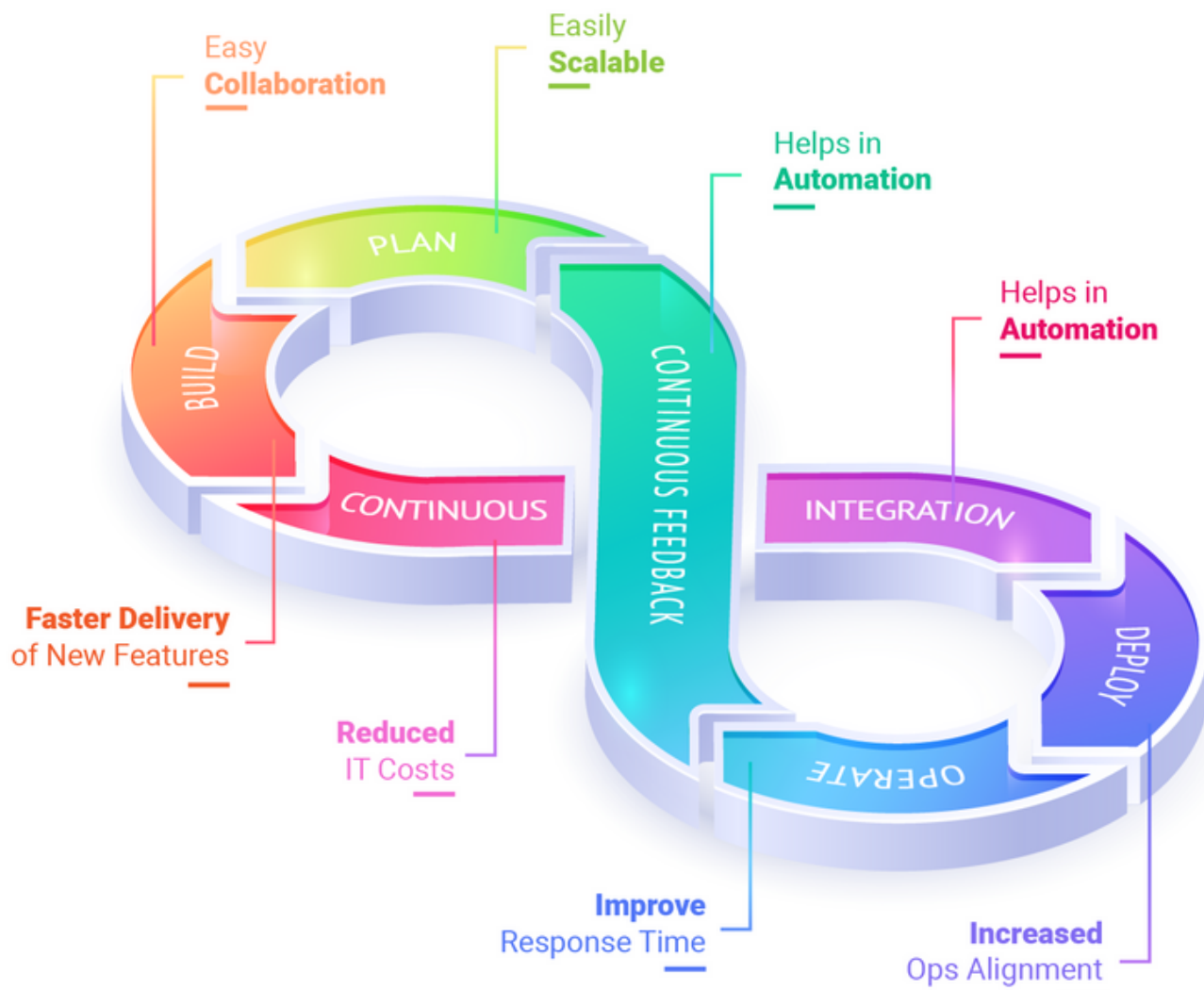


MASTERING DEVOPS WITH EDUREKA

# A Beginners Guide to DevOps Basics

Don't just Learn it, Master it!



# TABLE OF CONTENTS

---

<b>1. INTRODUCTION TO DEVOPS</b>	<b>3</b>
Why DevOps?	
How DevOps helps?	
<b>2. DEVOPS TOOLS &amp; LIFECYCLE PHASES</b>	<b>6</b>
DevOps Life Phases	
<b>3. VERSION CONTROL WITH GIT &amp; GITHUB</b>	<b>8</b>
Top Git Commands	
<b>4. CONTINUOUS INTEGRATION WITH JENKINS</b>	<b>9</b>
Pipeline Concepts in Jenkins	
How to build CI/CD Pipeline using Jenkins?	
<b>5. CONTAINERIZATION WITH DOCKER</b>	<b>11</b>
Top Docker Commands	

# TABLE OF CONTENTS

<b>6. CONTINUOUS DEPLOYMENT WITH PUPPET</b>	<b>13</b>
Puppet for Infrastructure Automation	
<b>7. CONTINUOUS TESTING WITH SELENIUM</b>	<b>14</b>
Creating TestNG cases in Selenium	
<b>8. CONTINUOUS MONITORING WITH NAGIOS</b>	<b>15</b>
Install Nagios Core	
<b>9. TOP 30 DEVOPS INTERVIEW QUESTIONS</b>	<b>16</b>
<b>10. CAREER GUIDANCE</b>	<b>17</b>
How to become a DevOps Professional? Edureka's Structured Training Programs	

