

DEVOPS ROADMAP



TABLE OF CONTENTS

I. START HERE 3

II. DEVOPS ROADMAP

01 - Concepts of Software Development	4
02 - OS and Linux Basics	5
03 - Containerization - Docker	6
04 - CI/CD Pipelines	7
05 - Learn 1 Cloud Provider	8
06- Container Orchestration - Kubernetes	9
07- Monitoring and Observability	10
08 - Infrastructure as Code	11
09 - Scripting Language	12
10 - Version Control - Git	13

III. STARTING FROM...

Systems Administrator	15
Software Developer	16
Test Automation Engineer	17
Network Engineer	18
No or Little IT Background	19

IV. RECAP AND RESOURCES

Summary - DevOps Roadmap	21
--------------------------	-----------

Start Here

This is a **step by step path I would take as a DevOps professional and educator**, if I was starting from zero again. Showing you what would be the most efficient path to become a DevOps engineer based on the knowledge that I have now.

I hope with this I can help you in this highly rewarding, but challenging journey into DevOps.

And I wanted to make it **more individual** for you based on your background from which you are transitioning into DevOps:

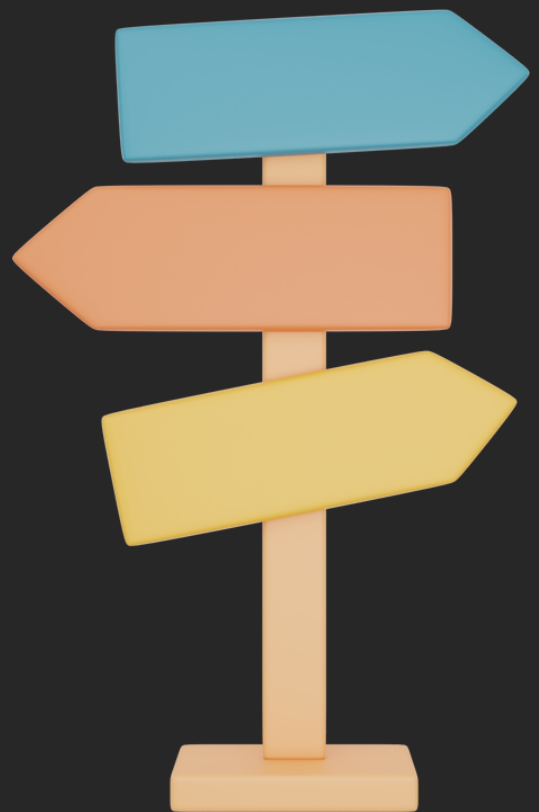
- system administrator
- software developer
- test automation engineer
- network engineer
- someone with zero or very little IT knowledge

So after the DevOps roadmap, you will find information on how to start in DevOps having any of these backgrounds.

One important thing beforehand:

Since DevOps covers the whole software development lifecycle, it means you work with lots of technologies. Plus DevOps is still evolving and there are lots of new tools being developed all the time.

So you have to be **comfortable with constantly learning** and expanding your knowledge, even after you have become a DevOps engineer.



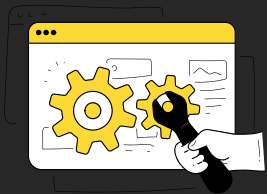
Concepts of Software Development

As a DevOps engineer you will not be programming the application, but as you are working closely with the development team to improve and automate tasks for them, you need to **understand the concepts** of:



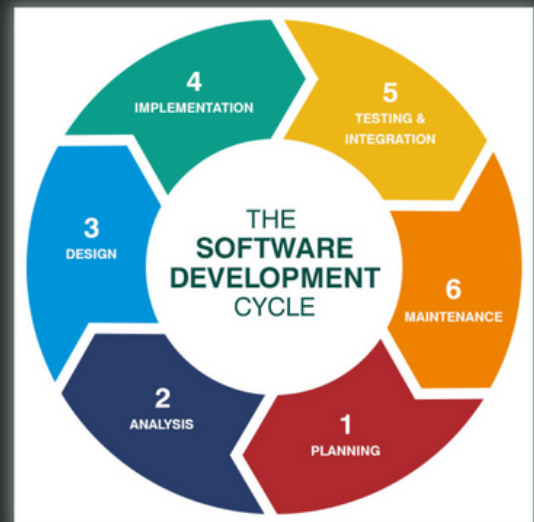
How developers work and collaborate
(Agile, Jira workflows)

What Git workflow they use



How applications are configured (Build & packaging Tools)

Automated testing and test scopes



And generally understand what the whole **software development lifecycle** covers from idea to code, all the way to releasing it to the end users!

