**Linux**

1. Linux is a free to use.
2. Every thing will be in the form of files
3. Kernel is the key layer of the linux operating system
4. Every thing will be stored in the form of files
5. Shell will be act as a interface between the User and the kernel.
6. Shell is an interpreter and it just used to displays output.
7. The reason why we are using linux compared to windows is the no of files we download the speed the operating system will be reduced.

**AWS**

1. It is a cloud platform for launching an instance or compute instance.
2. Having file-level granularity for security is very use full
3. Auto archive
4. Storage of capability with offering varitys of DBS
5. Security and highly encrypted.
6. The price of the features like instances, S3, Elastic IPs etc is not to high.
7. UPTIME
8. Conduct systems design, feasibility and cost studies and recommend cost-effective cloud solutions such as Amazon Web Services.
9. Created scripts in Python which integrated with Amazon API to control instance operations.
10. Launching Amazon EC2 cloud instances using Amazon Machine Images for AWS cloud.
11. Create, manage, and delete users and groups as per the request using Amazon Identity and Access Management.

The reason why we use AWS is compared to AZURE is

1. Auto delete archive is available is amazon where we have to purchase in Azure
2. Auto delete available
3. Encrypt data is smart feature in amazon it is still in progress in other cloud platforms
4. Reduced redununcy will be in the form of file level
5. Max size in Blob 5TB other supports only 1TB
6. Security will be launched with file level not in bucket level.

**Chef**

1. Writing Opscode CHEF Cookbooks to develop a continuous deployment pipeline for a wide range of Linux applications that run in LXD Containers, Virtual and Bare metal servers.
2. Develop Opscode Chef cookbooks and recipes to automate installation of COTS, FOSS, and system hardening on GPS OCX systems
3. Update existing Chef cookbook formatting to follow style guide and include idempotency
4. Installed Chef-Server Enterprise On-Premise/WorkStation/ Bootstrapped the Nodes using Knife.
5. Worked with Chef Enterprise Hosted as well as On-Premise, Installed Workstation, Bootstrapped Nodes.
6. Wrote Recipes and Cookbooks and uploaded them to Chef-server, Managed On-site OS/Applications/Services/Packages using Chef as well as AWS for EC2/S3/Route53 & ELB with Chef Cookbooks.
7. Wrote Chefspecs and Server specs for the unit testing and integration testing.
8. Improve infrastructure through the devDeploy Splunk server and work on automation code to build the server in the cloud using Chef.elopment of automation software modules known as OpscodeChef Cookbooks.
9. Create Chef Cookbooks and Recipes to maintain and automate various parts of infrastructure.
10. Create Chef coding best practices for existing development team
11. Refactor existing Opscode Chef Automation code.
12. Test Chef Cookbook modifications on cloud instances in AWS and using Test Kitchen and ChefSpec.
13. Create documentation for Chef best practices to be used by developers as a guide to Chef

**Ansible**

1. It’s a simple automation language that can perfectly describe an IT application infrastructure in in ansible playbooks
2. Its an automation engine that runs ansible playbooks
3. Ansible Tower by red hat is an enterprise framework for controlling securing and managing your ansible automation with a UI and restful API.
4. **Its is simple human readable automation, no special coding skills needed, it is powerful and agentless**
5. Implemented and designed AWS virtual servers by Ansible roles to ensure deployment of web applications.
6. Creating S3 buckets and setting permissions for buckets using Ansible roles and playbooks. UsingS3 storage to make fetch important documents during instance creation and Web development.
7. Used Ansible for creating subnets, security groups, route tables and ACL's for VPC creation and deploying EC2 instances on AWS.
8. Generated client/server certificates using Bulk API and copied the certificates on all the DEV hosts, wrote an ansible script for the same.
9. Involved in developing custom scripts using Python which act as backend for the Ansible scripts to get deployed in the client hosts through Infrared

