
* Once the RDS instance is up and running, get the credentials and connect your EC2 instance using a MySQL client.

Hint:

You should install mysql client on EC2, and connect the Host and Port of RDS with this client.

Post the screenshots once your EC2 instance can connect a MySQL server, that will be a small win for you.

Watch [this video](https://youtu.be/MrA6Rk1Y82E) for reference.

Happy Learning

# Day 45: Deploy Wordpress website on AWS

Over 30% of all websites on the internet use WordPress as their content management system (CMS). It is most often used to run blogs, but it can also be used to run e-commerce sites, message boards, and many other popular things. This guide will show you how to set up a WordPress blog site.

## Task-01

* As WordPress requires a MySQL database to store its data ,create an RDS as you did in Day 44

To configure this WordPress site, you will create the following resources in AWS: - An Amazon EC2 instance to install and host the WordPress application. - An Amazon RDS for MySQL database to store your WordPress data. - Setup the server and post your new Wordpress app.

Read [this](https://aws.amazon.com/getting-started/hands-on/deploy-wordpress-with-amazon-rds/) for a detailed explanation Happy Learning :)

# Day-46: Set up CloudWatch alarms and SNS topic in AWS

Hey learners, you have been using aws services atleast for last 45 days. Have you ever wondered what happen if for any service is charging you bill continously and you don’t know till you loose all your pocket money ?

Hahahaha😁, Well! we, as a responsible community ,always try to make it under free tier , but it’s good to know and setup something , which will inform you whenever bill touches a Threshold.

## What is Amazon CloudWatch?

Amazon CloudWatch monitors your Amazon Web Services (AWS) resources and the applications you run on AWS in real time. You can use CloudWatch to collect and track metrics, which are variables you can measure for your resources and applications.

Read more about cloudwatch from the official documentation [here](https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/WhatIsCloudWatch.html)

## What is Amazon SNS?

Amazon Simple Notification Service is a notification service provided as part of Amazon Web Services since 2010. It provides a low-cost infrastructure for mass delivery of messages, predominantly to mobile users.

Read more about it [here](https://docs.aws.amazon.com/sns/latest/dg/welcome.html)

## Task :

* Create a CloudWatch alarm that monitors your billing and send an email to you when a it reaches $2.

(You can keep it for your future use)

* Delete your billing Alarm that you created now.

(Now you also know how to delete as well. )

Need help with Cloudwatch? Check out this [official documentation](https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/monitor_estimated_charges_with_cloudwatch.html) for assistance.

Keep growing your AWS knowledge💥🙌

Happy Learning! :)

# Day 47: Test Knowledge on aws 💻 📈

Today, we will be test the aws knowledge on services in AWS, as part of the 90 Days of DevOps Challenge.

## Task-01

* Launch an EC2 instance using the AWS Management Console and connect to it using SSH.
* Install a web server on the EC2 instance and deploy a simple web application.
* Monitor the EC2 instance using Amazon CloudWatch and troubleshoot any issues that arise.

## Task-02

* Create an Auto Scaling group using the AWS Management Console and configure it to launch EC2 instances in response to changes in demand.
* Use Amazon CloudWatch to monitor the performance of the Auto Scaling group and the EC2 instances and troubleshoot any issues that arise.
* Use the AWS CLI to view the state of the Auto Scaling group and the EC2 instances and verify that the correct number of instances are running.

We hope that these tasks will give you hands-on experience with aws services and help you understand how these services work together. If you have any questions or face any issues while doing the tasks, please let us know.

Happy Learning :)

# Day-48 - ECS

Today will be a great learning for sure. I know many of you may not know about the term “ECS”. As you know, 90 Days Of DevOps Challange is mostly about ‘learning new’ , let’s learn then ;)

## What is ECS ?

* ECS (Elastic Container Service) is a fully-managed container orchestration service provided by Amazon Web Services (AWS). It allows you to run and manage Docker containers on a cluster of virtual machines (EC2 instances) without having to manage the underlying infrastructure.