OS & Linux Basics

As a DevOps engineer you are responsible for preparing and maintaining the infrastructure (servers) on which the application is deployed.

So you need to know the basics of how to administer a server and install different tools on it.

Basic concepts of Operating Systems you need to understand:



Shell Commands



Linux File System & Permissions



SSH Key Management



Virtualization



Since most servers use Linux OS, you need to know and feel comfortable using **Linux**, especially its Command Line Interface.

You also need to know the **basics of Networking & Security** in order to configure the infrastructure, like:

- Configure Firewalls to secure access
- Understand how IP addresses, ports and DNS works
- Load Balancers
- Proxies
- HTTP/HTTPS



However, to draw a line here between DevOps and IT Operations: You don't need to be the SysAdmin. So **no advanced knowledge** of server administration is needed here. It's enough to know the basics. There are own professions like SysAdmins, Networking or Security Professionals for more advanced use cases.

Containerization - Docker

As containers have become the new standard of software packaging, you will most probably run your application as a container.

This means you need to generally understand:

- concepts of virtualization
- concepts of containerization
- how to manage containerized applications on a server.



A container is a standard unit of software that packages up code and all its dependencies

so the application runs quickly and reliably on any computing environment.



Docker is by far the most popular container technology!

Some things you should know:

- Run containers
- Inspect active containers
- Docker Networking
- Persist data with Docker Volumes
- Dockerize apps using Dockerfiles
- Run multiple containers using Docker-Compose
- Work with Docker Repository



Containers and virtual machines have similar resource isolation and allocation benefits, but function differently. **VMs virtualize the whole OS. Containers virtualize only the application level of the OS.** Therefore, containers are more lightweight and faster.

CI/CD Pipelines

CI/CD is kind of the heart of DevOps.

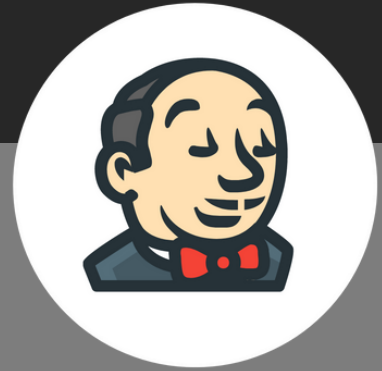
In DevOps, all code changes, like new features or bug fixes, need to be **integrated** in the existing application and **deployed** for the end user **continuously** and in an **automated** way.

Hence the term:

Continuous Integration and Continuous Deployment (CI/CD)

When the feature or bugfix is done, a pipeline running on a CI server (e.g. Jenkins) should be triggered automatically, which:

1. runs the tests
2. packages the application
3. builds a container Image
4. pushes the container Image to an image repository
5. deploys the new version to a server

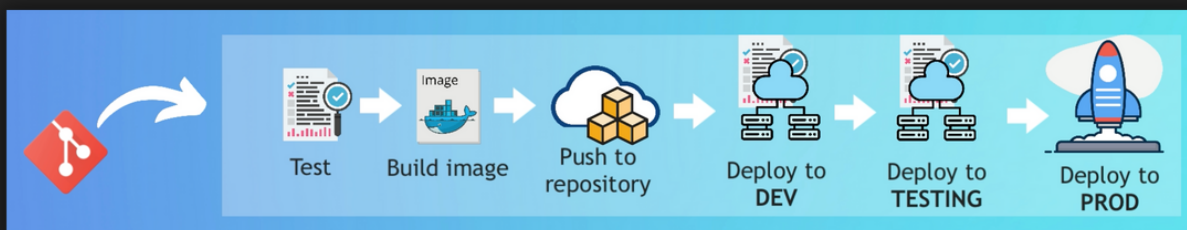


There are many CI/CD platforms out there. The most popular one currently is Jenkins

Other popular ones: GitLab, GitHub Actions, Travis CI, Bamboo

Skills you need to learn here:

- Setting up the CI/CD server
- Integrate code repository to trigger pipeline automatically
- Build Tools & Package Manager Tools to execute the tests and package the application
- Configuring artifact repositories (like Nexus) and integrate with pipeline



Learn one Cloud Provider

Nowadays many companies use **virtual infrastructure on the cloud**, instead of managing their own infrastructure. These are Infrastructure as a Service (IaaS) platforms, which offer a range of additional services, like backup, security, load balancing etc.



