TOP 30

DevOps

INTERVIE QUESTION

Created by- Brij Mohan

Q 1. Who is a DevOps engineer?

Ans : A DevOps engineer is a person who works with both software developers and the IT staff to ensure smooth code releases.

They are generally developers who develop an interest in the deployment and operations domain or the system admins who develop a passion for coding to move towards the development side.

In short, a DevOps engineer is someone who has an understanding of SDLC (Software Development Lifecycle) and of automation tools for developing CI/CD pipelines.

Q 2. Why DevOps has become famous?

Ans : These days, the market window of products has reduced drastically. We see new products almost daily. This provides a myriad of choices to consumers but it comes at a cost of heavy competition in the market. Organizations can not afford to release big features after a gap.

They tend to ship off small features as releases to the customers at regular intervals so that their products don't get lost in this sea of competition.

Customer satisfaction is now a motto to the organizations which has also become the goal of any product for its success. In order to achieve this, companies need to do the below things:

- Frequent feature deployments
- Reduce time between bug fixes
- Reduce failure rate of releases
- Quicker recovery time in case of release failures.

In order to achieve the above points and thereby achieving seamless product delivery, DevOps culture acts as a very useful tool.

Due to these advantages, multi-national companies like Amazon and Google have adopted the methodology which has resulted in their increased performance.

Q 3. What is the use of SSH?

Ans : SSH stands for Secure Shell and is an administrative protocol that lets users have access and control the remote servers over the Internet to work using the command line.

SSH is a secured encrypted version of the previously known Telnet which was unencrypted and not secure. This ensured that the communication with the remote server occurs in an encrypted form.

SSH also has a mechanism for remote user authentication, input communication between the client and the host, and sending the output back to the client.

Q 4. What is configuration management?

Ans : Configuration management (CM) is basically a practice of systematic handling of the changes in such a way that system does not lose its integrity over a period of time.

This involves certain policies, techniques, procedures, and tools for evaluating change proposals, managing them, and tracking their progress along with maintaining appropriate documentation for the same.

CM helps in providing administrative and technical directions to the design and development of the appreciation.

The following diagram gives a brief idea about what CM is all about:

Q 5. What is the importance of having configuration management in DevOps?

Ans : Configuration management (CM) helps the team in the automation of time-consuming and tedious tasks thereby enhancing the organization ' s performance and agility.

It also helps in bringing consistency and improving the product development process by employing means of design streamlining, extensive documentation, control, and change implementation during various phases/releases of the project.

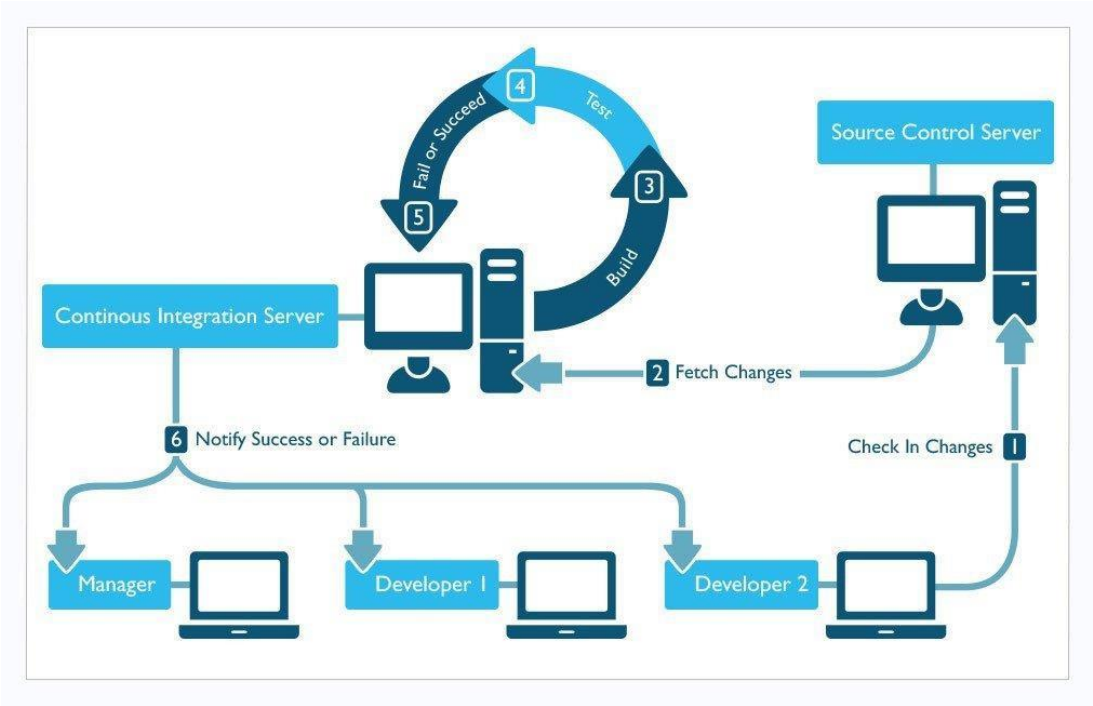
Q 6. What does CAMS stand for in DevOps?