## Chapter 1

# INTRODUCTION TO DEVOPS



*DevOps is a software development strategy that bridges the gap between the developers and the IT staff. With DevOps, organizations can release small features very quickly and incorporate the feedback which they receive immediately. DevOps process involves a lot of development, testing, and deployment of technologies for developing automated CI/ CD pipelines.*

## DEVOPS FEATURES



Easily Scalable

Continuous Build, Test, Integrate and  
Deploy

Provide Collaborations

Ops Alignment



Enables Automation

Improved Business Agility



Faster Delivery

Better Response Time



Reduced IT Costs

Increase Customer Satisfaction



## 1.1 Why DevOps?

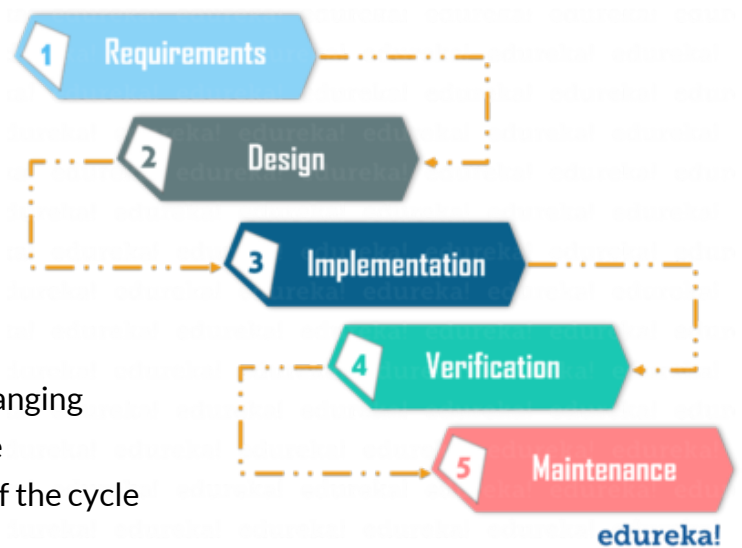
Before DevOps came into existence, IT Companies developed software using traditional methods such as [Waterfall Model & Agile Methodology](#). Let's have a quick overview of what these methodologies are and how exactly do they work.

### Waterfall Model

The Waterfall Model is a model of software development that is pretty straightforward and linear. This model follows a top-down approach.

#### Challenges

- a. Risky and uncertain
- b. Lack of visibility of the current progress
- c. Not suitable when the requirements keep changing
- d. Difficult to make changes in the testing phase
- e. The end product is available only at the end of the cycle
- f. Not suitable for large and complex projects



### Agile Methodology



Agile is an iterative based software development approach where the software project is broken down into various iterations or sprints. Every iteration has phases like the waterfall model such as requirements gathering, design, development, testing, and maintenance.

#### Challenges

- a. Highly dependent on clear customer requirements
- b. Quite difficult to predict time and effort for larger projects
- c. Not suitable for complex projects
- d. Lacks documentation efficiency
- e. Increased maintainability risks

## 1.2 How DevOps helps?

*DevOps integrates developers and operations teams to improve collaboration and productivity.*

According to the [DevOps culture](#), a single group of Engineers (developers, system admins, QA, Testers, etc., turned into DevOps Engineers) has end-to-end responsibility of the application (software) right from gathering the requirement to development, to testing, to infrastructure deployment, to application deployment and finally monitoring & gathering feedback from the end-users, then again implementing the changes.