With ECS, you can easily deploy, manage, and scale your containerized applications using the AWS Management Console, the AWS CLI, or the API. ECS supports both “Fargate” and “EC2 launch types”, which means you can run your containers on AWS-managed infrastructure or your own EC2 instances.

ECS also integrates with other AWS services, such as Elastic Load Balancing, Auto Scaling, and Amazon VPC, allowing you to build scalable and highly available applications. Additionally, ECS has support for Docker Compose and Kubernetes, making it easy to adopt existing container workflows.

Overall, ECS is a powerful and flexible container orchestration service that can help simplify the deployment and management of containerized applications in AWS.

## Difference between EKS and ECS ?

* EKS (Elastic Kubernetes Service) and ECS (Elastic Container Service) are both container orchestration platforms provided by Amazon Web Services (AWS). While both platforms allow you to run containerized applications in the AWS cloud, there are some differences between the two.

**Architecture**: ECS is based on a centralized architecture, where there is a control plane that manages the scheduling of containers on EC2 instances. On the other hand, EKS is based on a distributed architecture, where the Kubernetes control plane is distributed across multiple EC2 instances.

**Kubernetes Support**: EKS is a fully managed Kubernetes service, meaning that it supports Kubernetes natively and allows you to run your Kubernetes workloads on AWS without having to manage the Kubernetes control plane. ECS, on the other hand, has its own orchestration engine and does not support Kubernetes natively.

**Scaling**: EKS is designed to automatically scale your Kubernetes cluster based on demand, whereas ECS requires you to configure scaling policies for your tasks and services.

**Flexibility**: EKS provides more flexibility than ECS in terms of container orchestration, as it allows you to customize and configure Kubernetes to meet your specific requirements. ECS is more restrictive in terms of the options available for container orchestration.

**Community**: Kubernetes has a large and active open-source community, which means that EKS benefits from a wide range of community-driven development and support. ECS, on the other hand, has a smaller community and is largely driven by AWS itself.

In summary, EKS is a good choice if you want to use Kubernetes to manage your containerized workloads on AWS, while ECS is a good choice if you want a simpler, more managed platform for running your containerized applications.

# Task :

Set up ECS (Elastic Container Service) by setting up Nginx on ECS.

# Day 49 - INTERVIEW QUESTIONS ON AWS

Hey people, we have listened to your suggestions and we are looking forward to get more! As you people have asked to put more interview based questions as part of Daily Task, So here it it :)

## INTERVIEW QUESTIONS:

* Name 5 aws services you have used and what’s the use cases?
* What are the tools used to send logs to the cloud environment?
* What are IAM Roles? How do you create /manage them?
* How to upgrade or downgrade a system with zero downtime?
* What is infrastructure as code and how do you use it?
* What is a load balancer? Give scenarios of each kind of balancer based on your experience.
* What is CloudFormation and why is it used for?
* Difference between AWS CloudFormation and AWS Elastic Beanstalk?
* What are the kinds of security attacks that can occur on the cloud? And how can we minimize them?
* Can we recover the EC2 instance when we have lost the key?
* What is a gateway?
* What is the difference between the Amazon Rds, Dynamodb, and Redshift?
* Do you prefer to host a website on S3? What’s the reason if your answer is either yes or no?

Let’s share your answer on LinkedIn in best possible way thinking you are in a interview table. Happy Learning !! :)

# Day 50: Your CI/CD pipeline on AWS - Part-1 🚀 ☁

What if I tell you, in next 4 days, you’ll be making a CI/CD pipeline on AWS with these tools.

* CodeCommit
* CodeBuild
* CodeDeploy
* CodePipeline
* S3

## What is CodeCommit ?

* CodeCommit is a managed source control service by AWS that allows users to store, manage, and version their source code and artifacts securely and at scale. It supports Git, integrates with other AWS services, enables collaboration through branch and merge workflows, and provides audit logs and compliance reports to meet regulatory requirements and track changes. Overall, CodeCommit provides developers with a reliable and efficient way to manage their codebase and set up a CI/CD pipeline for their software development projects.