Ans : CAMS stands for Culture, Automation, Measurement, and Sharing. It represents the core deeds of DevOps.

Q 7. What is Continuous Integration (CI)?

Ans : Continuous Integration (CI) is a software development practice that makes sure developers integrate their code into a shared repository as and when they are done working on the feature.

Each integration is verified by means of an automated build process that allows teams to detect problems in their code at a very early stage rather than finding them after the deployment.



Based on the above flow, we can have a brief overview of the CI process.

- Developers regularly check out code into their local workspaces and work on the features assigned to them.
- Once they are done working on it, the code is committed and pushed to the remote shared repository which is handled by making use of effective version control tools like git.
- The CI server keeps track of the changes done to the shared repository and it pulls the changes as soon as it detects them.
- The CI server then triggers the build of the code and runs unit and integration test cases if set up.
- The team is informed of the build results. In case of the build failure, the team has to work on fixing the issue as early as possible, and then the process repeats.

Q 8. Why is Continuous Integration needed?

Ans : By incorporating Continuous Integration for both development and testing, it has been found that the software quality has improved and the time taken for delivering the features of the software has drastically reduced.

This also allows the development team to detect and fix errors at the initial stage as each and every commit to the shared repository is built automatically and run against the unit and integration test cases.

Q 9. What is Continuous Testing (CT)?

Ans : Continuous Testing (CT) is that phase of DevOps which involves the process of running the automated test cases as part of an automated software delivery pipeline with the sole aim of getting immediate feedback regarding the quality and validation of business risks associated with the automated build of code developed by the developers.

Using this phase will help the team to test each build continuously (as soon as the code developed is pushed) thereby giving the dev teams a chance to get instant feedback on their work and ensuring that these problems don ' t arrive in the later stages of SDLC cycle.

Doing this would drastically speed up the workflow followed by the developer to develop the project due to the lack of manual intervention steps to rebuild the project and run the automated test cases every time the changes are made.

Q 10. What are the three important DevOps KPIs?

Ans : Few KPIs of DevOps are given below:

- Reduce the average time taken to recover from a failure.
- Increase Deployment frequency in which the deployment occurs.
- Reduced Percentage of failed deployments.

Q 11. Explain the different phases in DevOps methodology.

Ans : DevOps mainly has 6 phases and they are:

➢ Planning:

- This is the first phase of a DevOps lifecycle that involves a thorough understanding of the project to ultimately develop the best product.
- When done properly, this phase gives various inputs required for the development and operations phases.
- This phase also helps the organization to gain clarity regarding the project development and management process.
- Tools like Google Apps, Asana, Microsoft teams, etc are used for this purpose.

