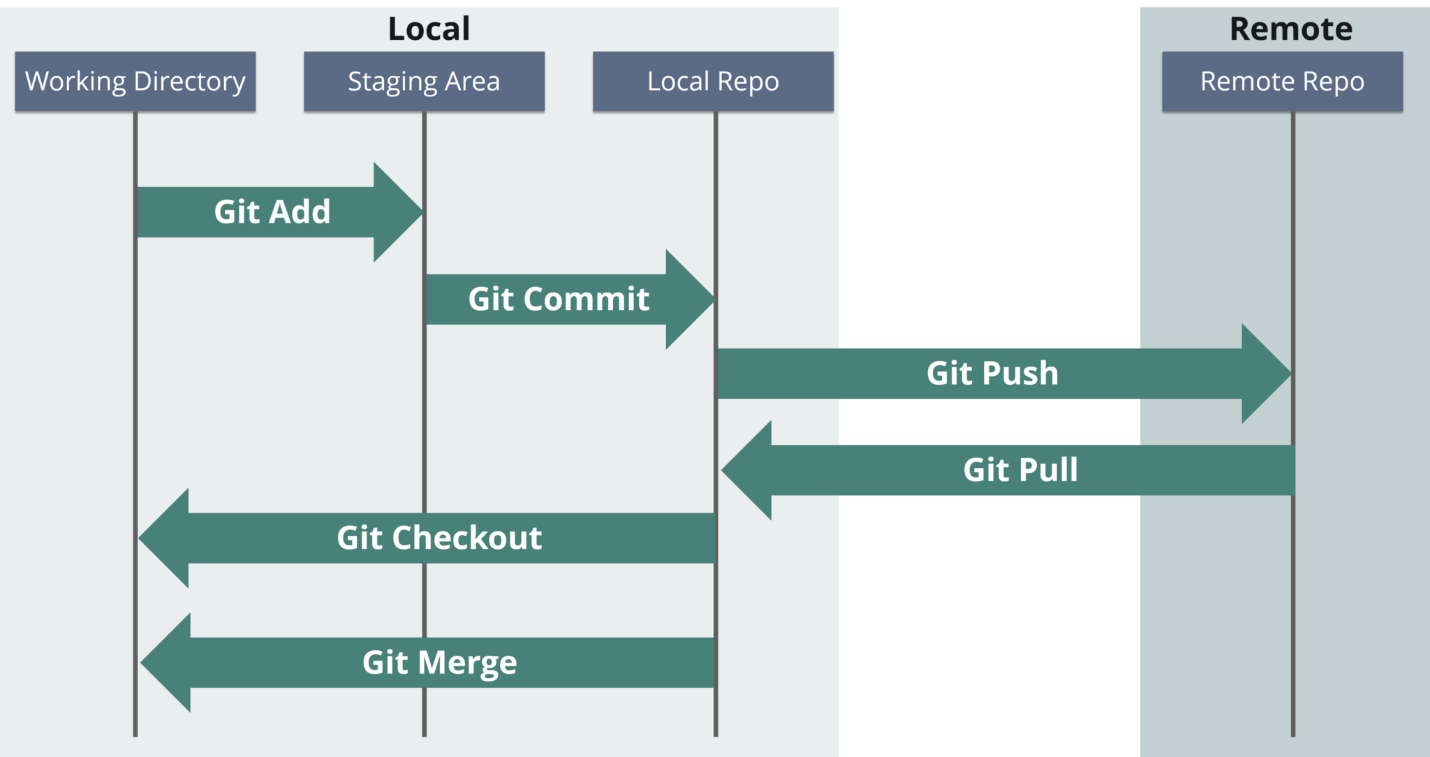
Consider the following road-map:



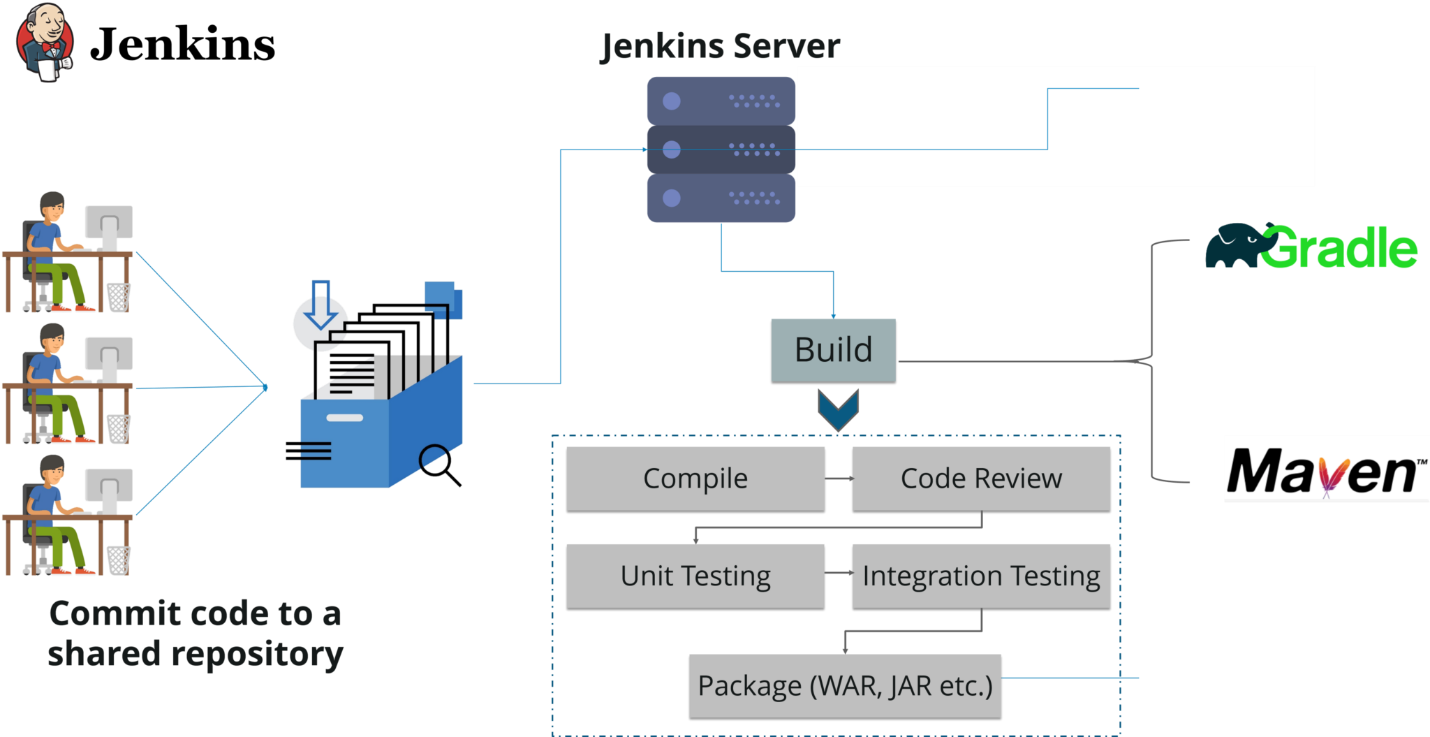
**1. Source Code Management Skills (Version Control System):**

Be proficient with at least one version control/ source code management tool, preferably Git, you should have good hands-on knowledge of Git and Git workflows.



