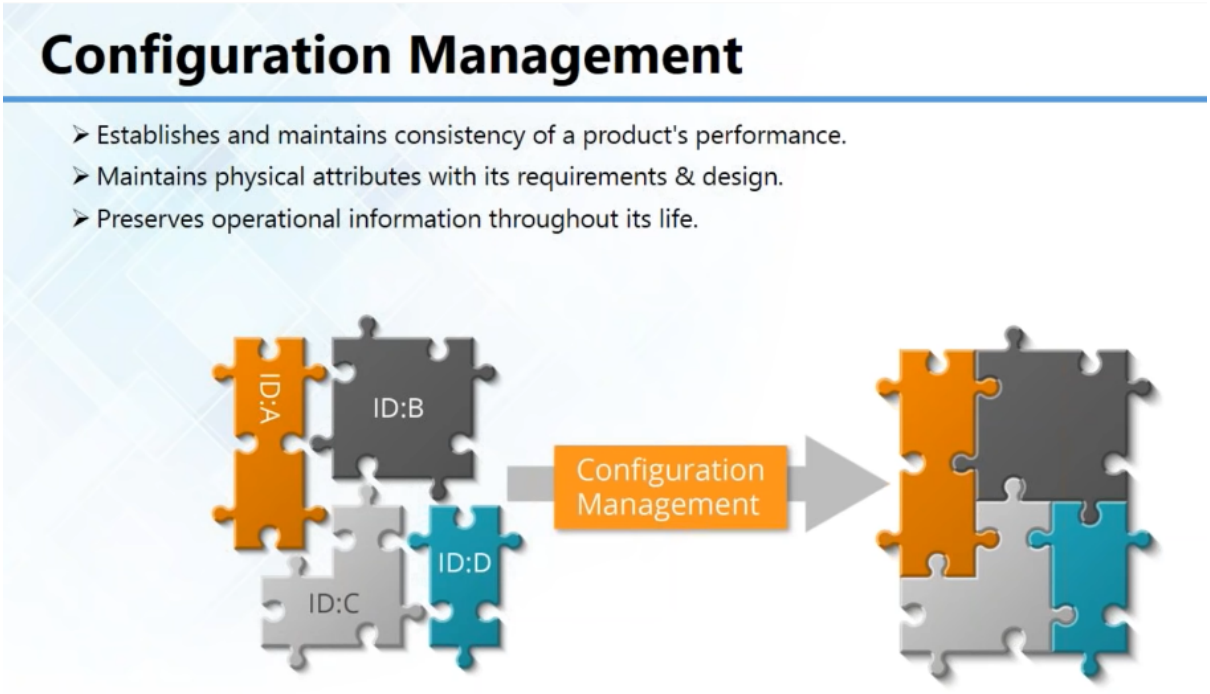
Configuration management is the management of your software on top of your hardware.

configuration management is applied over the entire lifecycle of your system and hence it provides you with a very good visibility and control.



Visibility:

visibility it means that you can continuously check and monitor the performances of all your systems so at any time the performance of any of your system is degrading the configuration management system will notify you and hence you can prevent errors before it actually.

Control :

control I mean that you have the power to change anything if any of your servers fade you can reconfigure it again to repair it so that it is up and running again or you can even replace the server if needed.

Historical Data:

the configuration management system holds the entire historical data of your infrastructure it documents all the snapshots of every version of your infrastructure.

so over all the configuration management process facilitates the orderly management of your system information and system changes so that it can use it for beneficial purposes.

Snapshot:

The snapshot means that it records the configuration details which are nothing but data descriptions of the overall state of your system at a time and where does it store all the snapshots while configuration management system has their own repository which is known as we **CMDB** or it's called the configuration management database so this is where all the configuration details get stored.

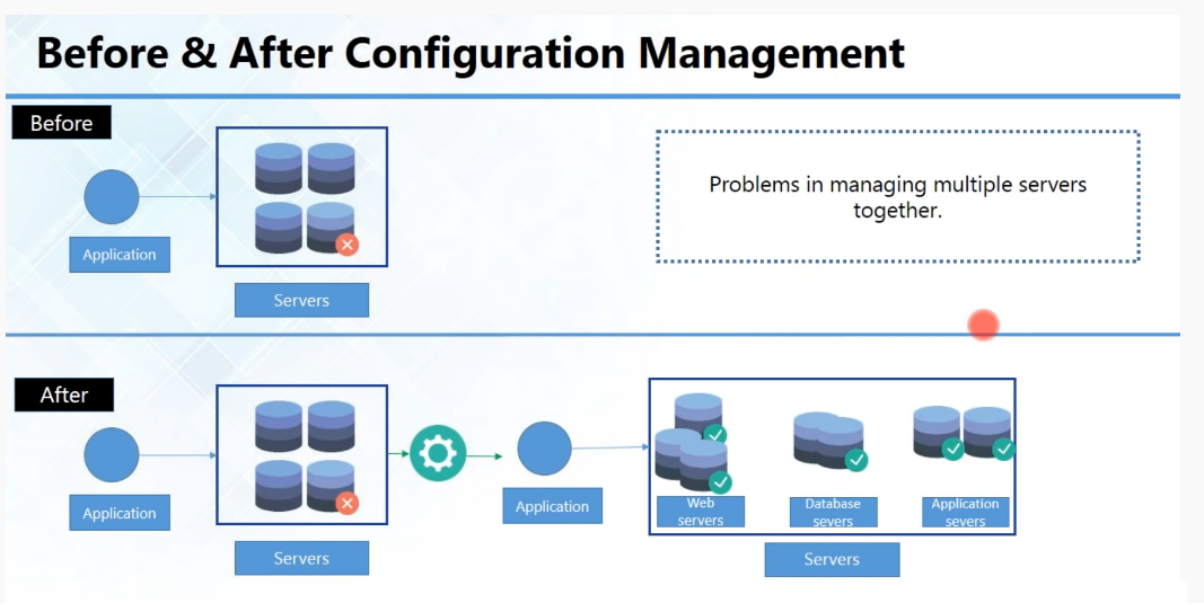
Necessities of Configuration management and problems before that.

Types of Servers: Webservers, Database Servers, Application Servers etc.

Problems In the Market:

The first problem is managing multiple servers.

In earlier days, all servers are managed manually, system administrator need to login to every system and make the changes if needed. This was taking lot of time for the administrator and no time found for administrator to monitor the performance of the application.



Baselining:

After configuration systems in place, first thing happened is organizing the servers in efficient manger and doing proper grouping like Webservers/Database Servers/Application Servers. This is called baselining.

For Example, if I wanted to install **LAMP** stack for my requirement, LAMP stack is a bundle of software, where L stands for Linux, A stands for Apache server, M stands for MySql, P stands for PHP. So, I need this different software for different purposes.

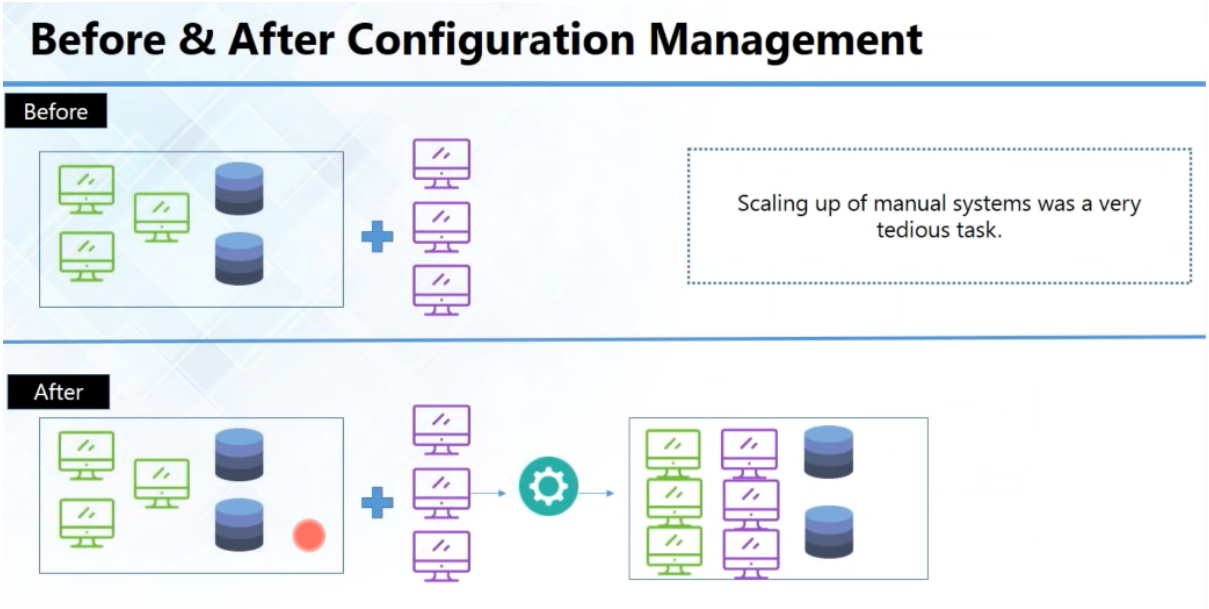
Apache server to host my web pages, PHP for Web Development, Linux for Operating system, MySQL for Data Manipulation Language or data definition language.

With Configuration management, I can know where to install which software and it is easy to identify a problem, for example, if any webpage is not working , we can go and check Webservers and I don't have to check database server etc..If I am not able to insert data/ delete data, I can search in DB servers. So, we can manage and maintain all the server efficiently with this system.

Second problem: Scaling up and Scaling Down.

Providing the sources like RAM, Memory and CPU usage based on the traffic and usage of the system.

Ex : If there is a mega sale for Christmas, it is going to be huge rise in the traffic, so, you might need more web servers to handle that amount of servers, and you might be needing of load balancers to handle and distribute the traffic, even If you want to add a new hardware within short span of time, you need to provision the machines as needed with the help of configuration management.