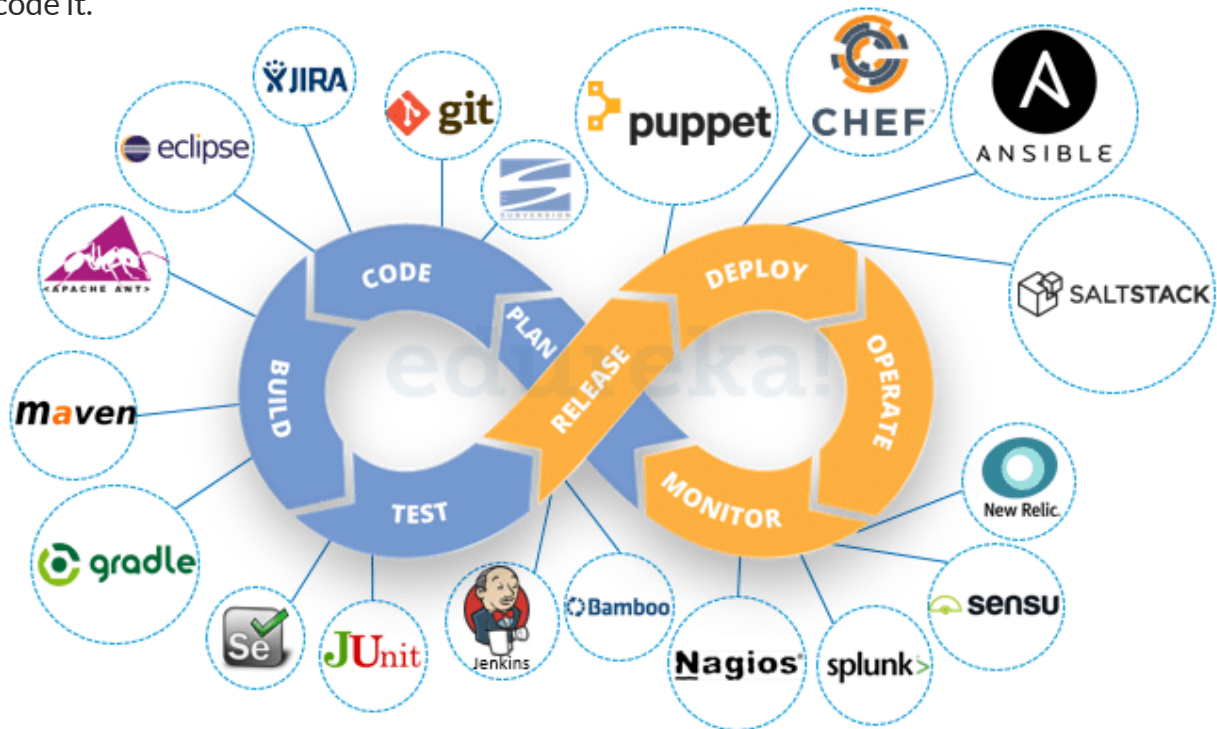
Dev Challenges	DevOps Solution
Waiting time for code deployment	Continuous integration ensures there is a quick development of code, faster testing, and a speedy feedback mechanism.
Pressure of work on old , pending and new code	There is no waiting time to deploy the code. Hence the developer focuses on building the current code.

Ops Challenges	DevOps Solution
Difficult to maintain uptime of the production environment	<b>Containerization/Virtualization</b> ensures there is a simulated environment created to run the software as containers offer great reliability for service uptime.
Tools to automate infrastructure management are not effective	<b>Configuration Management</b> helps you to organize and execute configuration plans, consistently provision the system & proactively manage their infrastructure.
No. of servers to be monitored increases	<b>Continuous Monitoring</b>
Difficult to diagnose and provide feedback on the product	Effective monitoring and feedbacks systems is established through Nagios. Thus effective administration is assured.

## Chapter 2

# DEVOPS TOOLS & LIFECYCLE PHASES

To expedite and actualize DevOps processes apart from culturally accepting it, one also needs various [DevOps tools](#) like Puppet, Jenkins, GIT, Chef, Docker, Selenium, AWS, etc., to achieve automation at various stages. Now take a look at the below DevOps diagram with various DevOps Tools closely and try to decode it.



These tools have been categorized into various stages of DevOps. Hence it is important for you to understand the DevOps Lifecycle stages first.

## 2.1 DevOps Lifecycle Phases

[DevOps Lifecycle](#) can be broadly broken down into the below-listed stages:

- Continuous Development
- Continuous Integration
- Continuous Testing
- Continuous Monitoring
- Virtualization and Containerization

Let's now learn more about each of these stages one by one.

**S  
T  
A  
G  
E  
1****CONTINUOUS DEVELOPMENT**

**Tools:** Git, SVN, Mercurial, CV

This is the phase that involves 'planning' and 'coding' of the software. You decide the project vision during the planning phase and the developers begin developing the code for the application.

**CONTINUOUS INTEGRATION**

**Tools:** Jenkins, TeamCity, Travis

This stage is the core of the entire DevOps life cycle. It is a practice in which the developers require to commit changes to the source code more frequently. This may be either on a daily or weekly basis.

**S  
T  
A  
G  
E  
2****S  
T  
A  
G  
E  
3****CONTINUOUS TESTING**

**Tools:** Jenkins, Selenium TestNG, JUnit

This is the stage where you test the developed software continuously for bugs using automation testing tools.

**CONTINUOUS DEPLOYMENT**

**Configuration Management Tools – Chef, Puppet, Ansible**  
**Containerization Tools – Docker, Vagrant**

This is the stage where you deploy the code on the production servers. It is also important to ensure that you correctly deploy the code on all the servers. The mentioned set of tools here help in achieving Continuous Deployment (CD).

**S  
T  
A  
G  
E  
4****S  
T  
A  
G  
E  
5****CONTINUOUS MONITORING**

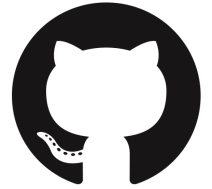
**Tools:** Splunk, ELK Stack, Nagios, New Relic

This is a very critical stage of the DevOps lifecycle where you continuously monitor the performance of your application. Here you record vital information about the use of the software. You then process this information to check the proper functionality of the application. You resolve system errors such as low memory, server not reachable, etc in this phase.