# Task-01 :

* Set up a code repository on CodeCommit and clone it on your local.
* You need to setup GitCredentials in your AWS IAM.
* Use those credentials in your local and then clone the repository from CodeCommit

# Task-02 :

* Add a new file from local and commit to your local branch
* Push the local changes to CodeCommit repository.

For more details watch [this](https://youtu.be/p5i3cMCQ760) video.

Happy Learning :)

# Day 51: Your CI/CD pipeline on AWS - Part 2 🚀 ☁

On your journey of making a CI/CD pipeline on AWS with these tools, you completed AWS CodeCommit.

Next few days you’ll learn these tools/services:

* CodeBuild
* CodeDeploy
* CodePipeline
* S3

## What is CodeBuild ?

* AWS CodeBuild is a fully managed build service in the cloud. CodeBuild compiles your source code, runs unit tests, and produces artifacts that are ready to deploy. CodeBuild eliminates the need to provision, manage, and scale your own build servers.

# Task-01 :

* Read about Buildspec file for Codebuild.
* create a simple index.html file in CodeCommit Repository
* you have to build the index.html using nginx server

# Task-02 :

* Add buildspec.yaml file to CodeCommit Repository and complete the build process.

For more details watch [this](https://youtu.be/p5i3cMCQ760) video.

Happy Learning :)

# Day 52: Your CI/CD pipeline on AWS - Part 3 🚀 ☁

On your journey of making a CI/CD pipeline on AWS with these tools, you completed AWS CodeCommit & CodeBuild.

Next few days you’ll learn these tools/services:

* CodeDeploy
* CodePipeline
* S3

## What is CodeDeploy ?

* AWS CodeDeploy is a deployment service that automates application deployments to Amazon EC2 instances, on-premises instances, serverless Lambda functions, or Amazon ECS services.

CodeDeploy can deploy application content that runs on a server and is stored in Amazon S3 buckets, GitHub repositories, or Bitbucket repositories. CodeDeploy can also deploy a serverless Lambda function. You do not need to make changes to your existing code before you can use CodeDeploy.

# Task-01 :

* Read about Appspec.yaml file for CodeDeploy.
* Deploy index.html file on EC2 machine using nginx
* you have to setup a CodeDeploy agent in order to deploy code on EC2

# Task-02 :

* Add appspec.yaml file to CodeCommit Repository and complete the deployment process.

For more details watch [this](https://youtu.be/IUF-pfbYGvg) video.

Happy Learning :)

# Day 53: Your CI/CD pipeline on AWS - Part 4 🚀 ☁

On your journey of making a CI/CD pipeline on AWS with these tools, you completed AWS CodeCommit, CodeBuild & CodeDeploy.

Finish Off in style with AWS CodePipeline

## What is CodePipeline ?

* CodePipeline builds, tests, and deploys your code every time there is a code change, based on the release process models you define. Think of it as a CI/CD Pipeline service

# Task-01 :

* Create a Deployment group of Ec2 Instance.
* Create a CodePipeline that gets the code from CodeCommit, Builds the code using CodeBuild and deploys it to a Deployment Group.

For more details watch [this](https://youtu.be/IUF-pfbYGvg) video.

Happy Learning :)

# Day 54: Understanding Infrastructure as Code and Configuration Management

## What’s the difference bhaiyya?

When it comes to the cloud, Infrastructure as Code (IaC) and Configuration Management (CM) are inseparable. With IaC, a descriptive model is used for infrastructure management. To name a few examples of infrastructure: networks, virtual computers, and load balancers. Using an IaC model always results in the same setting.

Throughout the lifecycle of a product, Configuration Management (CM) ensures that the performance, functional and physical inputs, requirements, design, and operations of that product remain consistent.