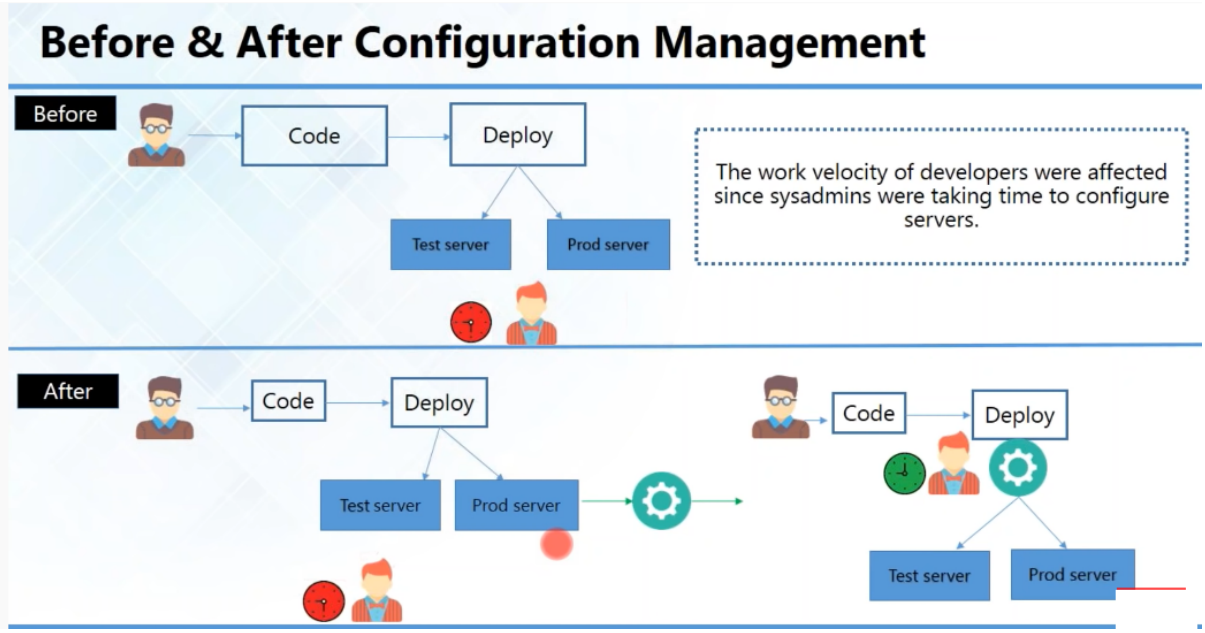
Configuration management uses the **Scripts, Playbooks** to do the upscaling and we can disable additional machines with the help of same Scripts or Playbooks.

So, Configuration management is helpful for auto scaling of infrastructure based on need of usage of application.

The next problem is Work Velocity:

After agile into the picture, all the developers are pushing their code into repositories, but to build, install and deploy in the servers, server administrator need to configure every system manually, which is a very time taking process. and developers need to wait for long to get the configuration set up.

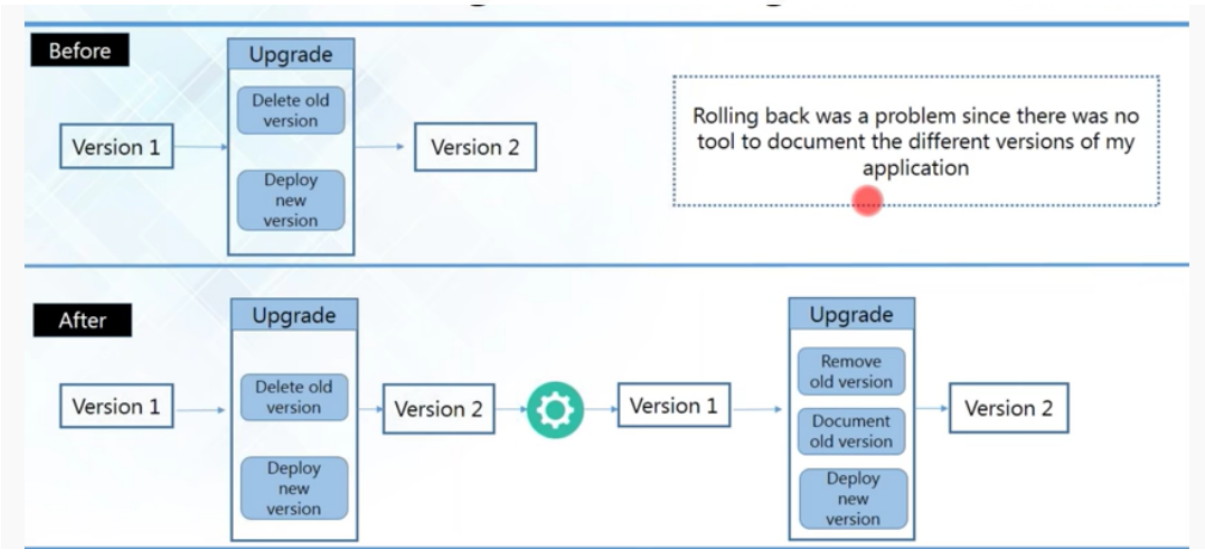
With the help of Configuration Management tools, Developers and System Administrators are going in the same velocity by giving configured servers for deployment



.

Last problem is Rolling back:

With configuration management tools, we can roll back any version any time depends on customers feedback or change of requirements in ZERO Downtime.



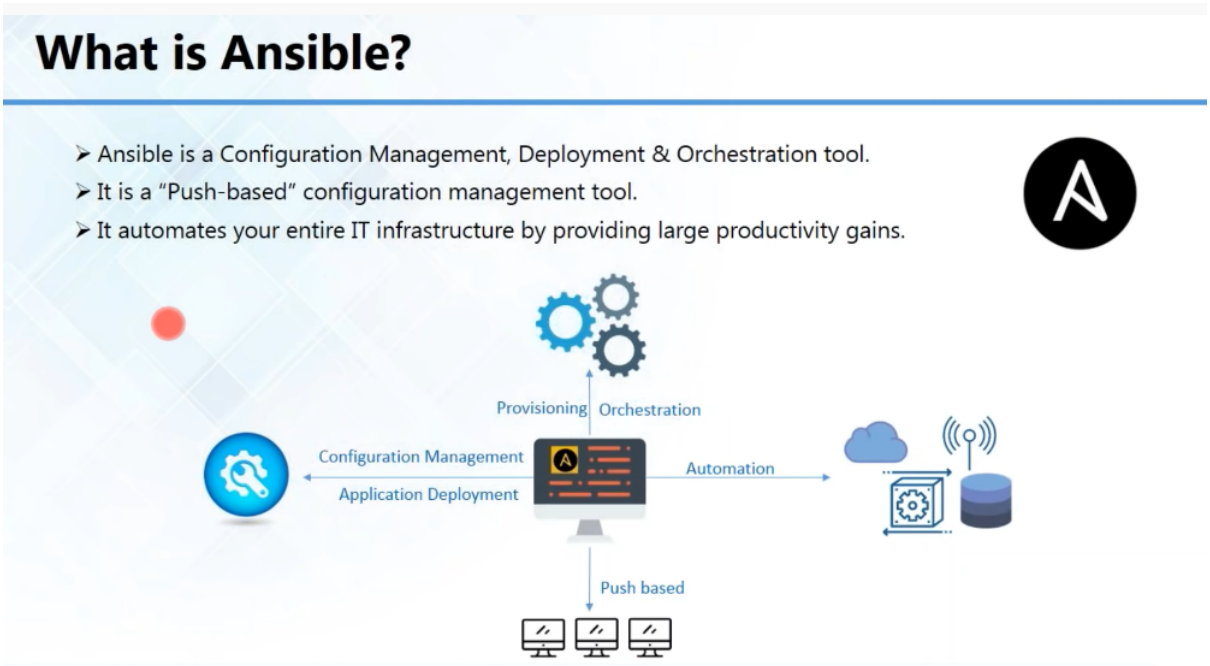
Four most popular tools in the market.

Ansible and Salt stack are called PUSH based configuration management tool.

PUSH based means, you can directly push your configuration changes directly on to your node machines.

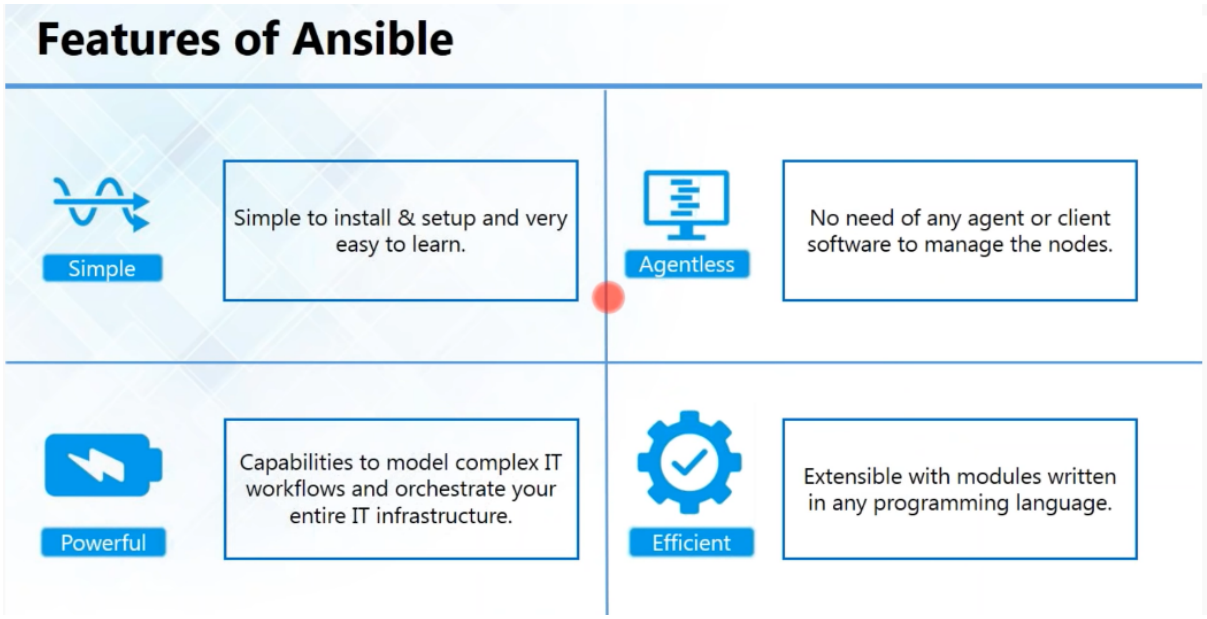
Chef and Puppet are PULL based configuration management tools, that means, they rely on Central Server for managing the configurations.