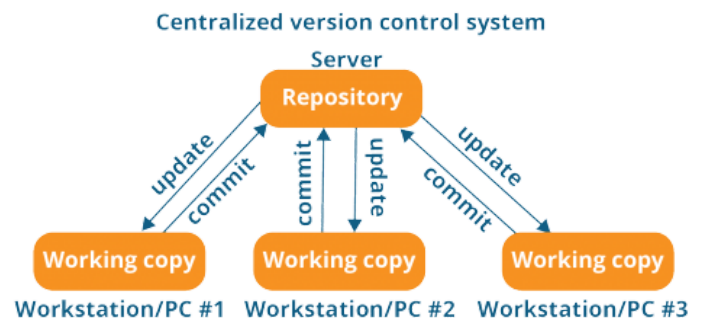
## Chapter 3

# VERSION CONTROL WITH GIT & GITHUB



Git is a Distributed Version Control tool that supports distributed non-linear workflows by providing data assurance for developing quality software.


Every programmer maintains a local repository on its own, which is actually the copy or clone of the central repository on their hard drive. They can commit and update their local repository without any interference. They can update their local repositories with new data from the central server by an operation called “pull” and affect changes to the main repository by an operation called “push” from their local repository.



Git plays a vital role when it comes to managing the code that the collaborators contribute to the shared repository. Whereas this code is then extracted for performing continuous integration to create a 'build' and test it on the test server and eventually deploy it on the production.

Git is a version control tool that will allow you to perform all kinds of operations to fetch data from the central server or push data to it whereas GitHub is a code hosting platform for version control collaboration. GitHub is a company that allows you to host a central repository on a remote server.

## TOP GIT COMMANDS

COMMAND 	USAGE	DESCRIPTION
<b>git config</b>	git config --global user.name "[name]" git config --global user.email "[email address]"	Used to set the author name and email address respectively to be used with your commits
<b>git init</b>	git init [repository name]	Used to start a new repository
<b>git clone</b>	git clone [url]	Used to obtain a repository from an existing URL
<b>git add</b>	git add [file]	Used to add a file to the staging area
<b>git status</b>	git status	Used to list all the files that have to be committed

## Chapter 4

# CONTINUOUS INTEGRATION WITH JENKINS



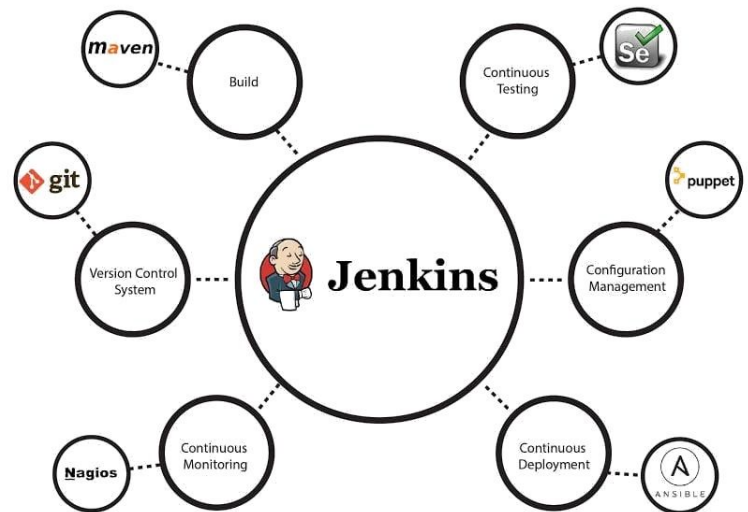
**Jenkins** is an open-source automation tool written in Java with plugins built for Continuous Integration purposes. It is used to build and test your software projects continuously making it easier for developers to integrate changes to the project, and making it easier for users to obtain a fresh 'build'.

The following are some facts about Jenkins that makes it better than other Continuous Integration tools:

**Adoption:** Jenkins is widespread, with more than 147,000 active installations and over 1 million users around the world.

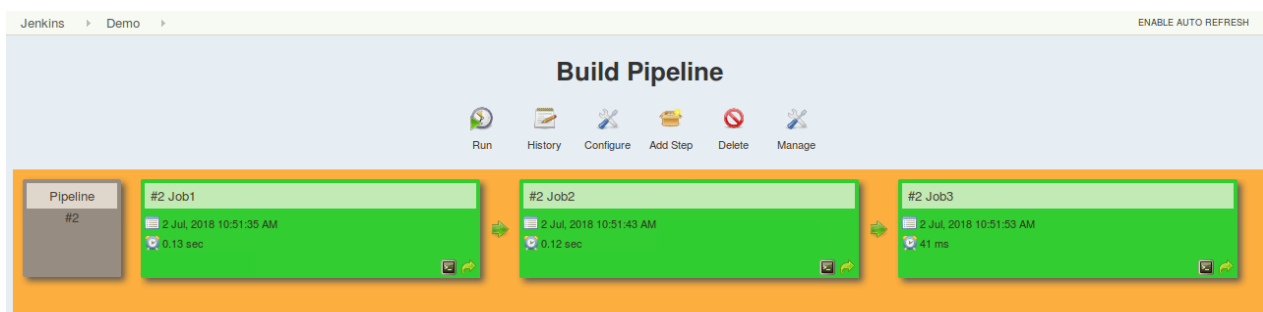
**Plugins:** Jenkins is interconnected with over 1,000 plugins that allow it to integrate with most of the development, testing and deployment tools.

To carry out continuous delivery, Jenkins introduced a new feature called the [Jenkins pipeline](#).