# Task-01

* Read more about IaC and Config. Management Tools
* Give differences on both with suitable examples
* What are most commont IaC and Config management Tools?

Write a blog on this topic in the most creative way and post it on linkedIn :)

happy learning…

# Day 55: Understanding Configuration Management with Ansible

## What’s this Ansible?

Ansible is an open-source automation tool, or platform, used for IT tasks such as configuration management, application deployment, intraservice orchestration, and provisioning

## Task-01

* Installation of Ansible on AWS EC2 (Master Node) sudo apt-add-repository ppa:ansible/ansible sudo apt update sudo apt install ansible

## Task-02

* read more about Hosts file sudo nano /etc/ansible/hosts ansible-inventory --list -y

## Task-03

* Setup 2 more EC2 instances with same Private keys as the previous instance (Node)
* Copy the private key to master server where Ansible is setup
* Try a ping command using ansible to the Nodes.

Write a blog on this topic with screenshots in the most creative way and post it on linkedIn :)

happy learning…

# Day 56: Understanding Ad-hoc commands in Ansible

Ansible ad hoc commands are one-liners designed to achieve a very specific task they are like quick snippets and your compact swiss army knife when you want to do a quick task across multiple machines.

To put simply, Ansible ad hoc commands are one-liner Linux shell commands and playbooks are like a shell script, a collective of many commands with logic.

Ansible ad hoc commands come handy when you want to perform a quick task.

## Task-01

* write an ansible ad hoc ping command to ping 3 servers from inventory file
* Write an ansible ad hoc command to check uptime
* You can refer to [this](https://www.middlewareinventory.com/blog/ansible-ad-hoc-commands/) blog to understand the different examples of ad-hoc commands and try out them, post the screenshots in a blog with an explanation.

happy Learning :)

# Day 57: Ansible Hands-on with video

Ansible is fun, you saw in last few days how easy it is.

Let’s make it fun now, by using a video explanation for Ansible.

## Task-01

* Write a Blog explanation for the [ansible video](https://youtu.be/SGB7EdiP39E)

happy Learning :)

# Day 58: Ansible Playbooks

Ansible playbooks run multiple tasks, assign roles, and define configurations, deployment steps, and variables. If you’re using multiple servers, Ansible playbooks organize the steps between the assembled machines or servers and get them organized and running in the way the users need them to. Consider playbooks as the equivalent of instruction manuals.

## Task-01

* Write an ansible playbook to create a file on a different server
* Write an ansible playbook to create a new user.
* Write an ansible playbook to install docker on a group of servers

Watch [this](https://youtu.be/089mRKoJTzo) video to learn about ansible Playbooks

## Task-02

* Write a blog about writing ansible playbooks with the best practices.

Let me or anyone in the community know if you face any challenges

happy Learning :)

# Day 59: Ansible Project 🔥

Ansible playbooks are amazing, as you learned yesterday. What if you deploy a simple web app using ansible, sounds like a good project, right?

## Task-01

* create 3 EC2 instances . make sure all three are created with same key pair
* Install Ansible in host server
* copy the private key from local to Host server (Ansible\_host) at (/home/ubuntu/.ssh)
* access the inventory file using sudo vim /etc/ansible/hosts
* Create a playbook to install Nginx
* deploy a sample webpage using the ansible playbook

Read [this](https://medium.com/@sandeep010498/learn-ansible-with-real-time-project-cf6a0a512d45) Blog by [Sandeep Singh](https://medium.com/@sandeep010498) to clear all your doubts

Let me or anyone in the community know if you face any challenges

happy Learning :)

# Day 60 - Terraform🔥

Hello Learners , you guys are doing every task by creating an ec2 instance (mostly). Today let’s automate this process . How to do it ? Well Terraform is the solution . ## What is Terraform? Terraform is an infrastructure as code (IaC) tool that allows you to create, manage, and update infrastructure resources such as virtual machines, networks, and storage in a repeatable, scalable, and automated way.