Note: Central Server controls the nodes



- Ansible is a configuration management tool that can be used for **provisioning/arranging, orchestration, application deployment automation** and it's a push-based configuration management tool.
- It automates your entire IT Infrastructure and gives you large productivity gains and it automate everything like automate your cloud network, your servers, IT Process.
- Automation means, tuning your infrastructure administration into a **code base** and it describes all the process necessary for deploying a server with the help of set of Scripts, that can be reused again.

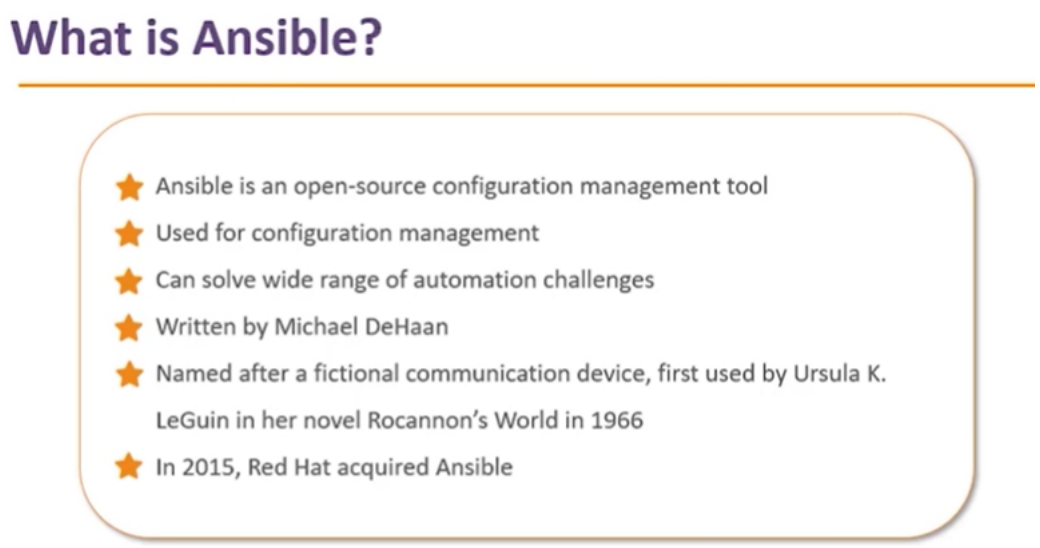
Writing the scripts in Ansible means, writing **play books**.

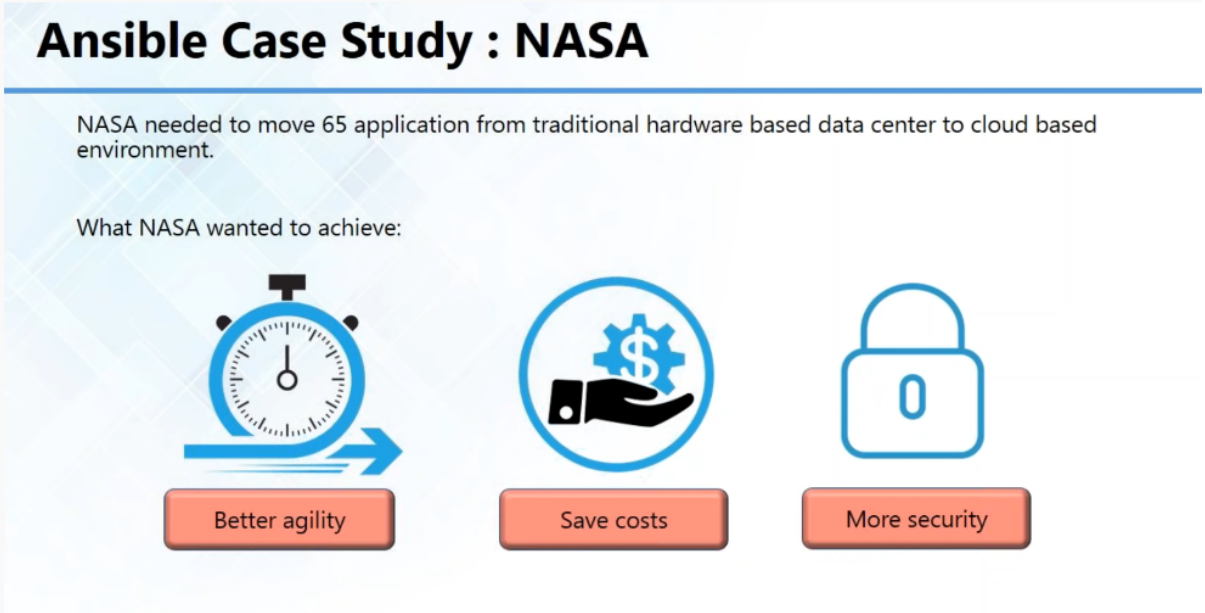


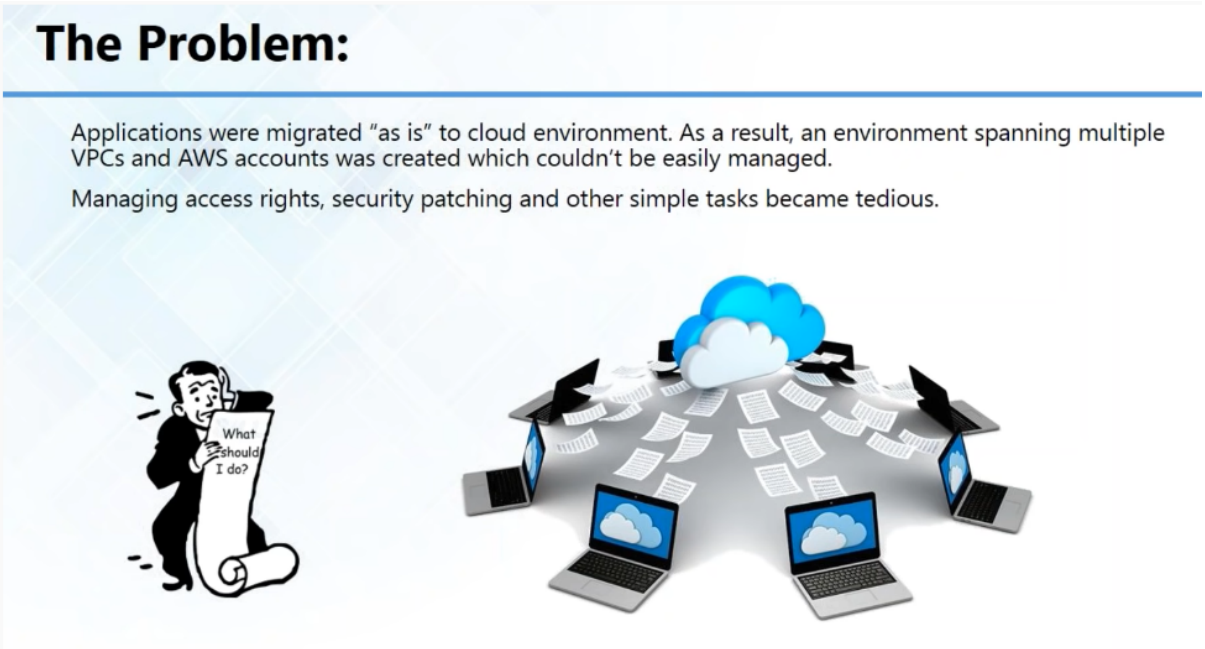
Simple: Ansible uses a simple syntax written in YAML called ***playbooks***. YAML is a human-readable data serialization language. It is extraordinarily simple. So, no special coding skills are required and even people in your IT organization, who do not know what is Ansible can likely read a playbook and understand what is happening. Ansible always executes tasks in order. It is simple to install too.

Agentless: Finally, Ansible is completely agentless. There are no agents/software or additional firewall ports that you need to install on the client systems or hosts which you want to automate

Powerful & Flexible: Ansible has powerful features that can enable you to model even the most complex IT workflows. In this aspect, Ansible's *batteries included approach* (This philosophy means that something is self-sufficient, comes out-of-the-box ready to use, with everything that is needed) can manage the infrastructure, networks, operating systems and services that you are already using, as Ansible provides you with hundreds of modules to manage them. Together Ansible's capabilities allow you to orchestrate the entire application environment regardless of where it is deployed.



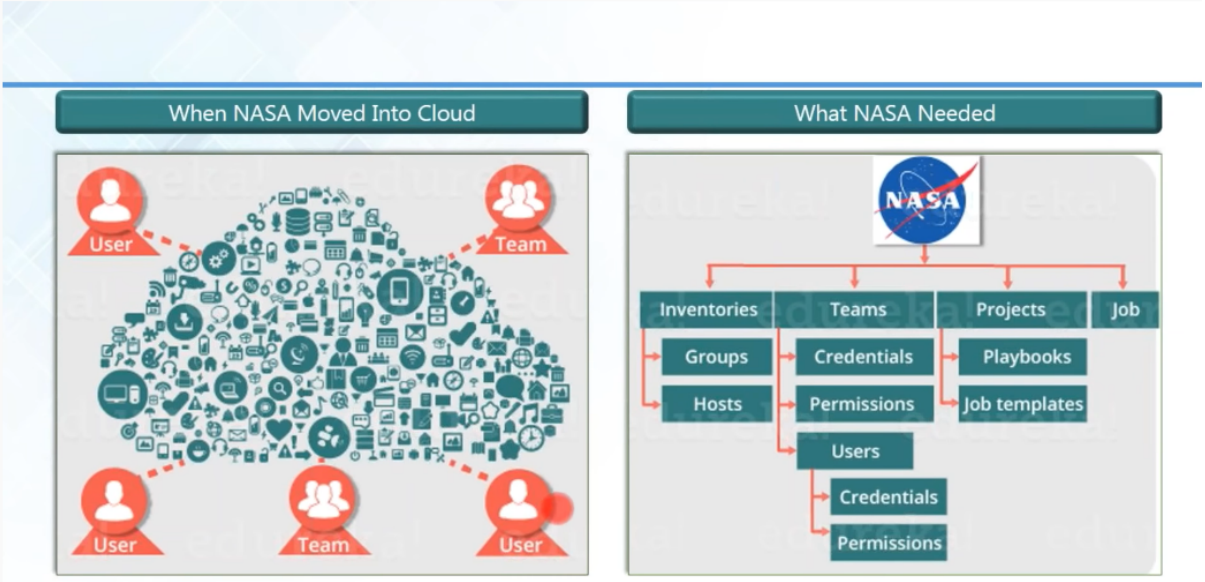




NASA needed to move 65 applications from a traditional hardware based data center to a cloud-based environment for better agility and cost savings. The rapid timeline resulted in many applications being migrated 'as it is' to a cloud environment. This created an environment which spanned multiple virtual private clouds (VPCs) and AWS accounts that could not be managed easily. Even simple things, like ensuring every system administrator had access to every server, or simple security patching, were extremely cumbersome.