These services are **platform-specific**. So you need to learn the services of that specific platform and learn how to manage the whole deployment infrastructure on it.

E.g. for AWS you should know the fundamentals of:

- IAM service - managing users and permissions
- VPC service - your private network
- EC2 service - virtual servers



AWS is the most powerful and most widely used IaaS platform, but also a difficult one.

Other popular ones: Microsoft Azure, Google Cloud

AWS has loads of services, but you only need to learn the services you/your company actually needs. E.g. when the K8s cluster runs on AWS you need to learn the EKS service as well.

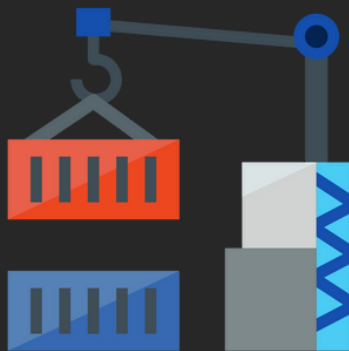


Once you learn one IaaS platform, it's easy to learn others

Container Orchestration - Kubernetes

Since containers are popular and easy to use, many companies are running hundreds or thousands of containers on multiple servers. This means these containers need to be managed somehow.

For this purpose there are container orchestration tools.



Container orchestration tools like Kubernetes, **automate** the deployment, scaling and management of containerized applications.



Kubernetes (also known as K8s) is the most popular container orchestration tool

So you need to learn:

- How Kubernetes works
- How to administer and manage the K8s cluster
- How to deploy applications on K8s

Specific K8s knowledge needed:

- Learn core components like, Deployment, Service, ConfigMap, Secret, StatefulSet, Ingress
- Kubernetes CLI (Kubectl)
- Persisting data with K8s Volumes
- Namespaces

Monitoring & Observability

Once software is in production, it is important to monitor it to **track the performance, discover problems** in your infrastructure and the application.

So one of your responsibilities as a DevOps engineer is to:

- setup software monitoring
- setup infrastructure monitoring, e.g. for your Kubernetes cluster and underlying servers
- visualize the data



Prometheus:

A popular monitoring and alerting tool



Grafana:

Analytics and interactive visualization tool

You should also understand how systems can collect and aggregate data with the goal of using it to troubleshoot, gain business insights etc.



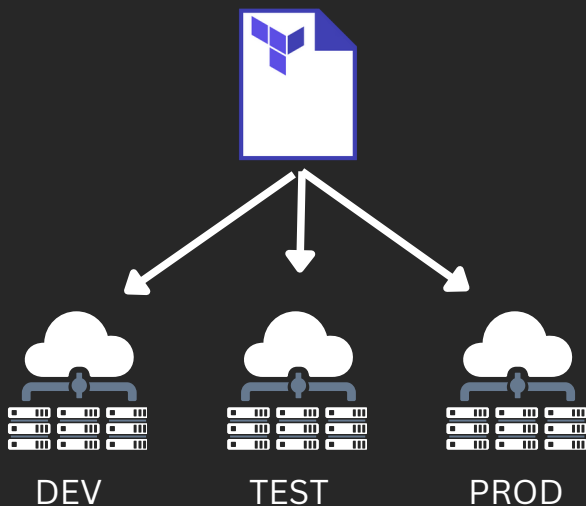
ELK Stack:

A popular log management stack

Infrastructure as Code

Manually creating and maintaining infrastructure is time consuming and error prone. Especially when you need to **replicate the infrastructure**, e.g. for a Development, Testing and Production environment.

In DevOps, we want to automate as much as possible and that's where Infrastructure as Code comes into the picture.