## 4.1 Pipeline Concepts in Jenkins

A pipeline is a collection of jobs that brings the software from version control into the hands of the end-users by using automation tools.



- 1 PIPELINE**  
This is a user-defined block that contains all the processes such as build, test, deploy, etc.
- 2 NODE**  
A node is a machine that executes an entire workflow.
- 3 AGENT**  
An agent is a directive that can run multiple builds with only one instance of Jenkins.
- 4 ANY**  
Runs the pipeline/ stage on any available agent.
- 5 NONE**  
This parameter is applied at the root of the pipeline and it indicates that there is no global agent for the entire pipeline and each stage must specify its own agent.
- 6 LABEL**  
Executes the pipeline/stage on the labelled agent.
- 7 DOCKER**  
This parameter uses Docker containers as an execution environment for the pipeline or a specific stage.
- 8 STAGES**  
This block contains all the work that needs to be carried out. The work is specified in the form of stages. There can be more than one stage within this directive. Each stage performs a specific task
- 9 STEPS**  
A series of steps can be defined within a stage block. These steps are carried out in sequence to execute a stage.

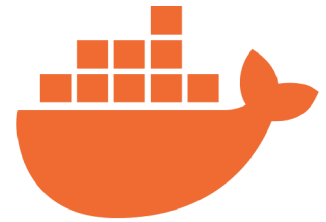
## 4.2 How to build CI/CD Pipeline using Jenkins?

### STEPS

1. Log in to Jenkins and select '**New item**' from the dashboard
2. Next, enter a name for your pipeline and select '**pipeline**' project. Click on '**OK**' to proceed
3. Scroll down to the pipeline and choose if you want a declarative pipeline or a scripted one
4. a. If you want a scripted pipeline then choose '**pipeline script**' and start typing your code  
b. If you want a declarative pipeline then select '**pipeline script from SCM**' to choose your SCM
5. Within the script, the path is the name of the Jenkinsfile that is going to be accessed from your SCM to run. Finally, click on '**apply**' and '**save**'
6. You have successfully created your first Jenkins pipeline

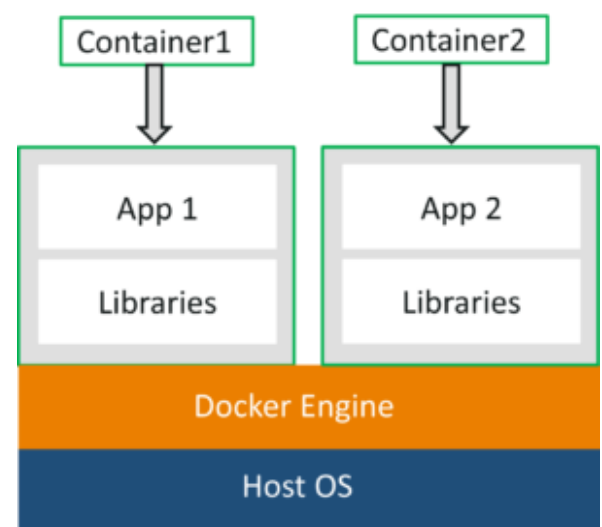
## Chapter 5

# CONTAINERIZATION WITH DOCKER



Containerization is a type of Virtualization that brings virtualization to the operating system level. [Docker](#) is a platform that packages an application and all its dependencies together in the form of containers. This containerization aspect ensures that the application works in any environment.

As you can see in the diagram, each and every application runs on separate containers and has its own set of dependencies & libraries. This makes sure that each application is independent of other applications, giving developers surety that they can build applications that will not interfere with one another.



Dockerfile, Docker Images & [Docker Containers](#) are three important terms that you need to understand while using Docker.

**01 Dockerfile**

A Dockerfile is a text document which contains all the commands that a user can call on the command line to assemble an image

**02 Docker Image**

In layman terms, Docker Image can be compared to a template which is used to create Docker Containers

**03 Docker Container**

It is a running instance of a Docker Image as they hold the entire package needed to run the application

## 5.1 Docker Commands

### 1. docker -version

This command is used to get the currently installed version of docker.

```
edureka@Manager-1: ~  
edureka@Manager-1:~$ docker --version  
Docker version 17.05.0-ce, build 89658be  
edureka@Manager-1:~$
```

### 2. docker pull

**Usage: docker pull <image name>**

This command is used to pull images from the docker repository(hub.docker.com)

### 3. docker run

**Usage: docker run -it -d <image name>**

This command is used to create a container from an image.

### 4. docker exec

**Usage: docker exec -it <container id> bash**

This command is used to access the running container.

### 5. docker stop

**Usage: docker stop <container id>**

This command stops a running container.

### 6. docker kill

**Usage: docker kill <container id>**

This command kills the container by stopping its execution immediately.

### 7. docker commit

**Usage: docker commit <container id> <username/imagename>**

This command creates a new image of an edited container on the local system.

### 8. docker push

**Usage: docker push <username/image name>**

This command is used to push an image to the docker hub repository.

### 9. docker build

**Usage: docker build <path to docker file>**

This command is used to build an image from a specified docker file.

## Chapter 6

# CONTINUOUS DEPLOYMENT WITH PUPPET



Puppet is a Configuration Management tool that is used for deploying, configuring and managing servers. Puppet uses a Master-Slave architecture in which the Master and Slave communicate through a secure encrypted channel with the help of SSL.