The solution was to leverage Ansible Tower to manage and schedule the cloud environment.

Hence, to solve the problems that NASA had with lack of centralized management and a diverse environment, they evaluated multiple solutions and decided on an implementation of Ansible Tower. NASA is now leveraging Ansible Tower to manage their environment in a very organized and scheduled way.



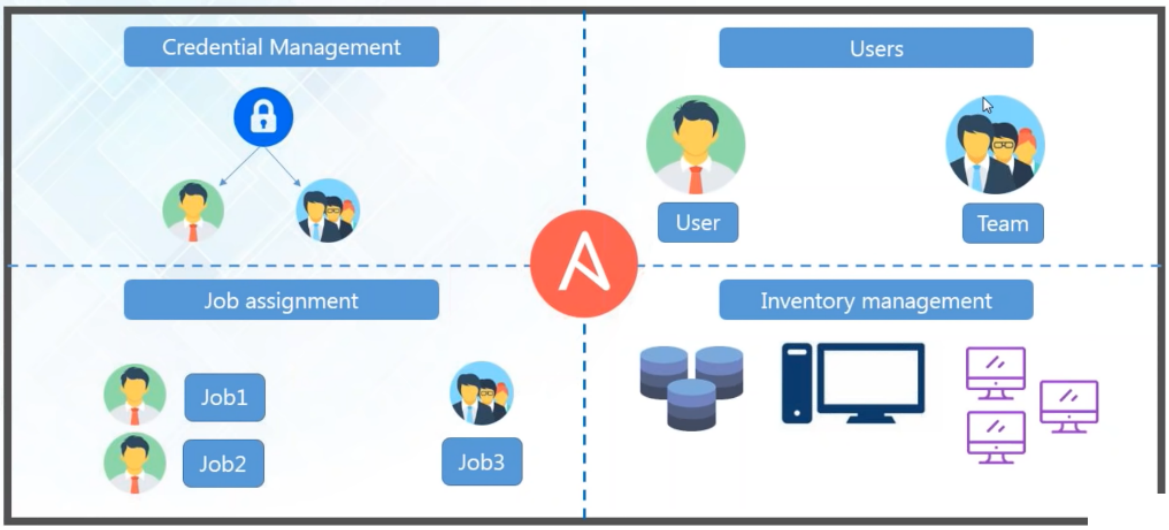
How NASA is using Ansible:

Ansible Tower provided with a dashboard which provided the status summary of all hosts and jobs which allowed NASA to group all contents and manage access permissions across different departments. It also helped to split up the organization by associating content and control permission for groups as well.

Ansible Tower is a web-based interface for managing Ansible. One of the top items in Ansible users' wish lists was an easy-to-use UI for managing quick deployments and monitoring one's configurations. Ansible management came up with Ansible Tower in response.

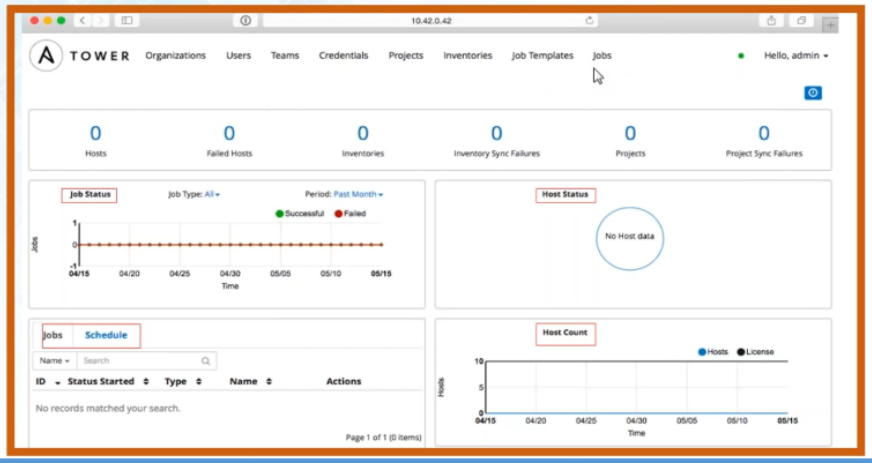
Further, Ansible divided the tasks among teams by assigning various roles. It managed the clean-up of old job history, activity streams, data marked for deletion and system tracking info. Refer to the diagram below to understand how Ansible has simplified the work of NASA.

Ansible Tower To The Rescue



Ansible Tower Dashboard

ANSIBLE TOWER by Red Hat



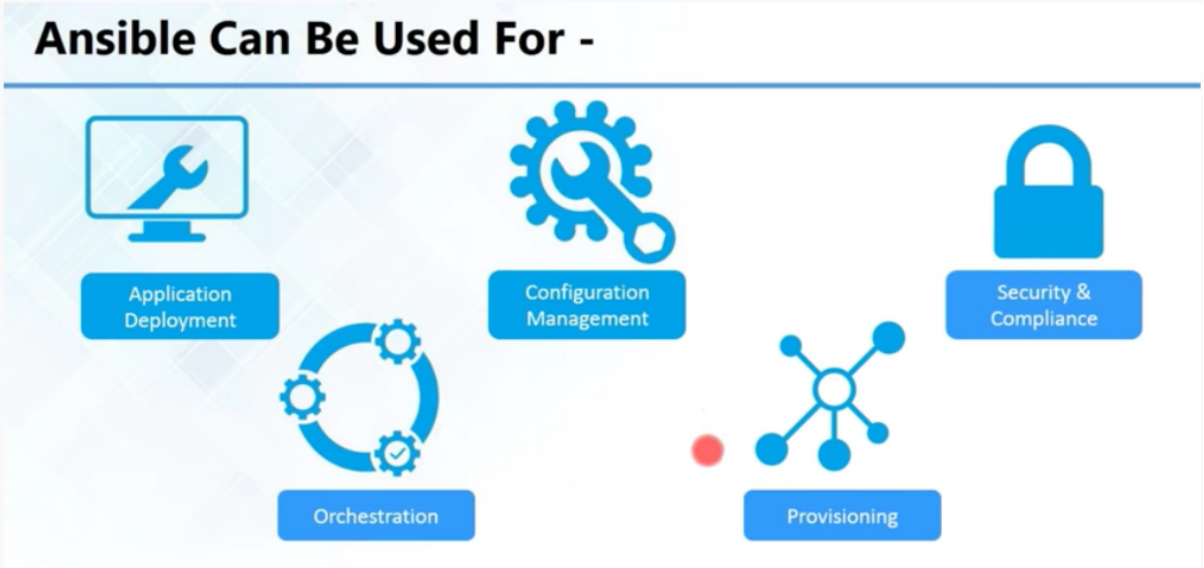
Provided status summary of all jobs

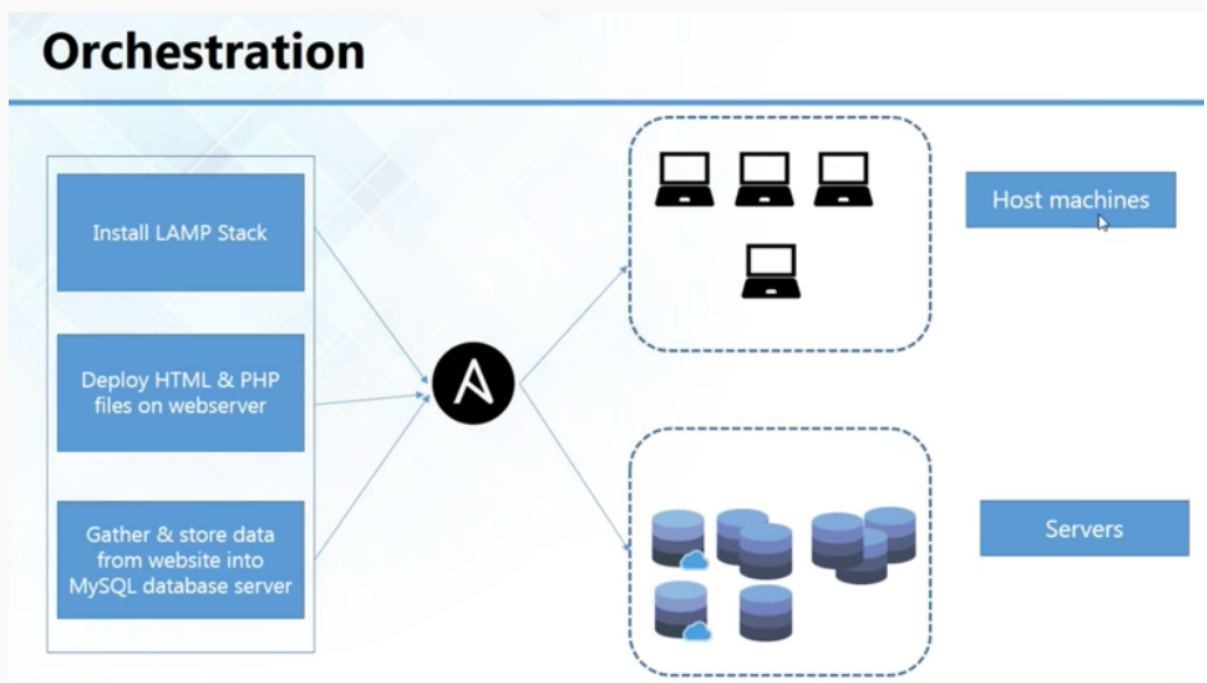
Divided organization into groups to manage access permissions