## Task 1:

Install Terraform on your system Refer this [link](https://phoenixnap.com/kb/how-to-install-terraform) for installation

## Task 2: Answer below questions

* Why we use terraform?
* What is Infrastructure as Code (IaC)?
* What is Resource?
* What is Provider?
* Whats is State file in terraform? What’s the importance of it ?
* What is Desired and Current State?

You can prepare for tomorrow’s task from [here](https://www.youtube.com/live/965CaSveIEI?feature=share)🚀🚀

We Hope this tasks will help you understand how to write a basic Terraform configuration file and basic commands on Terraform.

Don’t forget to post in on LinkedIn. Happy Learning:)

# Day 61- Terraform🔥

Hope you’ve already got the gist of What Working with Terraform would be like . Lets begin with day 2 of Terraform !

## Task 1:

find purpose of basic Terraform commands which you’ll use often

1. terraform init
2. terraform init -upgrade
3. terraform plan
4. terraform apply
5. terraform validate
6. terraform fmt
7. terraform destroy

Also along with these tasks its important to know about Terraform in general- Who are Terraform’s main competitors? The main competitors are:

Ansible Packer Cloud Foundry Kubernetes

Want a Free video Course for terraform? Click [here](https://bit.ly/tws-terraform)

Don’t forget to share your learnings on Linkedin ! Happy Learning :)

# Day 62 - Terraform and Docker 🔥

Terraform needs to be told which provider to be used in the automation, hence we need to give the provider name with source and version. For Docker, we can use this block of code in your main.tf

## Blocks and Resources in Terraform

## Terraform block

## Task-01

* Create a Terraform script with Blocks and Resources

terraform {  
 required\_providers {  
 docker = {  
 source = "kreuzwerker/docker"  
 version = "~> 2.21.0"  
}  
}  
}

### Note: kreuzwerker/docker, is shorthand for registry.terraform.io/kreuzwerker/docker.

## Provider Block

The provider block configures the specified provider, in this case, docker. A provider is a plugin that Terraform uses to create and manage your resources.

provider "docker" {}

## Resource

Use resource blocks to define components of your infrastructure. A resource might be a physical or virtual component such as a Docker container, or it can be a logical resource such as a Heroku application.

Resource blocks have two strings before the block: the resource type and the resource name. In this example, the first resource type is docker\_image and the name is Nginx.

## Task-02

* Create a resource Block for an nginx docker image

Hint:

resource "docker\_image" "nginx" {  
 name = "nginx:latest"  
 keep\_locally = false  
}

* Create a resource Block for running a docker container for nginx

resource "docker\_container" "nginx" {  
 image = docker\_image.nginx.latest  
 name = "tutorial"  
 ports {  
 internal = 80  
 external = 80  
 }  
}

Note: In case Docker is not installed

sudo apt-get install docker.io sudo docker ps sudo chown $USER /var/run/docker.sock

# Video Course

I can imagine, Terraform can be tricky, so best to use a Free video Course for terraform [here](https://bit.ly/tws-terraform)

Happy Learning :)

# Day 63 - Terraform Variables

variables in Terraform are quite important, as you need to hold values of names of instance, configs , etc.

We can create a variables.tf file which will hold all the variables.

variable "filename" {  
default = "/home/ubuntu/terrform-tutorials/terraform-variables/demo-var.txt"  
}

variable "content" {  
default = "This is coming from a variable which was updated"  
}

These variables can be accessed by var object in main.tf

## Task-01

* Create a local file using Terraform Hint:

resource "local\_file" "devops" {  
filename = var.filename  
content = var.content  
}

## Data Types in Terraform

## Map

variable "file\_contents" {  
type = map  
default = {  
"statement1" = "this is cool"  
"statement2" = "this is cooler"  
}  
}

## Task-02

* Use terraform to demonstrate usage of List, Set and Object datatypes
* Put proper screenshots of the outputs

Use terraform refresh

To refresh the state by your configuration file, reloads the variables

# Video Course

I can imagine, Terraform can be tricky, so best to use a Free video Course for terraform [here](https://bit.ly/tws-terraform)

Happy Learning :)

# Day 64 - Terraform with AWS

Provisioning on AWS is quite easy and straightforward with Terraform.