With IaC we **use code to create and configure infrastructure** and there are 2 types of IaC tools you need to know:

1. Infrastructure provisioning
2. Configuration management



Terraform is the most popular infrastructure provisioning tool

Ansible is the most popular configuration management tool



Benefits of having everything as code :

- ✓ Encourage collaboration in a team
- ✓ Document changes to infrastructure
- ✓ Transparency of the infrastructure state
- ✓ Accessibility to that information in a centralized place versus being scattered on people's local machines in the form of some scripts.

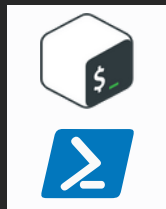
Scripting Language

Since you are closely working with developers and system administrators to also automate tasks for development and operations, you will need to write scripts and small applications to automate them.

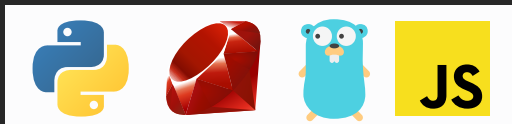
For that, you will need some scripting or basic programming skills.

Examples: utility scripts like flushing the cache, starting the builds and deployments etc.

This could be an **OS-specific scripting language** like bash or Powershell.



But what's more demanded is an **OS-independent** language like Python, Ruby or Go.



These languages are more powerful and flexible. If you know one of these, it will make you much more valuable as a DevOps engineer.



Python is one of the most popular programming languages and easy to learn

There are many programming languages, but I would recommend starting with Python. Python is widely used, easy to learn and used for many different use cases, especially in DevOps.

You don't need the same level as a software developer. Learning how to write scripts with Python will be enough.



And the good thing is, programming concepts stay the same, so when you learn one language well, you can easily learn new ones quite quickly.

Version Control - Git

You write all automation logic as code. And just application code, automation code should also be **managed and hosted** on a **version control tool**, like Git.



K8s config files



Python scripts



Infrastructure as Code

Git is a **CLI Tool**, which you install locally. It enables the tracking of changes in the source code and enables better collaboration on code.

So you need to learn:

- The core Git commands, like git clone, git branch, git pull/push, git merge etc
- But also how to collaborate on a project, like create pull requests, code reviews, branching



You need to learn Git. It's the most popular and widely used version control tool

Your files are stored centrally in a remote Git repository on the web. Most popular Git repositories are GitHub and GitLab.

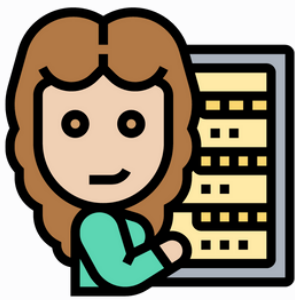


Do NOT store secrets and passwords in your Git repository

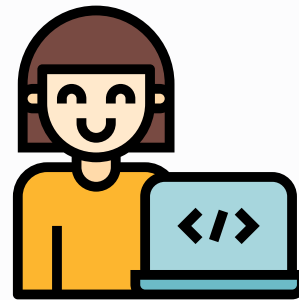
Starting from..

Having those DevOps skills is the final goal, but many of you start your DevOps journey having **various different backgrounds**.

So the starting point is different for all of you. You may be starting as a systems administrator or software engineer or test automation engineer etc or may not have an IT background at all and want to transition into DevOps:



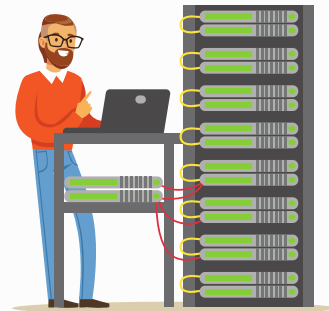
Systems Administrator



Software Developer

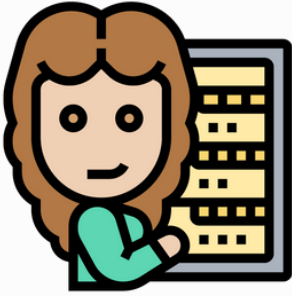


Test Automation Engineer



Network Engineer

Starting as a... Systems Administrator



Systems Administrator

You know how to administer servers and other systems. So you already have some skills in:

- setting up infrastructure
- configuring and preparing it for deployment
- working with operating systems, installing and running software
- security, networking configuration

Some other tasks you might do are things like:

- monitoring systems,
- health, backup and disaster recovery,
- database administration,
- network administration or
- security administration



So you already have a lot of **skills you can use in the deployment and operations side of DevOps.**



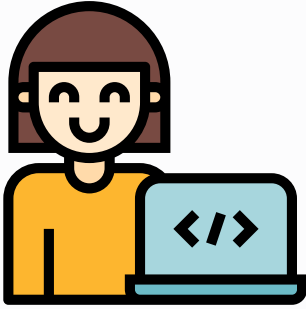
**START
HERE**

The big part **missing** here to start in DevOps is learning the **software development basics**:

- Understanding the Git workflows
- How developers work
- How to create a CI/CD pipeline



Starting as a... Software Developer



Software Developer

If you are a software developer, you have a pretty good background, because you already know important parts of DevOps, which are software development workflows and release pipelines.

Your programming skills will also be great help in writing automated scripts for various parts of the application development and deployment processes.



**START
HERE**

But you are **missing skills in server management**.

So you need to start by learning about:

- Linux, OS basics and virtual machines
- creating and configuring servers
- configuring infrastructure security, networking etc.



And since most modern applications run on cloud, you need to learn how to do all these on a cloud infrastructure.

So that would be your starting point when learning DevOps as a software developer.

And once you have that foundation you can build on that by learning about how containers work on top of the virtual machines and how to run applications in containers and how to run containers on a platform like Kubernetes etc.

Starting as a... Test Automation Engineer



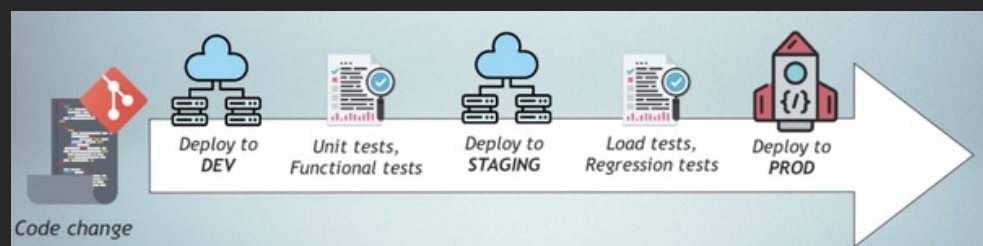
Test Automation Engineer

Another common background people have when transitioning into DevOps is a test automation engineering.

Here you may have a **bit more catching up** to do and more skills to learn compared to developers or systems administrators, but you can definitely reuse many of your skills in DevOps.

You most probably know how the developers work, like the agile processes, Jira workflows and so on. And as part of your test automation knowledge you understand the different testing scopes.

You also understand **how to test different aspects of an application** and that knowledge is really helpful for setting up an automated CI/CD pipeline, because in order to automate and streamline delivering your application changes all the way to the production environment, you need extensive automated testing:



**START
HERE**

Like developers, you are **missing skills in server management**. So you need to start by learning about:

- virtual machines and Linux basics
- creating and configuring servers
- configuring infrastructure security, networking etc.

Starting as a... Network Engineer



Network Engineer

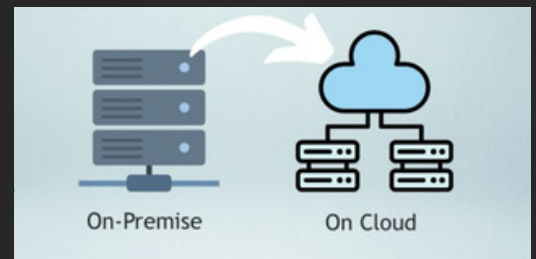
Another common background people have when transitioning into DevOps is network engineering. This is probably the farthest from DevOps compared to the other three, but you still have some skills that you can bring into DevOps as a network engineer.



As a network engineer you know how to configure devices and networking between devices. So you have valuable knowledge in configuring networking for infrastructure on premise.

Transition to Cloud Network Engineer

But as most companies are moving their infrastructure to cloud, many network engineers transition to cloud network engineering, configuring **virtual** routes, switches etc.