**Shell**

1. Used Shell/Perl scripts to automate the deployment process
2. Responsible for writing/modifying scripts using Shell, K-Shell and bash for day-to-day administration
3. Used Shell to automate the deployment process.
4. Used Shell and Perl scripting to deploy artifacts that are built by Maven
5. Involved in writing and organizing Shell and Perl scripting for building complex software systems.
6. Developed shell scripts for automation of the build and release process, developed Custom Scripts to monitor repositories, Server storage.
7. Created and managed shell scripts for the automation of build and deployment process
8. Created and maintained the Shell deployment scripts for web application servers.
9. Developed Python and shell scripts for automation of the build and release process.
10. Writing scripts in Bash shell for optimizing day to day administration

**Cloud formation**

1. Contiuous integration tool Jenkins was used to build the project and has been installed using cloud formation templates
2. Implemented app logging service using logging tools, such as Splunk, Graylog2 and developed and deployed stacks using AWS Cloud Formation templates
3. Involved completely on migrating on-perm applications into Cloud using Cloud formation templates with high security by adjusting security groups flowing inbound and outbound both on Windows and Linux instances.
4. Deployed and configured MYSQL RDS Instance on AWS using Cloud formation.
5. Deployed EC2 instances, attached security groups, attached profiles and roles using AWS Cloud Formation.
6. Utilize Cloud formation and Puppet by creating DevOps processes for consistent and reliable deployment methodology.
7. Automated infrastructure deployment process using Cloud Formation and Terraform remove human error and mitigate concerns with scalability of infrastructure.
8. Automated the cloud deployements using Ansible and AWS Cloud Formation Templates
9. Involved in designing and deploying multiple applications utilizing almost all of the AWS stack (Including EC2, Route53, S3, RDS, Dynamo DB, SNS, SQS, IAM) focusing on high-availability, fault tolerance, and auto-scaling in AWS Cloud Formation.
10. Setting up ec2 stack using Cloud Formation as well as with Terraform scripts.

**Cloud Watch**

1. Configured Cloud Watch and DataDog to monitor real-time granular metrics of all the AWS Services and configured individual dashboards for each resource Agents
2. Created alarms in Cloud Watch service for monitoring the server's performance, CPU Utilization, disk usage etc.
3. Worked on AWS Cloud Watch, Cloud Formation, Cloud Trail services and Cloud Front to set up and manage cached content delivery
4. Created monitors, alarms and notifications for EC2 hosts using Cloud Watch Monitored System Performance Managed Disk Space LVM (Logical Volume Manger) and performed system Backup and Recovery
5. Performed application server builds in EC2 environment and monitoring them using cloud watch
6. Implemented and maintained the monitoring and alerting of production and corporate servers/storage using AWS Cloud watch
7. Worked on AWS cloud watch for monitoring the application infrastructure and used AWS email services for notifying.
8. Created monitors, alarms and notifications for AWS EC2 hosts using Cloud Watch.
9. Install, configure and administer log analyzer tool Cloud Watch Automated the release pipeline to achieve zero touch deployments using Jenkins, Hudson.
10. Experience in creating alarms and notifications for EC2 instances using Cloud Watch.

**Nagios**

1. Implemented and configured Nagios for continuous monitoring of applications in the production environment and enabled notifications via emails and text messages.
2. Used Nagios as a monitoring tool to identify and resolve infrastructure problems before they affect critical processes and also worked on Nagios Event handlers in case of automatic restart of failed applications and services.
3. Wrote Nagios Bash Plugins for monitoring the custom processes, monitoring custom disk spaces in Windows and Linux. Updated monitoring parameter in Nagios with active and passive check.
4. Integrated Nagios SMTP mailbox with Mule ESB to send the alerts to Cloud Management Platform through API.
5. Working on Configuration of Nagios for automatic system performance alerts.
6. Worked on programming using python and knowledge of Java.
7. Implement new and maintain old monitoring systems (Nagios).
8. Write new and fix or rewrite old Nagios plug-ins (Bash and Perl) for production applications and databases.
9. Based on projects requirements, need to Setup new AWS instances and configured web/database server and configure daily status mails and adding it to nagios monitoring.
10. Adding new agents to OSSEC server and nagios server for monitoring.
11. Configured Nagios Monitoring server with different Linux services such as HTTP, SMTP, Disk space, CPU usage, Load Average etc.
12. Monitoring of All UNIX/Linux flavor like Linux and windows through nagios.