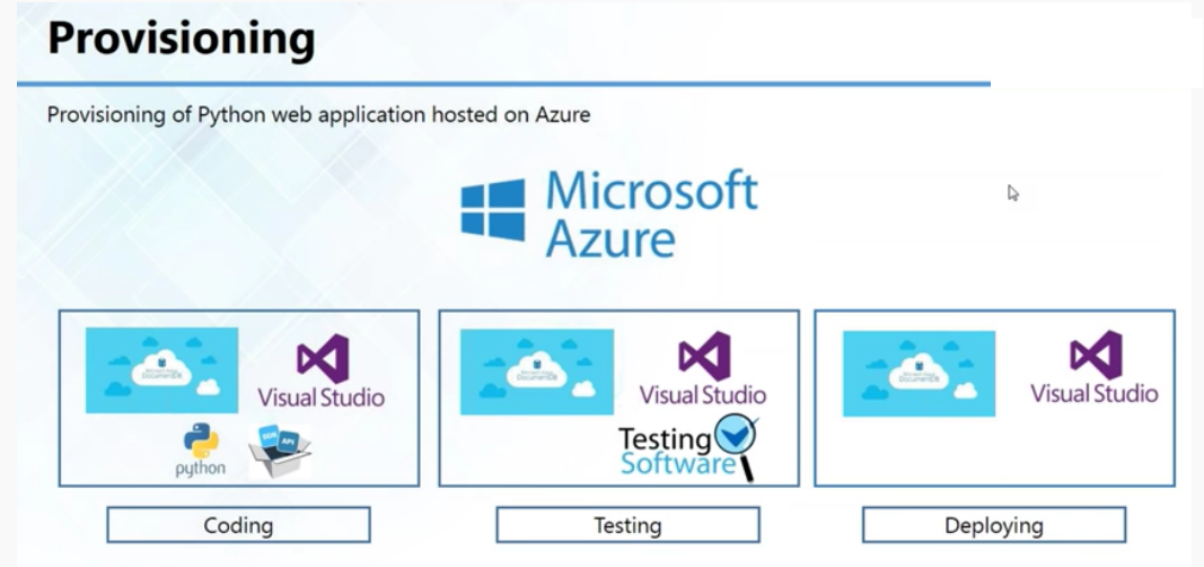
Results:

- Updating **nasa.gov** went from 1 hour to 5 minutes.
- Security patching updates went from multi-day process to 45 minutes.
- Provisioning OS accounts across entire environment in under 10 minutes.
- Application stacks set up time reduced from 1-2 hours to 10 minutes.
- Achieved near real-time RAM & disk monitoring.
- Baselining standard AMIs went from 1 hour manual process to becoming a seamless invisible background process.

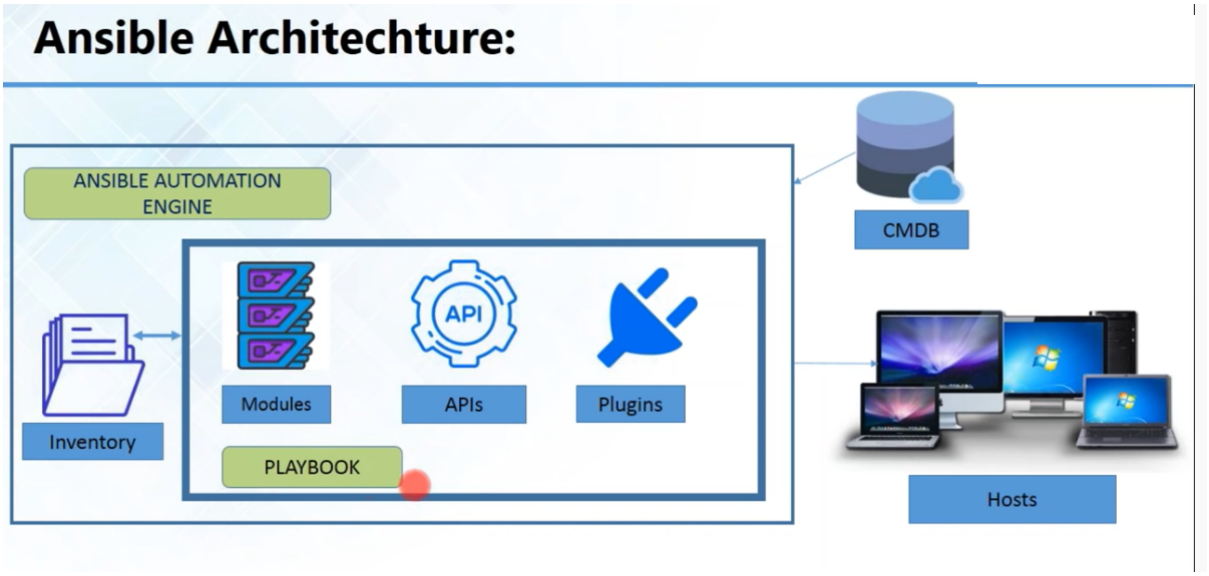
Features of Ansible:







Provisioning: Your apps must live somewhere. If you're PXE (Preboot eXecution Environment) booting and kick starting bare-metal servers or Virtual Machines, or creating virtual or cloud instances from templates**, Ansible & Ansible Tower** helps to streamline this process.



As you can see, in the diagram above, the Ansible automation engine has a direct interaction with the users who write playbooks to execute the Ansible Automation engine. It also interacts with cloud services and Configuration Management Database (CMDB).

The Ansible Automation engine consists of:

- **Inventories:** Ansible inventories are lists of hosts (nodes) along with their IP addresses, servers, databases etc. which needs to be managed. Ansible then takes action via a transport – SSH for UNIX, Linux or Networking devices and WinRM for Windows system.
- **APIs:** APIs in Ansible are used as transport for Cloud services, public or private.
- **Modules:** Modules are executed directly on remote hosts through playbooks. The modules can control system resources, like services, packages, or files (anything really), or execute system commands. There are over 450 Ansible-provided modules that automate nearly every part of your environment. For e.g.
    - Cloud Modules like *cloudformation* which creates or deletes an AWS cloud formation stack;
    - Database modules like *mssql_db* which creates/removes MYSQL databases from remote hosts.
- **Plugins:** Plugins allows to execute Ansible tasks as a job build step. Plugins are pieces of code that augment Ansible's core functionality. Ansible ships with several handy plugins, and you can easily write your own. For example,
    - *Action* plugins are front ends to modules and can execute tasks on the controller before calling the modules themselves.
    - *Cache* plugins are used to keep a cache of 'facts' to avoid costly fact-gathering operations.
    - *Callback* plugins enable you to hook into Ansible events for display or logging purposes.

There are a few more components in Ansible Architecture which are explained below:

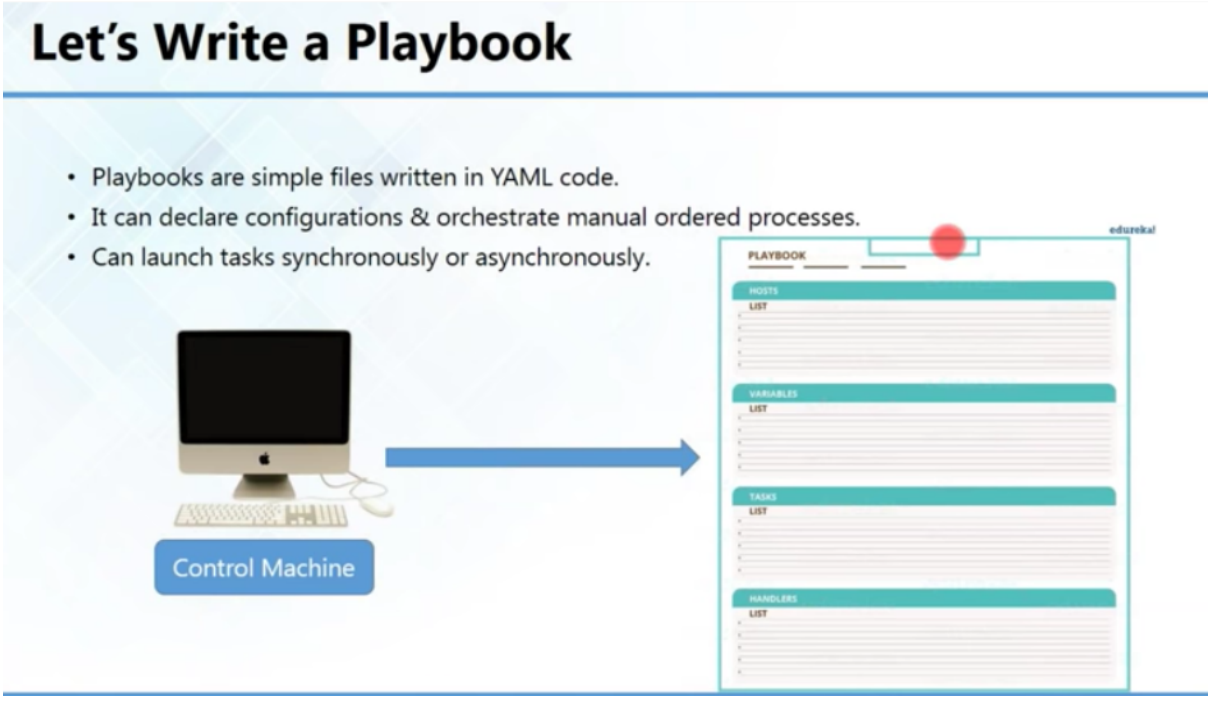
Networking: Ansible can also be used to automate different networks. Ansible uses the same simple, powerful, and the agentless automation framework IT operations and development are already using. It uses a data model (a playbook or role) that is separate from the Ansible automation engine that easily spans different network hardware.

Hosts: The hosts in the Ansible architecture are just node systems which are getting automated by Ansible. It can be any kind of machine – Windows, Linux, RedHat etc.

Playbooks: Playbooks are simple files written in YAML format which describes the tasks to be executed by Ansible. Playbooks can declare configurations, but they can also orchestrate the steps of any manual ordered process, even if it contains jump statements. They can launch tasks synchronously or asynchronously.

CMDB: It is a repository that acts as a data warehouse for IT installations. It holds data relating to a collection of IT assets (commonly referred to as configuration items (CI)), as well as to describe relationships between such assets.