With this knowledge you have an advantage to understand networking in containers and Kubernetes, which is how most modern applications are running. Networking in containers and K8s is pretty complex, especially when we need to secure and troubleshoot those networks.

Some network engineers even know scripting in bash or python for example, which is another helpful skill when it comes to automation part of DevOps.

**START
HERE**

So a good starting point for network engineers is to move to cloud engineering first and then move on to containers and Kubernetes

Starting with...

No or little IT Background



Non IT Background

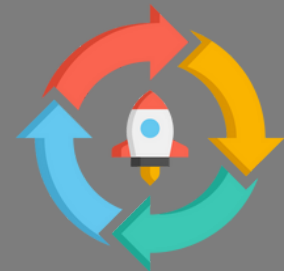
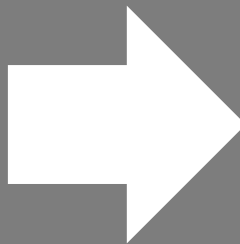
Some people want to get into DevOps having **very little to no IT background**. This means there are probably some of you reading this, thinking about getting into DevOps without much IT pre-knowledge and want to know what the path is to DevOps.

Now this is a very tricky one, because **DevOps is NOT an entry-level profession** in IT. It's not the first thing you learn when you want to get into the IT field.

Now why is that?



DevOps is about **automating processes** in software development and deployment lifecycle that **people have done manually** before.



This means, before you automate processes and tasks that are done manually, you **first need to understand what those processes and tasks are in the first place**. If you don't understand those, you won't know what you're automating or why you even need DevOps.

Starting with...

No or little IT Background

**START
HERE**

1 - Understand complete software development lifecycle



Go find some example projects, where you create a super simple web application and learn how to deploy it to a virtual server. In this process you will learn the steps of developing, packaging, maybe automatically testing and then deploying an example web application on a Linux server on a cloud platform.

This will **teach you the basics of the complete software development lifecycle**, but most importantly it will make you understand each step in the complete workflow and what goes into that.

2 - How software development teams collaborate

After that, go ahead and watch some tutorials about agile and scrum methods and how software development teams collaborate and work in IT projects.



3 - DevOps Pre-Requisites and 4 - DevOps Skills

These skills will actually be enough to start learning DevOps with our DevOps Bootcamp for example, because in our bootcamp, you actually learn Linux, Git and all these basic tools from scratch.



But again you need to understand those workflows first in order to understand, why we're using Git, why we need Jenkins, why we're learning Linux and scripting, we we use containers etc.

Summary

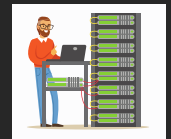
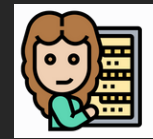
DevOps Roadmap



1 - Getting the prerequisites right

First step is to get the DevOps prerequisites right. So depending on which background and pre-knowledge you have, you need to first make sure to get any missing prerequisite knowledge.

So as a system administrator or a network engineer, learn the software development workflows. As a developer, learn the basics of infrastructure, virtual servers etc.



Of course with zero IT background, you have to get all this prerequisite knowledge from server administration to development first. So you have a more difficult entry, but it is possible if you know what to learn.



2 - Cloud, Docker, K8s



After learning the prerequisites, you can already get started with important DevOps skills of working with containers and container orchestration tools. So basically learning Docker and Kubernetes to help your teams deploy and efficiently run the application.

And since most of the modern applications and Kubernetes clusters are running on cloud, you need to learn cloud infrastructure, how to work with cloud infrastructure, how to configure it, how to scale it and so on.

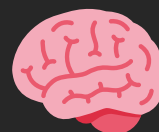
3 - Automation



As a DevOps professional automation skills are one of the most important ones. And as the heart of DevOps, learning to build CI/CD pipelines is an essential skill.

Finally you will learn how to automate parts of the complete DevOps processes one by one using the concepts and tools of what's called X as code: IaC, Configuration as Code, Security as Code, Policy as Code and so on, which basically means just automating everything in the form of code!

4 - Go from there. Keep learning



DevOps is evolving and new tools are being developed all the time. So as a DevOps professional, you should learn how to evaluate and test many new tools, always with the same goal to optimize and automate existing processes and make them more efficient.