**Projects**

* [Projects](https://darey.io/project-listings/)
* [Docs](https://www.darey.io/projects/)
* [(STEP 21) PROJECT 11: Ansible – Automate Project 7 to 10](https://www.darey.io/docs/ansible-configuration-management-automate-project-7-to-10/)

**ANSIBLE CONFIGURATION MANAGEMENT – AUTOMATE PROJECT 7 TO 10**

[Start Project](https://www.darey.io/docs/ansible-configuration-management-automate-project-7-to-10/?checkread=1)Submit Project for Review

You have been implementing some interesting projects up untill now, and that is awesome.

In Projects 7 to 10 you had to perform a lot of manual operations to seet up virtual servers, install and configure required software, deploy your web application.

This Project will make you appreciate DevOps tools even more by making most of the routine tasks automated with [Ansible](https://en.wikipedia.org/wiki/Ansible_(software)) [Configuration Management](https://www.redhat.com/en/topics/automation/what-is-configuration-management#:~:text=Configuration%20management%20is%20a%20process,in%20a%20desired%2C%20consistent%20state.&text=Managing%20IT%20system%20configurations%20involves,building%20and%20maintaining%20those%20systems.), at the same time you will become confident at writing code using declarative language such as [YAML](https://en.wikipedia.org/wiki/YAML).

Let us get started!

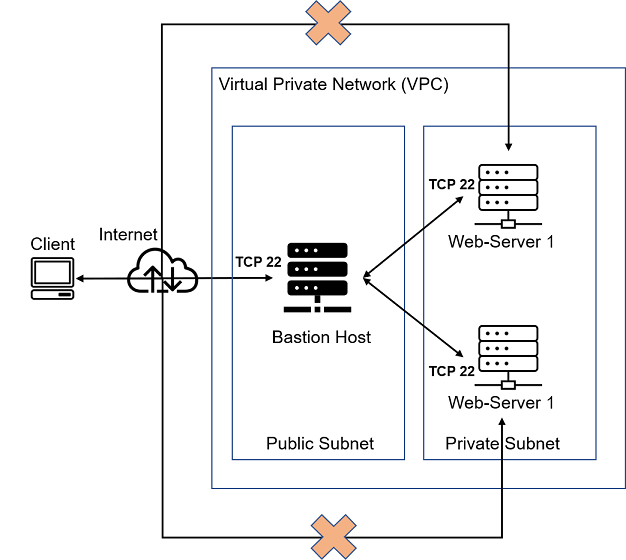
Instructions On How To Submit Your Work For Review And Feedback

To submit your work for review and feedback – follow [**this instruction**](https://darey.io/docs/how-to-submit-your-work-for-review-and-feedback-2/).

Ansible Client as a Jump Server (Bastion Host)

A [**Jump Server**](https://en.wikipedia.org/wiki/Jump_server) (sometimes also referred as [**Bastion Host**](https://en.wikipedia.org/wiki/Bastion_host)) is an intermediary server through which access to internal network can be provided. If you think about the current architecture you are working on, ideally, the webservers would be inside a secured network which cannot be reached directly from the Internet. That means, even DevOps engineers cannot SSH into the Web servers directly and can only access it through a Jump Server – it provide better security and reduces [attack surface](https://en.wikipedia.org/wiki/Attack_surface).

On the diagram below the Virtual Private Network (VPC) is divided into [two subnets](https://docs.aws.amazon.com/vpc/latest/userguide/VPC_Subnets.html) – Public subnet has public IP addresses and Private subnet is only reachable by private IP addresses.



When you reach [Project 15](https://www.darey.io/docs/aws-cloud-solution-for-2-company-websites-using-a-reverse-proxy-technology/), you will see a Bastion host in proper action. But for now, we will develop **Ansible** scripts to simulate the use of a Jump box/Bastion host to access our Web Servers.

Task

* Install and configure Ansible client to act as a Jump Server/Bastion Host
* Create a simple Ansible playbook to automate servers configuration

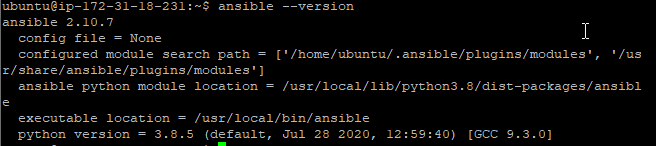
**INSTALL AND CONFIGURE ANSIBLE ON EC2 INSTANCE**

1. Update Name tag on your Jenkins EC2 Instance to Jenkins-Ansible. We will use this server to run playbooks.
2. In your GitHub account create a new repository and name it ansible-config-mgt.
3. Instal **Ansible**

sudo apt update

sudo apt install ansible

Check your Ansible version by running ansible --version



1. Configure Jenkins build job to save your repository content every time you change it – this will solidify your Jenkins configuration skills acquired in [Project 9](https://www.darey.io/docs/tooling-website-deployment-automation-with-continuous-integration-introduction-to-jenkins/).