Cloud: It is a network of remote servers hosted on the Internet to store, manage, and process data, rather than a local server. You can launch your resources and instances on cloud and connect to your servers.



Ansible:

YAML Is a Data Serialization Language. You will be writing your playbooks in control machines. Playbook starts with 3 dashes on the top.

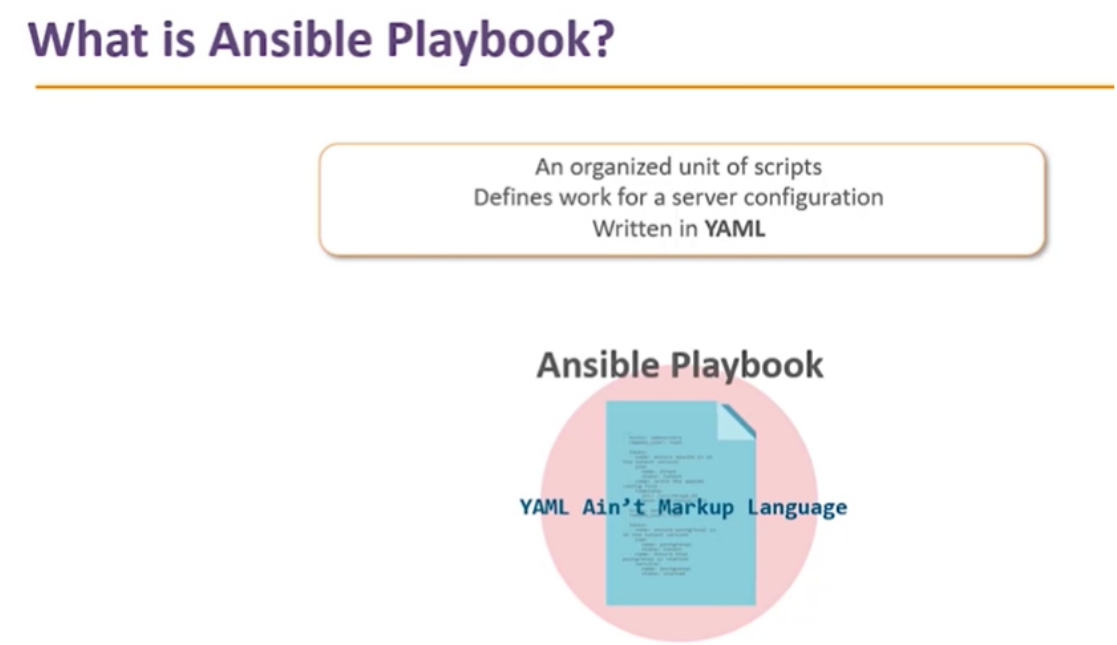
Hosts – List of your host machines, where do you want this play book to run.

Variables – to store the facts

Tasks - Get executed in the same order..ex: install software a,b,c etc.

Handles – Triggers or conditions.

Tasks will be executed in the same order how you write them.



Example :

--- (Three Dashes on the TOP)

- hosts: webservers

- tasks:

  -name: install apache2

    apt: name=apache2 update-cache=yes state=latest

(Here I am using the apt module to download the package so this is the syntax of writing the apt module to give the name of the package which is Apache to update cache is equal to yes it means that it'll make sure that apt-get is already updated in your node machine before it installs the Apache - and in mentioned state equal to latest it means that it will download the latest version.)

notify:

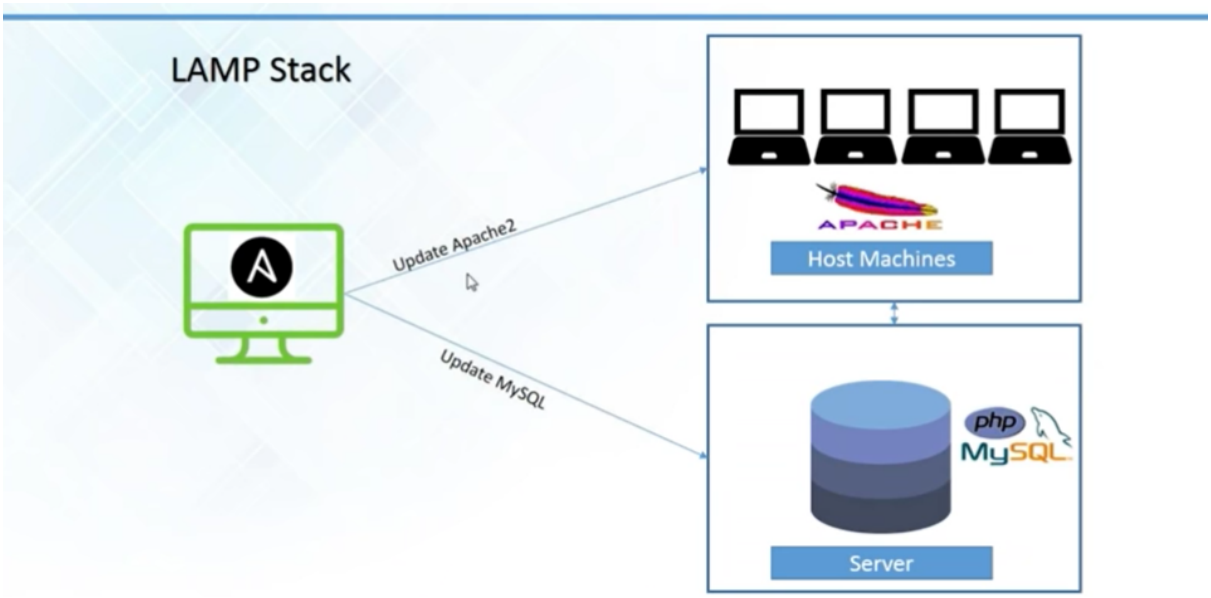
-restart apache2

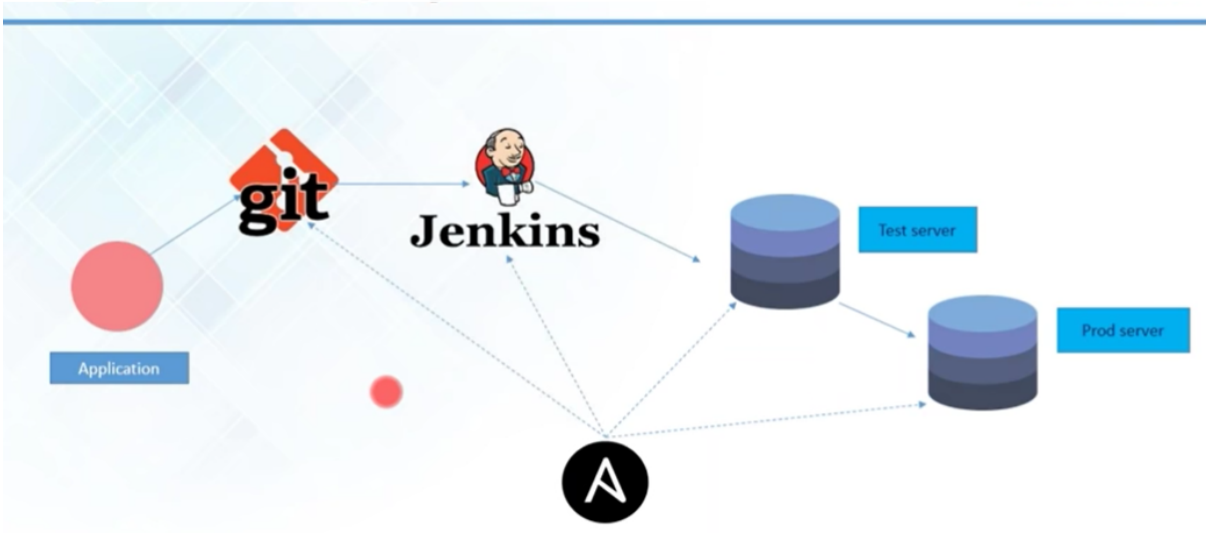
Handlers

-name : restart apache2

service.name = apache2 state = restarted

https://www.bogotobogo.com/DevOps/Ansible/Ansible-Handlers.php

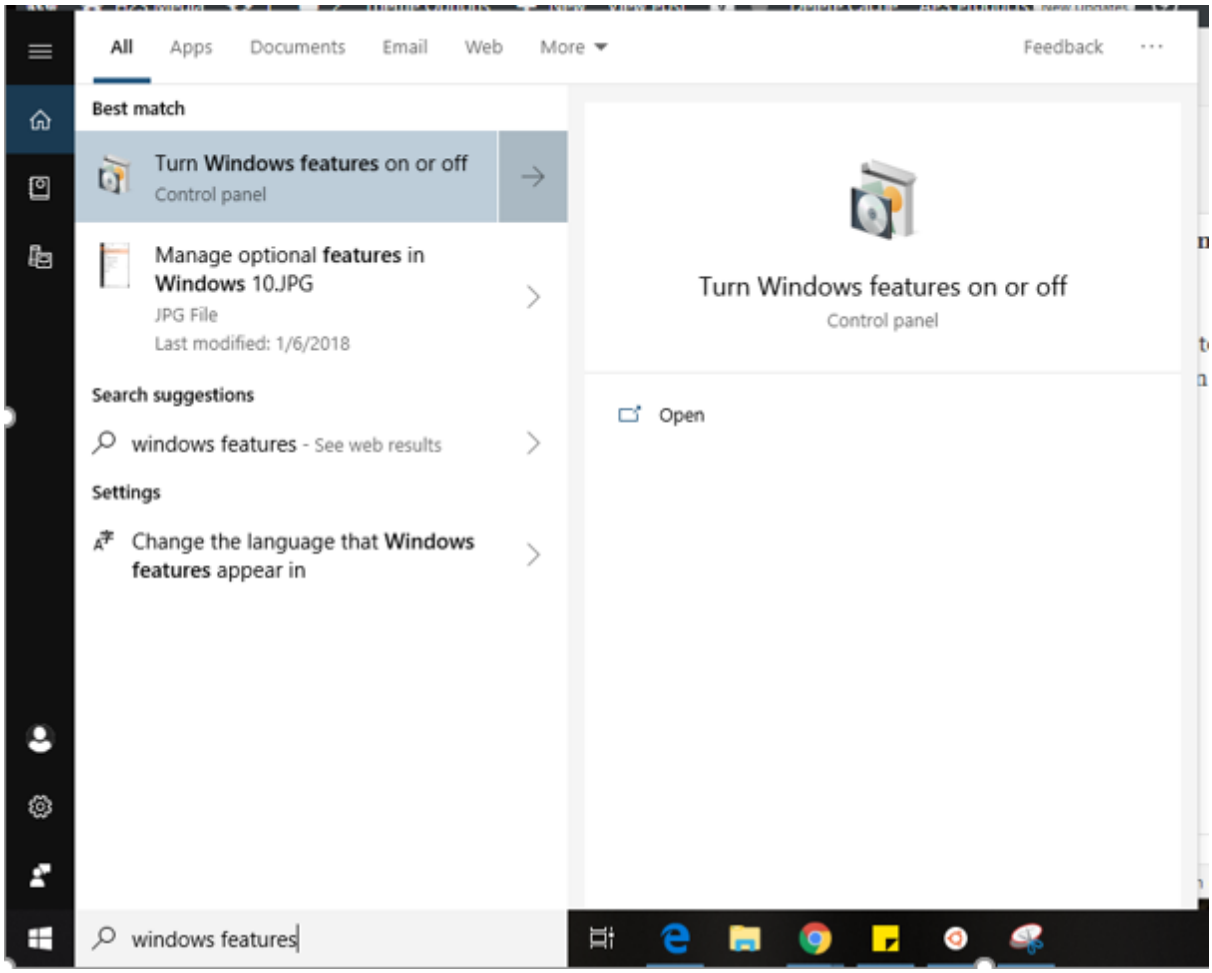




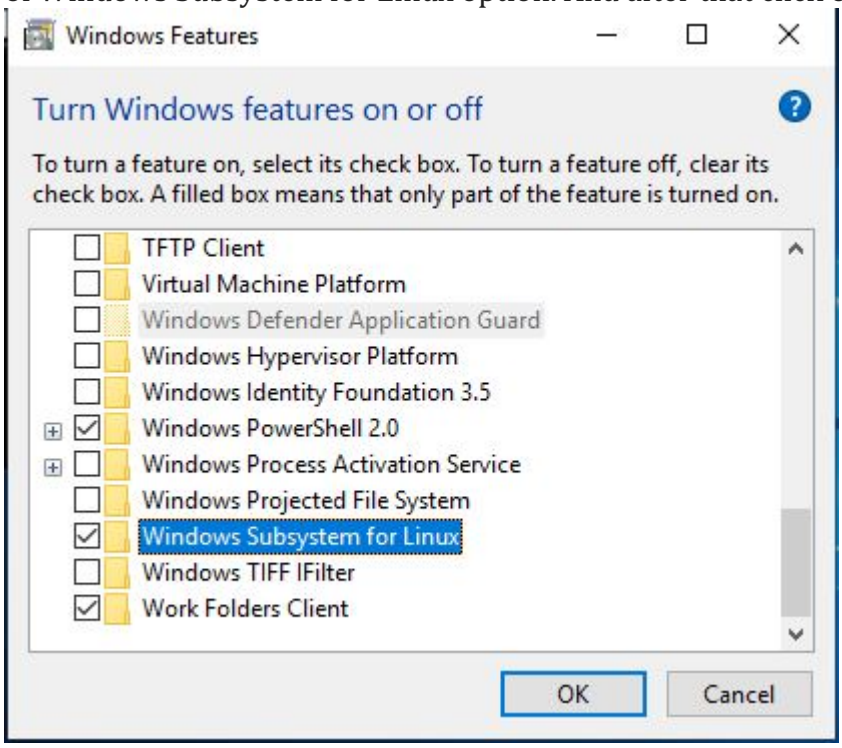
Reference Project : https://www.howtoforge.com/ansible-guide-create-ansible-playbook-for-lemp-stack/

Install Ansible on Windows 10

- Open the Window's **Turn Windows features on or off** section.
- Select the **Windows Subsystem for Linux** to activate it.
- Go to the **Microsoft app store.**
- Search for **Linux**.
- Multiple Linux system will appear like **Debian, Ubuntu, OpenSuse**
- Select the **Ubuntu** or any other Linux you want to install the Ansible. Here we are using **Ubuntu 18.04**
- Once the **Ubuntu installed on Windows 10**, it will ask you to create the user.
- Now add the **Ansible PPA repo** on Windows 10's Ubuntu app.
- Use installation commands to **install the Ansible on Windows 10 Linux** subsystem.
- Now you can run Ansible on Windows to perform different management and automation tasks.
  - **Step 1: Turn Windows features on or off**
  - Basically, this features already on the Windows 10 and we just need to turn it on from the features option. For that just search for **Windows features** in the Search box. And when the "**Turn Windows features on or off** " appears click on that.

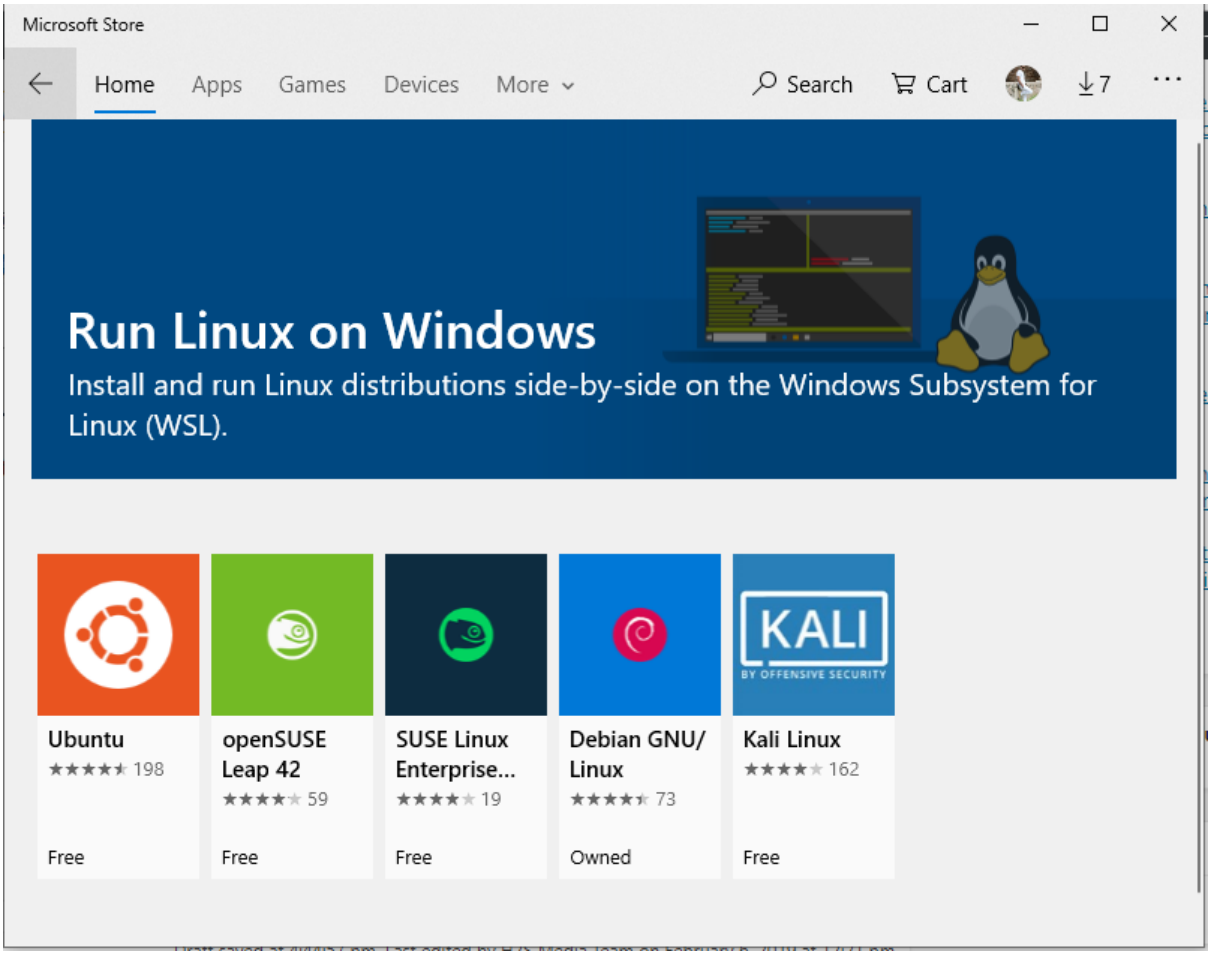


- **Step 2: Install the Windows SubSystem for Linux**
- Now a window will open with a bunch of features. Scroll down and check the box of Windows Subsystem for Linux option. And after that click on the **OK** button.