## KEY TERMS IN PUPPET PROGRAMMING

1

Manifests

2

Classes

3

Resources

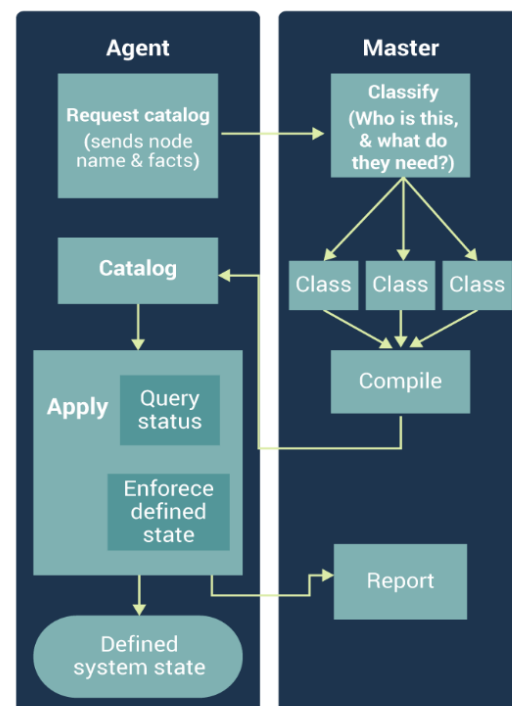
4

Puppet Modules

### 6.1 Puppet for Infrastructure Automation

Puppet uses a declarative model-based approach to IT automation. This enables Puppet to define infrastructure as code and enforce system configuration with programs.

Puppet's declarative language is used to describe the desired state of the system in files called manifests. Manifests describe how you should configure your network and operating system resources, such as files, packages, and services. Puppet compiles manifests into catalogs and it applies each catalog to its corresponding node to ensure that configuration of the node is correct across your infrastructure.



## Chapter 7

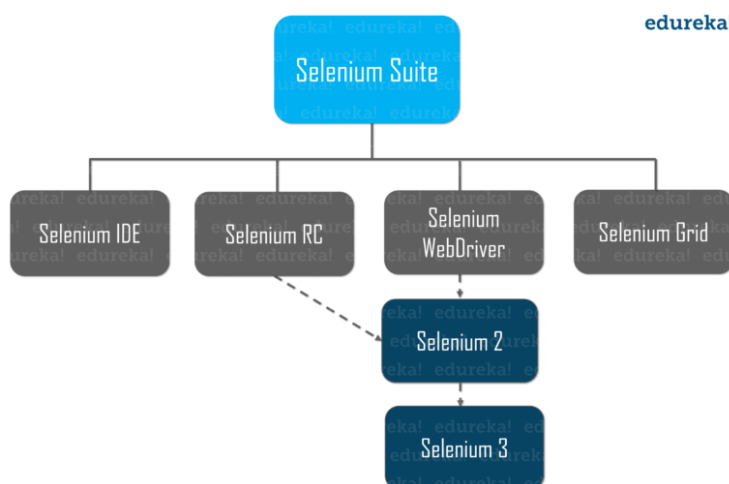
# CONTINUOUS TESTING WITH SELENIUM



**Selenium** is an open-source tool that is used for automating the tests carried out on web browsers. Test scripts can be written in various programming languages and carried out in any Operating Systems. It can be integrated with tools such as TestNG & JUnit for managing test cases and generating reports.


Selenium is mainly comprised of a suite of tools, which include:

- 1 SELENIUM IDE**  
It allows us to record and playback the scripts
- 2 SELENIUM RC**  
Selenium RC server is an HTTP proxy server
- 3 SELENIUM WEBDRIVER**  
It is a browser automation framework that accepts commands and sends them to a browser
- 4 SELENIUM GRID**  
It is used to run tests on different machines against different browsers in parallel



## 7.1 Creating TestNG cases in Selenium

### STEPS

1. Press **Ctrl+N** or Right-click on the **Test Case** folder, select "**TestNG Class**" under TestNG category and click **Next**
2. Set the Class name as '**TestNG**'. Under Annotations, check "**@BeforeMethod**", "**@AfterMethod**" and click **Finish**
3. Now it will display the newly created TestNg class under the Test Case package(folder)
4. Now write the first TestNG test case. You can divide the test case into three parts:
  - @BeforeMethod** : Launch Firefox and direct it to the Base URL
  - @Test** : Enter Username & Password to Login, Print console message and Log out
  - @AfterMethod** : Close Firefox browser
5. Run the test by right click on the test case script and select **Run As > TestNG Test** 