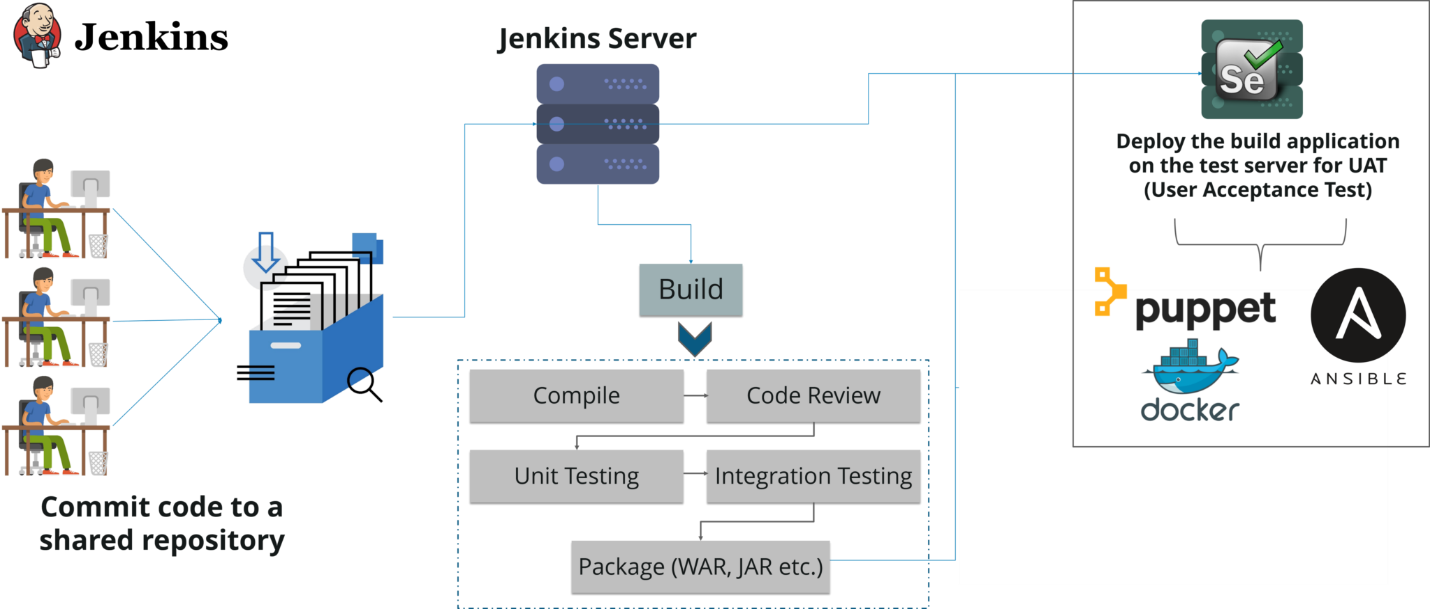
**2. Learn How To Integrate Build Tools And Source Code Management Tools For Continuous Build/ Integration (CI):**

Once you know Git, the next step is to understand how you can automate the build process, how you can continuously build the latest commit to the source code repository. For that, you should understand how automation servers like Jenkins work. Basically, how you can integrate Build tools like Maven with Source code Management tools like Git. This process is termed as Continuous Integration. Yes, I am talking about developing Continuous Integration pipeline.



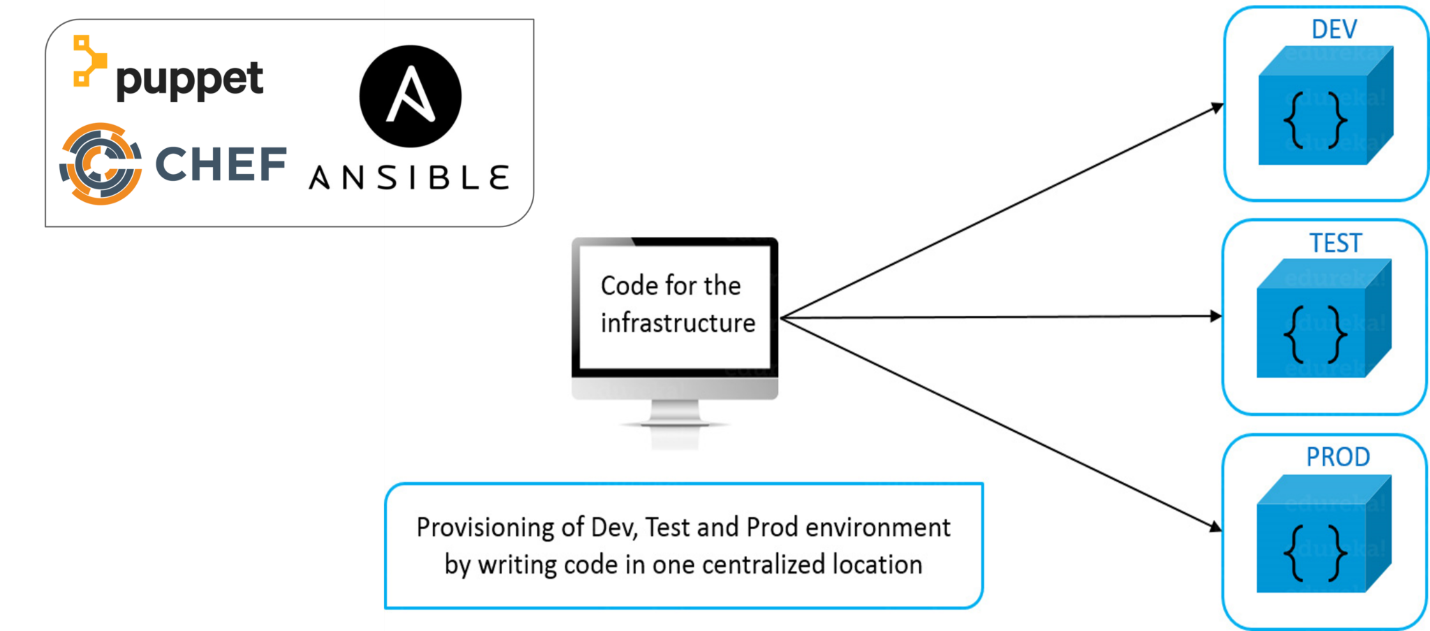
## ****3. Learn How To Take Continuous Integration To The Next Step (Continuous Delivery) For Continuous Testing:****