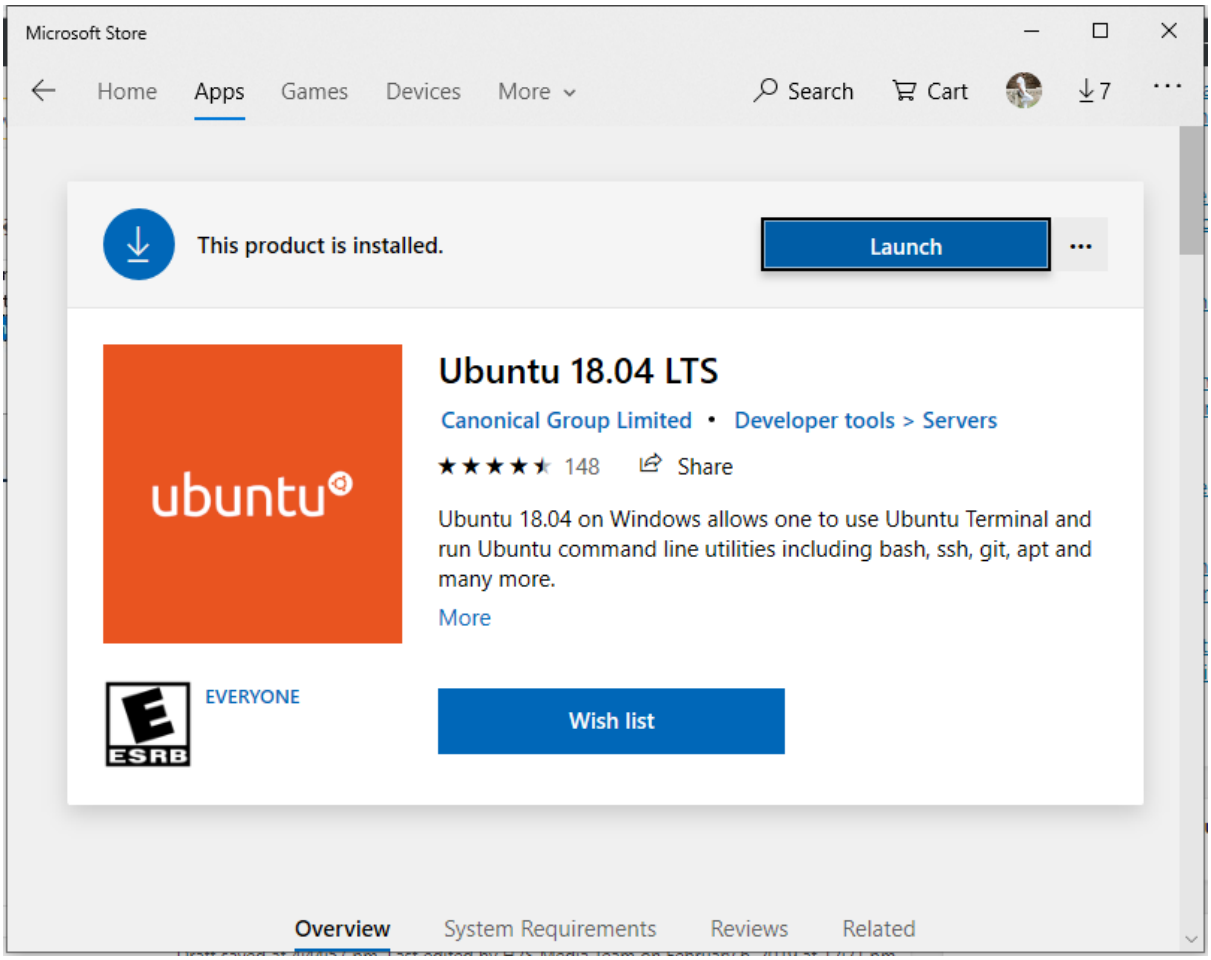
Step 3: Open the Microsoft Store

To open the App store of Microsoft for Windows click on the search box and type **Microsoft store**. The moment it will appear, click on that.



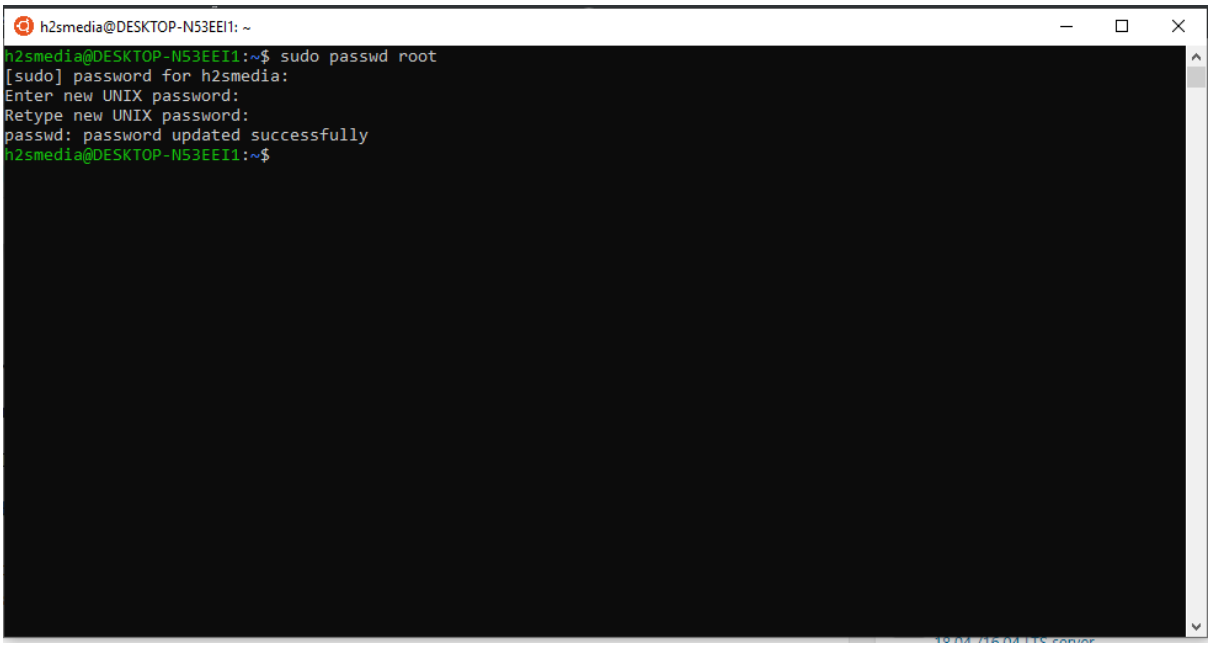
Step 4: Install Ubuntu 18.04 on Windows 10

Now search for the Ubuntu and install the latest version which is 18.04. As soon as it appears, you will see **GET** button, click on that and install it on your Windows 10. After the installation, you will see a **launch** button, use that to open the **Ubuntu Bash**. You can install any other Linux system such as Debian, Kali Linux, and OpenSuSE to install the Ansible.



Step 5: Set the credentials

When the Ubuntu Bash opens, it will ask you to set the username and password for default user of your Ubuntu on Windows. We can also set the root account password from here. Just type ***sudo passwd root***



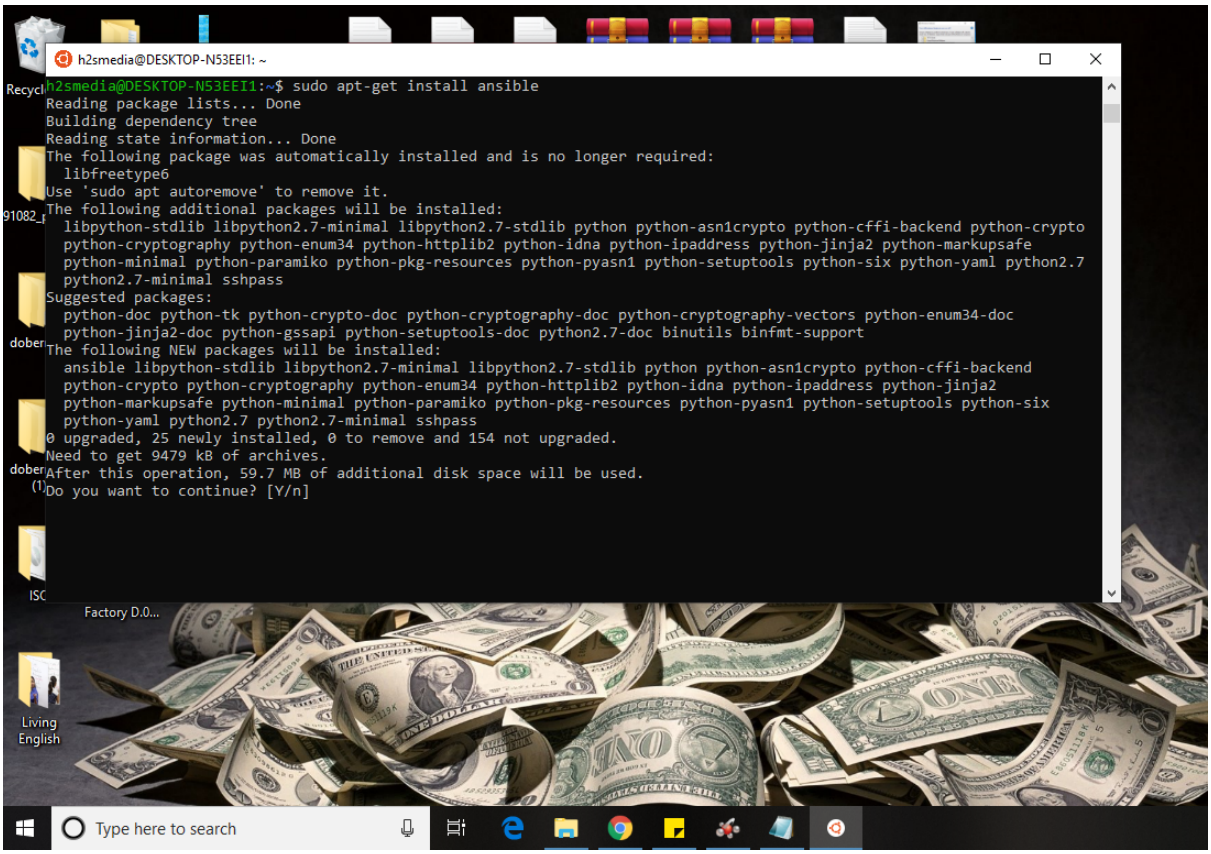
Step 6: Install Ansible on Windows

For installing and running Ansible on Windows just use the below-given commands:

```
sudo apt-get update

sudo apt-get install software-properties-common

sudo apt-add-repository ppa:ansible/ansible

sudo apt-get update

sudo apt-get install ansible
```

Note: For older Ubuntu versions such as Ubuntu 14.04, 15.04, and 16.04, we need to add the repo of Ansible but the latest version such as Ubuntu 18.04 can get the Ansible installation files directly from via its package management.

Press **Y** when it asks for…



Testing about the installation.

```
sgrbala@DESKTOP-C0FVCIJ:~$ ansible --version
ansible 2.9.2
  config file = /etc/ansible/ansible.cfg
  configured module search path = [u'/home/sgrbala/.ansible/plugins/modules', u'/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python2.7/dist-packages/ansible
  executable location = /usr/bin/ansible
  python version = 2.7.15+ (default, Oct  7 2019, 17:39:04) [GCC 7.4.0]
sgrbala@DESKTOP-C0FVCIJ:~$
```

Note :

You will have to give the following commands for installing ansible.

>>sudo apt-add-repository ppa:ansible/ansible

>>sudo apt-get install ansible

sudo apt-add-repository ppa:ansible/ansible