**Webserver**

1. Maintained IIS webserver during deployments and production for .net applications
2. Configured Apache webserver to proxy requests to Tomcat 6.0.x/7.0.x.
3. Configured Iplanet webserver to proxy requests to WebLogic Application Server 10.x.
4. Providing deploy support for Sunone 7 Iplanet webserver.
5. Configured IIS/Apache webserver to proxy requests to WebLogic Application Server 10.x.
6. Installed and configured Weblogic Server 10.3/11g and Sun one iplanet webserver
7. Installed and configured Apache 2.2 WebServer and Oracle10i database to work with WebLogic Server 10.x/11g.
8. Deployed the static code in to front end apache webserver.
9. Created and Deployed the running instances of the Webservers, NFS Servers and Database Servers Using Ansible as a Configuration Management tool.
10. Created Amazon VPC to create public-facing subnet for webservers with internet access, and backend databases & application servers in a private-facing subnet with no Internet access.
11. Created Amazon VPC to create public-facing subnet for webservers with internet access, and backend databases & application servers in a private-facing subnet with no Internet access.

**Git**

1. Responsible for design and maintenance of the GIT repositories and the access control strategies.
2. Created the branches in GIT to implement the parallel development process.
3. Used GIT for branching and merging.
4. Integrated JIRA and GIT with Pre-commit hooks and also involved in setting up JIRA as defect tracking system alongside configured various workflows, customizations, and plugins for the JIRA bug/issue tracker.
5. Installed and configured GIT and communicating with the repositories in GITHUB.
6. Used the version control system GIT to access the repositories and used in coordinating with CI tools.
7. Integrated maven with GIT to manage and deploy project related tags.
8. Performed necessary day to day Subversion/GIT support for different projects.
9. Experience migrating SVN repositories to GIT.
10. Worked in Git implementation containing various Remote repositories for a single application.
11. Maintained GIT source code repository and local mirrors; perform branching, tagging, merging and maintenance tasks for windows host and Mac builds.
12. Connected continuous integration system with GIT version control repository and continually build as the check-in's come from the developer.
13. Installed and Administered on GIT Server, migrated Projects from Subversion to GIT.
14. Worked with GitHub to manage source code repositories and performed branching, merging and tagging depending on requirement

**JENKINS**

1. Implemented Continuous Integration using Jenkins.
2. Extensively worked on Jenkins Freestyle and pipeline jobs for end-to-end automation to build, test and deliver artifacts.
3. Created docker slave container to run integration testing. This was later added as Jenkins pipeline.
4. Setup Nexus repository to store artifacts from Jenkins server using maven.
5. Continuous improvement in integration workflow, project testing, and implementation of continuous integration pipeline with Jenkins. Implemented code coverage and unit test plug-ins with Maven in Jenkins.
6. Used Jenkins, Build forge for Continuous Integration and deployment into Tomcat Application Server.
7. Installing, configuring and administering Jenkins CI tool on Linux machines.
8. Implemented the setup for Master slave architecture to improve the Performance of Jenkins.
9. Used Jenkins for enterprise scale infrastructure configuration and application deployments.
10. Profoundly did Continuous integration using Hudson's/Jenkins, Anthill pro, Team city and Bamboo, which requires developers to integrate code into a shared repository several times a day.
11. Built end to end CI/CD Pipelines in Jenkins to retrieve code, compile applications, perform tests and push build artifacts to Nexus Artifactory.
12. Created job chains with Jenkins Job Builder, Parameterized Triggers, and target host deployments. Utilized many Jenkins plugins and Jenkins API.
13. Used Jenkins for enterprise scale infrastructure configuration and application deployments into Artifactory.
14. Provided backup support for Jenkins build environment.
15. Adopted Jenkins for CI practice and revised existing build procedures. Created multiple Maven, Shell scripts for build automation and deployment.
16. Worked on configuring the Jenkins to use MetaCase Software to build Java code and also to do the whole C.I process on the java code generated by MetaCase.