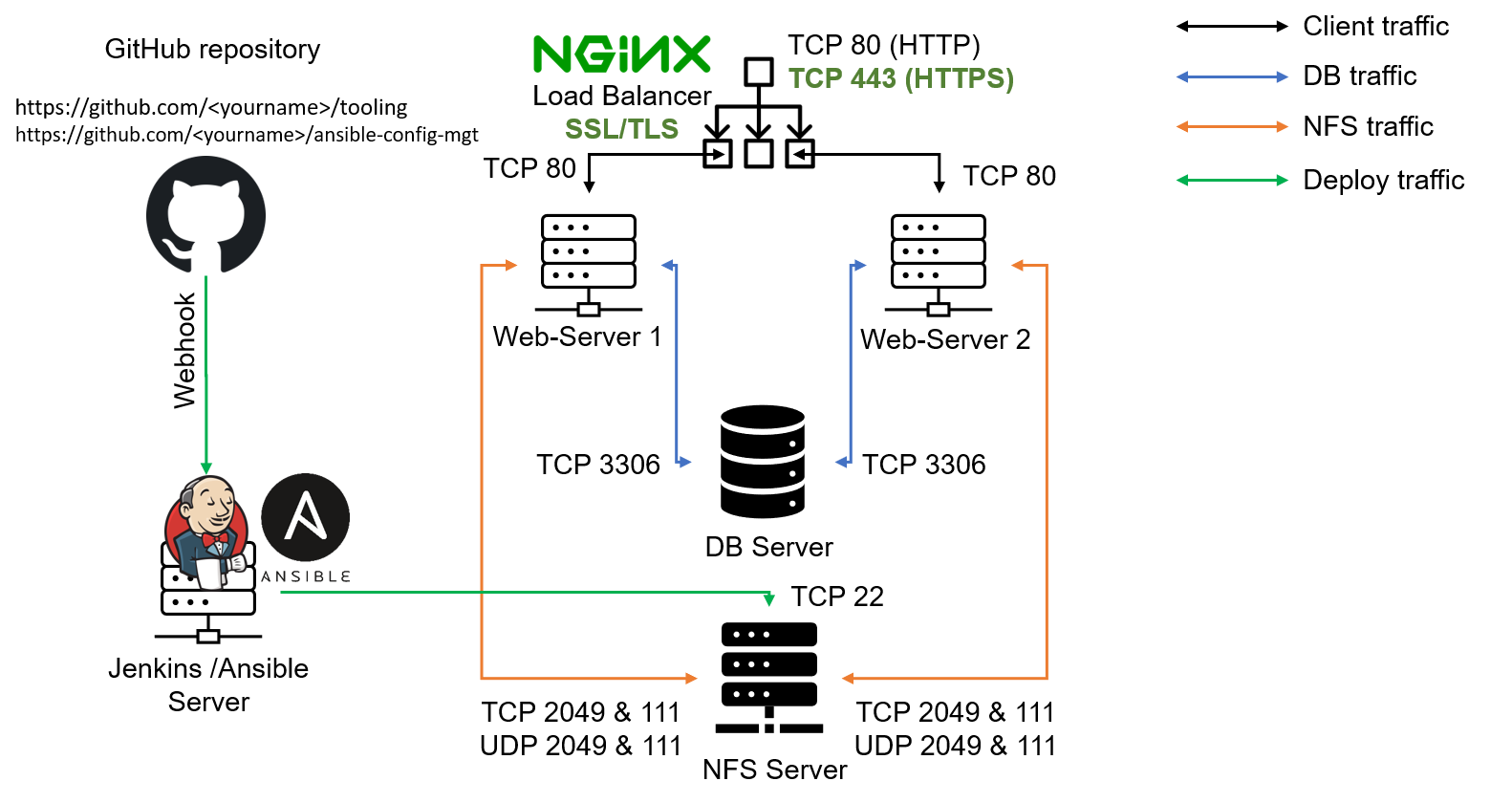
* Create a new Freestyle project ansible in Jenkins and point it to your ‘ansible-config-mgt’ repository.
* Configure Webhook in GitHub and set webhook to trigger ansible build.
* Configure a Post-build job to save all (\*\*) files, like you did it in Project 9.

1. Test your setup by making some change in README.MD file in master branch and make sure that builds starts automatically and Jenkins saves the files (build artifacts) in following folder

ls /var/lib/jenkins/jobs/ansible/builds/<build\_number>/archive/

**Note:** Trigger Jenkins project execution only for /main (master) branch.

Now your setup will look like this:



**Tip** Every time you stop/start your Jenkins-Ansible server – you have to reconfigure GitHub webhook to a new IP address, in order to avoid it, it makes sense to allocate an [Elastic IP](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/elastic-ip-addresses-eip.html) to your Jenkins-Ansible server (you have done it before to your LB server in [Project 10](https://professional-pbl.darey.io/en/latest/project10.html)). Note that Elastic IP is free only when it is being allocated to an EC2 Instance, so do not forget to release Elastic IP once you terminate your EC2 Instance.