## Prerequisites

### AWS CLI installed

The AWS Command Line Interface (AWS CLI) is a unified tool to manage your AWS services. With just one tool to download and configure, you can control multiple AWS services from the command line and automate them through scripts.

### AWS IAM user

IAM (Identity Access Management) AWS Identity and Access Management (IAM) is a web service that helps you securely control access to AWS resources. You use IAM to control who is authenticated (signed in) and authorized (has permissions) to use resources.

*In order to connect your AWS account and Terraform, you need the access keys and secret access keys exported to your machine.*

export AWS\_ACCESS\_KEY\_ID=<access key>  
export AWS\_SECRET\_ACCESS\_KEY=<secret access key>

### Install required providers

terraform {  
 required\_providers {  
 aws = {  
 source = "hashicorp/aws"  
 version = "~> 4.16"  
}  
}  
 required\_version = ">= 1.2.0"  
}

Add the region where you want your instances to be

provider "aws" {  
region = "us-east-1"  
}

## Task-01

* Provision an AWS EC2 instance using Terraform

Hint:

resource "aws\_instance" "aws\_ec2\_test" {  
 count = 4  
 ami = "ami-08c40ec9ead489470"  
 instance\_type = "t2.micro"  
 tags = {  
 Name = "TerraformTestServerInstance"  
 }  
}

# Video Course

I can imagine, Terraform can be tricky, so best to use a Free video Course for terraform [here](https://bit.ly/tws-terraform)

Happy Learning :)

# Day 65 - Working with Terraform Resources 🚀

Yesterday, we saw how to create a Terraform script with Blocks and Resources. Today, we will dive deeper into Terraform resources.

## Understanding Terraform Resources

A resource in Terraform represents a component of your infrastructure, such as a physical server, a virtual machine, a DNS record, or an S3 bucket. Resources have attributes that define their properties and behaviors, such as the size and location of a virtual machine or the domain name of a DNS record.

When you define a resource in Terraform, you specify the type of resource, a unique name for the resource, and the attributes that define the resource. Terraform uses the resource block to define resources in your Terraform configuration.

## Task 1: Create a security group

To allow traffic to the EC2 instance, you need to create a security group. Follow these steps:

In your main.tf file, add the following code to create a security group:

resource "aws\_security\_group" "web\_server" {  
 name\_prefix = "web-server-sg"  
  
 ingress {  
 from\_port = 80  
 to\_port = 80  
 protocol = "tcp"  
 cidr\_blocks = ["0.0.0.0/0"]  
 }  
}

* Run terraform init to initialize the Terraform project.
* Run terraform apply to create the security group.

## Task 2: Create an EC2 instance

* Now you can create an EC2 instance with Terraform. Follow these steps:
* In your main.tf file, add the following code to create an EC2 instance:

resource "aws\_instance" "web\_server" {  
 ami = "ami-0557a15b87f6559cf"  
 instance\_type = "t2.micro"  
 key\_name = "my-key-pair"  
 security\_groups = [  
 aws\_security\_group.web\_server.name  
 ]  
  
 user\_data = <<-EOF  
 #!/bin/bash  
 echo "<html><body><h1>Welcome to my website!</h1></body></html>" > index.html  
 nohup python -m SimpleHTTPServer 80 &  
 EOF  
}

Note: Replace the ami and key\_name values with your own. You can find a list of available AMIs in the AWS documentation.

Run terraform apply to create the EC2 instance.

## Task 3: Access your website

* Now that your EC2 instance is up and running, you can access the website you just hosted on it. Follow these steps:

Happy Terraforming!