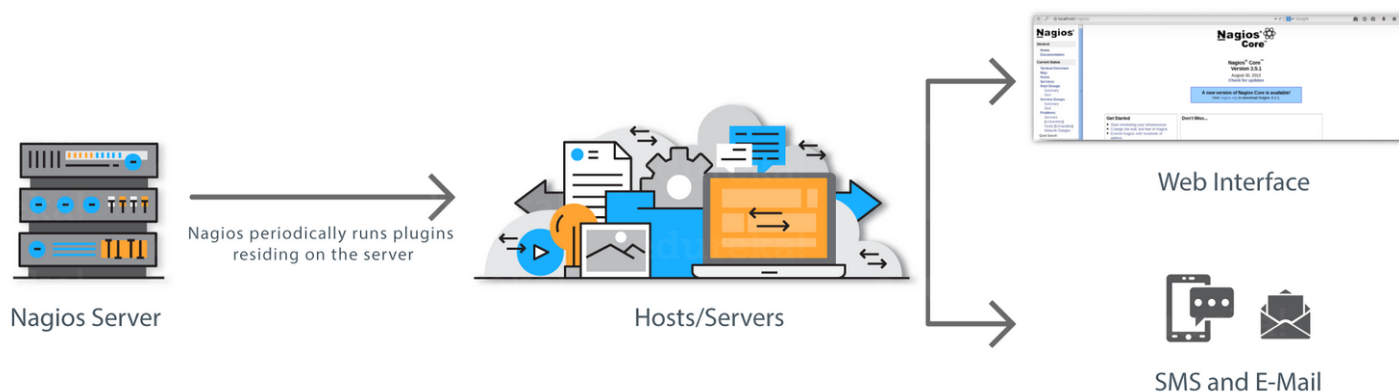
## Chapter 8

# CONTINUOUS MONITORING WITH NAGIOS



Nagios is used for Continuous monitoring of systems, applications, services, and business processes, etc., in a DevOps culture. In the event of a failure, Nagios can alert technical staff of the problem, allowing them to begin remediation processes before outages affect business processes, end-users, or customers.

Nagios runs on a server, usually as a daemon or a service. It periodically runs plugins residing on the same server, they contact hosts or servers on your network or on the internet. One can view the status information using the web interface. You can also receive email or SMS notifications if something happens. The Nagios daemon behaves like a scheduler that runs certain scripts at certain moments. It stores the results of those scripts and will run other scripts if these results change.



## 8.1 Install Nagios Core

**STEPS**

1. Install required packages on the Monitoring Server
2. Install Nagios Core, Nagios plugins and NRPE (Nagios Remote Plugin Executor)
3. Set Nagios password to access the web interface
4. Install NRPE in client



## Chapter 9

# FREQUENTLY ASKED INTERVIEW QUESTIONS

**DevOps** is a buzzword in the industry right now. Every organization is using DevOps practices but what are these companies looking for in **DevOps Professionals!** This chapter covers the questions that will help you in your DevOps Interviews and open up various DevOps career opportunities as a DevOps aspirant.

1. What are the fundamental differences between DevOps & Agile?
2. What is the need for DevOps?
3. Which are the top DevOps tools?
4. Explain with a use case where DevOps can be used in industry/ real-life.
5. What are the anti-patterns of DevOps?
6. What is Version Control? Explain some of its benefits.
7. Describe branching strategies you have used.
8. What is Git? Explain some basic Git commands.
9. What is Git rebase and how can it be used to resolve conflicts in a feature branch before the merge?
10. How do you find a list of files that have changed in a particular commit?
11. How do you set up a script to run every time a repository receives new commits through push?
12. Why do you need a Continuous Integration of development & testing?
13. How Jenkins can be moved or copied from one server to another?

14. Explain how to create a backup and copy files in Jenkins?
15. Explain how you can set up Jenkins job?
16. How will you secure Jenkins?
17. What is Automation Testing? Explain some of its benefits.
18. How to automate Testing in the DevOps lifecycle?
19. Which Testing tool are you comfortable with and what are the benefits of that tool?
20. What are the Testing types supported by Selenium?
21. Which among Puppet, Chef, SaltStack and Ansible is the best Configuration Management (CM) tool? Why?
22. What are the goals of Configuration management processes?
23. What is an Ansible module?
24. What is Nagios? How does it work?
25. What is NRPE (Nagios Remote Plugin Executor) in Nagios?
26. How exactly are containers (Docker in our case) different from hypervisor virtualization? What are the benefits?
27. What is a Docker container?
28. What is Dockerfile used for?
29. Do I lose my data when the Docker container exits?
30. How to create a Docker container?

## 100+ DEVOPS INTERVIEW QUESTIONS & ANSWERS

# CAREER GUIDANCE

## WHO IS A DEVOPS PROFESSIONAL?

DevOps Professional is somebody who understands the Software Development Lifecycle and has the outright understanding of various automation tools for developing digital pipelines (CI/ CD pipelines).

### DevOps Evangelist

DevOps Evangelist ensures that the DevOps strategy is implemented in the end-to-end development of the product, while bringing about a positive difference in the environment. The DevOps Evangelist also finds ways to improve the existing architecture of the product keeping in mind the various automation tools available and the skills that 'Dev' and 'Ops' guy consist of. Managing other DevOps roles and obtaining full efficiency from the team is his primary target.

### Automation Expert

An automation expert must understand what can be automated and how a product stack can be integrated with another product stack. They analyze, design and implement strategies for continuous deployments while ensuring high availability of production and pre-production systems.

### Software Developer/ Tester

This role(s) does the actual root-level development of the software. The traditional coders and programmers fall under this bracket. Besides development, the professionals are also responsible for unit testing, deployment, and ongoing monitoring as well.