Next step is to learn how you can automate testing stage, also how you can make it less complex because I believe you all have faced problems like the code works fine in the dev environment but not in the test environment. You must understand how you can replicate the Dev environment in your testing servers. For this, you should understand how tools like Puppet, Chef etc. work, you can also use Docker containers for this purpose, so I would say learning Docker is a must. Also, you must learn how you can continuously test your code after every commit. You again need to integrate Automation testing tools like Selenium with Jenkins. This is termed as Continuous Delivery.



## ****4. Learn How To Deploy And Configure Dev, Test And Prod Environment:****

Finally, you should understand how you can safely deploy the application in the production server. Again, you need tools like Puppet or Docker to configure and deploy it in the prod environment.



## ****5. Learn How You Can Use Monitoring Tools To Collect Useful Feedback:****

Also, it is very important for a DevOps Engineer to collect feedback and implement the changes quickly, for that you should have knowledge of monitoring tools like Nagios, Splunk etc.



## Source: <https://ifritltd.com/2017/08/22/dockerizing-jenkins-build-logs-with-elk-stack-filebeat-elasticsearch-logstash-and-kibana/>

## ****6. Understand How Cloud Service Platforms Work:****

Along with all these things you should have good cloud knowledge to combine the befits of Cloud and DevOps. Most of the organizations have their infrastructure on Cloud, so it is a must learn skill.



Who said becoming a DevOps Engineer is a cake walk. But, we all know  “**no pain no gain**” :)

Now the question is from where you can acquire the above-mentioned skills.

## ****Where Should I Learn DevOps?****

As I mentioned above to get the theoretical knowledge you can check the official website of all the major DevOps tools. But, that is not enough to get a DevOps job, you should have hands-on knowledge. For that, you need to work on projects, which might be there in your organization. If not, you can also consider the projects provided by edureka!, it will include all the stages involved in DevOps and also include all the major DevOps tools.

[**https://www.edureka.co/blog/top-10-devops-tools/**](https://www.edureka.co/blog/top-10-devops-tools/)

 **[Git Tutorial](https://www.edureka.co/blog/git-tutorial/" \t "_blank)**

 **[jenkins Tutorial](https://www.edureka.co/blog/jenkins-tutorial/" \t "_blank)**

 **[Docker Tutorial](https://www.edureka.co/blog/docker-tutorial" \t "_blank)**

 **[Puppet Tutorial](https://www.edureka.co/blog/puppet-tutorial/" \t "_blank)**

 **[Chef Tutorial](https://www.edureka.co/blog/chef-tutorial/" \t "_blank)**

 **[Ansible Tutorial](https://www.edureka.co/blog/ansible-tutorial/" \t "_blank)**

 **[Nagios Tutorial](https://www.edureka.co/blog/nagios-tutorial/" \t "_blank)**

 [**AWS Tutorial**](https://www.edureka.co/blog/what-is-aws/)