Step 2 – Prepare your development environment using Visual Studio Code

1. First part of ‘DevOps’ is ‘Dev’, which means you will require to write some codes and you shall have proper tools that will make your coding and debugging comfortable – you need an [Integrated development environment (IDE)](https://en.wikipedia.org/wiki/Integrated_development_environment) or [Source-code Editor](https://en.wikipedia.org/wiki/Source-code_editor). There is a plethora of different IDEs and Source-code Editors for different languages with their own advantages and drawbacks, you can choose whichever you are comfortable with, but we recommend one free and universal editor that will fully satisfy your needs – [**Visual Studio Code (VSC)**](https://en.wikipedia.org/wiki/Visual_Studio_Code), you can get it [here](https://code.visualstudio.com/download).
2. After you have successfully installed VSC, configure it to [connect to your newly created GitHub repository](https://www.darey.io/docs/install-and-configure-ansible-on-ec2-instance/www.youtube.com/watch?v=3Tn58KQvWtU&t).
3. Clone down your ansible-config-mgt repo to your Jenkins-Ansible instance

git clone <ansible-config-mgt repo link>

# BEGIN ANSIBLE DEVELOPMENT

1. In your ansible-config-mgt GitHub repository, create a new branch that will be used for development of a new feature.

**Tip**: Give your branches descriptive and comprehensive names, for example, if you use [Jira](https://www.atlassian.com/software/jira) or [Trello](https://trello.com/) as a project management tool – include ticket number (e.g. PRJ-145) in the name of your branch and add a topic and a brief description what this branch is about – a bugfix, hotfix, feature, release (e.g. feature/prj-145-lvm)

1. Checkout the newly created feature branch to your local machine and start building your code and directory structure
2. Create a directory and name it playbooks – it will be used to store all your playbook files.
3. Create a directory and name it inventory – it will be used to keep your hosts organised.
4. Within the playbooks folder, create your first playbook, and name it common.yml
5. Within the inventory folder, create an inventory file (.yml) for each environment (Development, Staging Testing and Production) dev, staging, uat, and prod respectively.

##### Step 4 – Set up an Ansible Inventory

An Ansible inventory file defines the hosts and groups of hosts upon which commands, modules, and tasks in a playbook operate. Since our intention is to execute Linux commands on remote hosts, and ensure that it is the intended configuration on a particular server that occurs. It is important to have a way to organize our hosts in such an Inventory.

Save below inventory structure in the inventory/dev file to start configuring your development servers. Ensure to replace the IP addresses according to your own setup.

**Note**: Ansible uses TCP port 22 by default, which means it needs to ssh into target servers from Jenkins-Ansible host – for this you can implement the concept of [ssh-agent](https://smallstep.com/blog/ssh-agent-explained/" \l ":~:text=ssh%2Dagent%20is%20a%20key,you%20connect%20to%20a%20server.&text=It%20doesn't%20allow%20your%20private%20keys%20to%20be%20exported." \t "_blank). Now you need to import your key into ssh-agent:

To learn how to setup SSH agent and connect VS Code to your Jenkins-Ansible instance, please see this video:

* For Windows users – [ssh-agent on windows](https://youtu.be/OplGrY74qog" \t "_blank)
* For Linux users – [ssh-agent on linux](https://youtu.be/OplGrY74qog" \t "_blank)

eval `ssh-agent -s`

ssh-add <path-to-private-key>

Confirm the key has been added with the command below, you should see the name of your key

ssh-add -l

Now, ssh into your Jenkins-Ansible server using ssh-agent

ssh -A ubuntu@public-ip

Also notice, that your Load Balancer user is ubuntu and user for RHEL-based servers is ec2-user.

Update your inventory/dev.yml file with this snippet of code:

[nfs]

<NFS-Server-Private-IP-Address> ansible\_ssh\_user='ec2-user'

[webservers]

<Web-Server1-Private-IP-Address> ansible\_ssh\_user='ec2-user'

<Web-Server2-Private-IP-Address> ansible\_ssh\_user='ec2-user'

[db]

<Database-Private-IP-Address> ansible\_ssh\_user='ec2-user'

[lb]

<Load-Balancer-Private-IP-Address> ansible\_ssh\_user='ubuntu'

# CREATE A COMMON PLAYBOOK

##### Step 5 – Create a Common Playbook

It is time to start giving Ansible the instructions on what you needs to be performed on all servers listed in inventory/dev.