### Release Manager

Release Manager's key area of focus is to co-ordinate and manage the product from development through deployment. Since this role is involved in an important manner, it should be donned by a technical person (manager) who understands how this technology works and how various structures fall in place.

### Quality Assurance

This professional guarantees the quality of the product by stepping beyond the traditional testing and quality checking. Here, the functionality of the product is tested to its limits to bring out every flaw and to improve on the under performance of every standalone feature because the experience of your client matters.

### Security Engineer

Security Engineers are as important as any other role because they are the one's monitoring the deliverability of the product. They work side by side with developers, embedding their recommendations (security patches) much earlier in this process. They also monitor the systems to check its performance, report any downtime faced by the system and drill down to find out what caused it.

**NEED EXPERT GUIDANCE?**

Talk to our experts and explore the right career opportunities!



**08035068109**  
**+1415 915 9044**





### DEVOPS CERTIFICATION TRAINING



Weekend/Weekday



Live Class



24 x 7 Technical Assistance

[www.edureka.co/devops-certification-training](http://www.edureka.co/devops-certification-training)

### KUBERNETES CERTIFICATION TRAINING



Weekend/Weekday



Live Class



24 x 7 Technical Assistance

[www.edureka.co/kubernetes-certification](http://www.edureka.co/kubernetes-certification)

### DEVOPS ENGINEER MASTERS PROGRAM



Weekend/Weekday



Live Class/SP



24 x 7 Technical Assistance

[www.edureka.co/masters-program/devops-engineer-training](http://www.edureka.co/masters-program/devops-engineer-training)

### DOCKER CERTIFIED ASSOCIATE TRAINING



Weekend



Live Class



24 x 7 Technical Assistance

[www.edureka.co/docker-training](http://www.edureka.co/docker-training)

## LEARNER'S REVIEWS



Nikhil R



Excellent course. Makes one familiar with most of the Devops tools in the market. **Well presented and detailed course.** Strongly recommend anyone looking to build a career in devops field.



Chetana M



The **Devops course** curriculum includes all necessary tools needed for current market. **Previous recordings and blogs** included in the course content were very helpful in gaining additional knowledge. I had a great learning experience with the trainer.



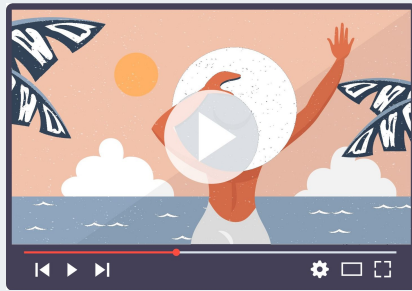
Kasthuri DS



I would recommend this course to my other friends as I found this DevOps training very useful. The trainer was very good in explaining all the concepts and clarifying the doubts for us. I was thrilled to do the **lab exercises** which gave me lot of confidence that I would be able to apply this in my job.

# Free Resources

# e!

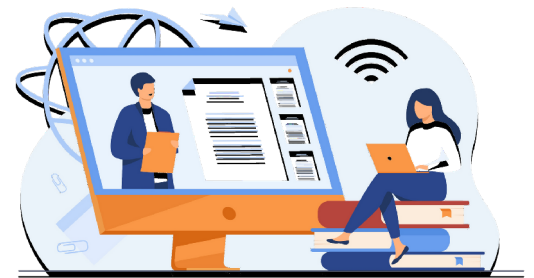


**3000+**  
Video Tutorials on  
**YouTube**

**2500+ Technical  
Blogs**



Active  
**Community**



**30+**  
Free Monthly  
**Webinars**

## About Us

There are countless online education marketplaces on the internet. And there's us. We are not the biggest. We are not the cheapest. But we are the fastest growing. We have the highest course completion rate in the industry. We aim to become the largest online learning ecosystem for continuing education, in partnership with corporates and academia. To achieve that we remain ridiculously committed to our students. Be it constant reminders, relentless masters or 24 x 7 online technical support - we will absolutely make sure that you run out of excuses to not complete the course.

## Contact Us

IndiQube ETA, 3rd Floor,  
No.38/4,  
Adjacent to Dell EMC2,  
Dodanekundi,  
Outer Ring Road, Bengaluru,  
Karnataka - 560048

☎ IN: 08035068109 | US: +1415 915 9044  
📷 [www.instagram.com/edureka.co/](http://www.instagram.com/edureka.co/)  
📘 [www.facebook.com/edurekaIN](http://www.facebook.com/edurekaIN)  
🌐 [www.linkedin.com/company/edureka/](http://www.linkedin.com/company/edureka/)  
📺 [www.youtube.com/user/edurekaIN](http://www.youtube.com/user/edurekaIN)  
📍 [t.me/s/edurekaupdates](https://t.me/s/edurekaupdates)  
🐦 [twitter.com/edurekaIN](https://twitter.com/edurekaIN)  
📌 [in.pinterest.com/edurekaco/](http://in.pinterest.com/edurekaco/)

## News & Media

INDIA  
TODAY

Edureka partners with  
NIT Warangal to upskill  
IT professionals in AI and  
Machine Learning

Deloitte.

Edureka (Brain4ce Education  
Solutions) tops Deloitte Tech  
Fast 50 2014 rankings

# edureka!