In common.yml playbook you will write configuration for repeatable, re-usable, and multi-machine tasks that is common to systems within the infrastructure.

Update your playbooks/common.yml file with following code:

---

- name: update web, nfs and db servers

hosts: webservers, nfs, db

remote\_user: ec2-user

become: yes

become\_user: root

tasks:

- name: ensure wireshark is at the latest version

yum:

name: wireshark

state: latest

- name: update LB server

hosts: lb

remote\_user: ubuntu

become: yes

become\_user: root

tasks:

- name: Update apt repo

apt:

update\_cache: yes

- name: ensure wireshark is at the latest version

apt:

name: wireshark

state: latest

Examine the code above and try to make sense out of it. This playbook is divided into two parts, each of them is intended to perform the same task: install [wireshark](https://en.wikipedia.org/wiki/Wireshark" \t "_blank) utility (or make sure it is updated to the latest version) on your RHEL 8 and Ubuntu servers. It uses root user to perform this task and respective package manager: yum for RHEL 8 and apt for Ubuntu.

Feel free to update this playbook with following tasks:

* Create a directory and a file inside it
* Change timezone on all servers
* Run some shell script
* …

For a better understanding of Ansible playbooks – [watch this video from RedHat](https://youtu.be/ZAdJ7CdN7DY) and read [this article](https://www.redhat.com/en/topics/automation/what-is-an-ansible-playbook).

##### Step 6 – Update GIT with the latest code

Now all of your directories and files live on your machine and you need to push changes made locally to GitHub.

In the real world, you will be working within a team of other DevOps engineers and developers. It is important to learn how to collaborate with help of GIT. In many organisations there is a development rule that do not allow to deploy any code before it has been reviewed by an extra pair of eyes – it is also called "Four eyes principle".

Now you have a separate branch, you will need to know how to raise a [Pull Request (PR)](https://docs.github.com/en/github/collaborating-with-issues-and-pull-requests/about-pull-requests), get your branch peer reviewed and merged to the master branch.

Commit your code into GitHub:

1. use git commands to add, commit and push your branch to GitHub.

git status

git add <selected files>

git commit -m "commit message"

1. Create a Pull request (PR)
2. Wear a hat of another developer for a second, and act as a reviewer.
3. If the reviewer is happy with your new feature development, merge the code to the master branch.
4. Head back on your terminal, checkout from the feature branch into the master, and pull down the latest changes.

Once your code changes appear in master branch – Jenkins will do its job and save all the files (build artifacts) to /var/lib/jenkins/jobs/ansible/builds/<build\_number>/archive/ directory on Jenkins-Ansible server.

# RUN FIRST ANSIBLE TEST

#### Step 7 – Run first Ansible test

Now, it is time to execute ansible-playbook command and verify if your playbook actually works:

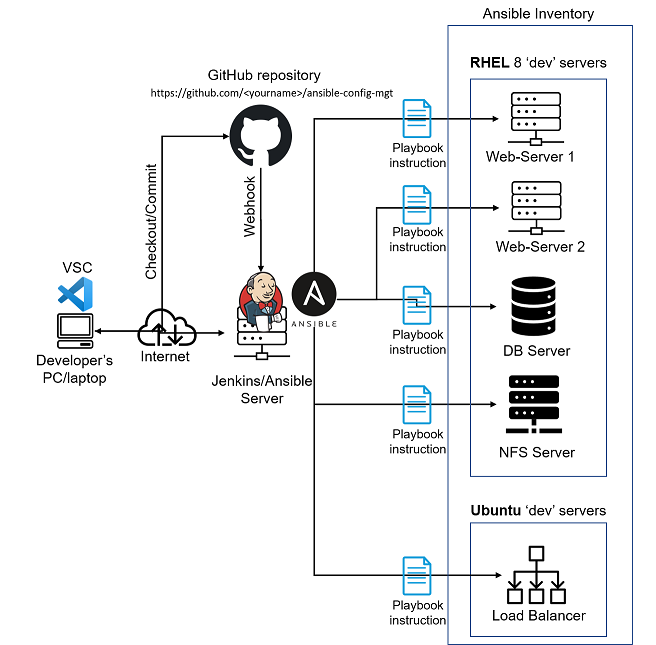
cd ansible-config-mgt

ansible-playbook -i inventory/dev.yml playbooks/common.yml

You can go to each of the servers and check if wireshark has been installed by running which wireshark or wireshark --version



Your updated with Ansible architecture now looks like this:



#### Optional step – Repeat once again

Update your ansible playbook with some new Ansible tasks and go through the full checkout -> change codes -> commit -> PR -> merge -> build -> ansible-playbook cycle again to see how easily you can manage a servers fleet of any size with just one command!

#### Congratulations

You have just automated your routine tasks by implementing your first Ansible project! There is more exciting projects ahead, so lets keep it moving!