➢ Development:

- The planning phase is followed by the Development phase where the project is built by developing system infrastructure, developing features by writing codes, and then defining test cases and the automation process.
- Developers store their codes in a code manager called remote repository which aids in team collaboration by allowing view, modification, and versioning of the code.
- Tools like git, IDEs like the eclipse, IntelliJ, and technological stacks like Node, Java, etc are used.

➢ Continuous Integration (CI):

- This phase allows for automation of code validation, build, and testing. This ensures that the changes are made properly without development environment errors and also allows the identification of errors at an initial stage.
- Tools like Jenkins, circleCI, etc are used here.

➢ Deployment:

- DevOps aids in the deployment automation process by making use of tools and scripts which has the final goal of automating the process by means of feature activation.
- Here, cloud services can be used as a force that assists in upgrade from finite infrastructure management to cost-optimized management with the potential to infinite resources.
- Tools like Microsoft Azure, Amazon Web Services, Heroku, etc are used.

➢ Operations:

- This phase usually occurs throughout the lifecycle of the product/software due to the dynamic infrastructural changes.
- This provides the team with opportunities for increasing the availability, scalability, and effective transformation of the product.
- Tools like Loggly, BlueJeans, Appdynamics, etc are used commonly in this phase.

➢ Monitoring:

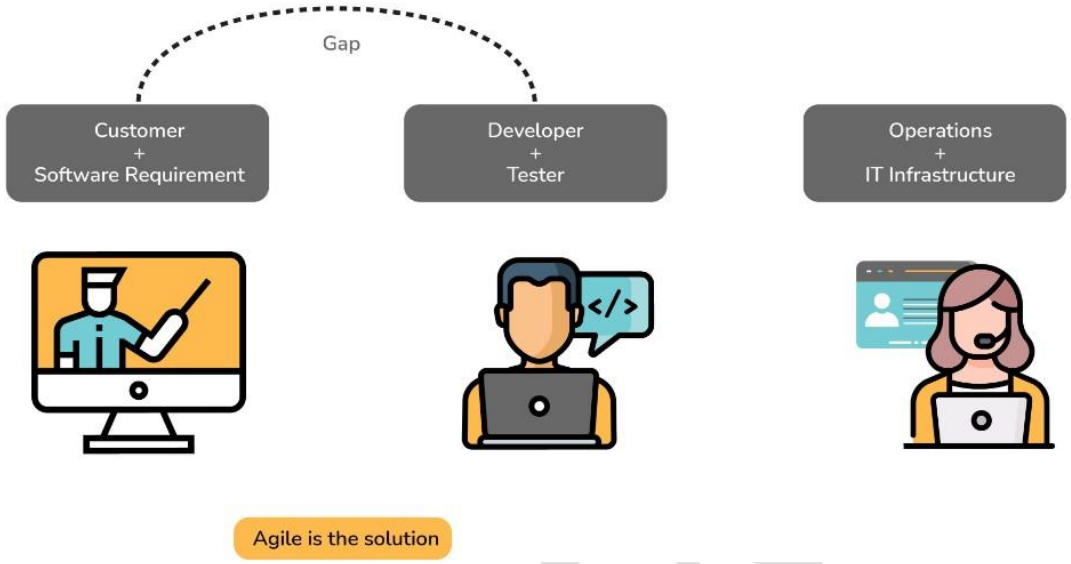
- Monitoring is a permanent phase of DevOps methodology. This phase is used for monitoring and analyzing information to know the status of software applications.
- Tools like Nagios, Splunk, etc are commonly used.

Q 12. How is DevOps different than the Agile Methodology?

Ans : DevOps is a practice or a culture that allows the collaboration of the development team and the operations team to come together for successful product development. This involves making use of practices like continuous development, integration, testing, deployment, and monitoring of the SDLC cycle.

DevOps tries to reduce the gap between the developers and the operations team for the effective launch of the product.

Agile is nothing but a software development methodology that focuses on incremental, iterative, and rapid releases of software features by involving the customer by means of feedback. This methodology removes the gap between the requirement understanding of the clients and the developer

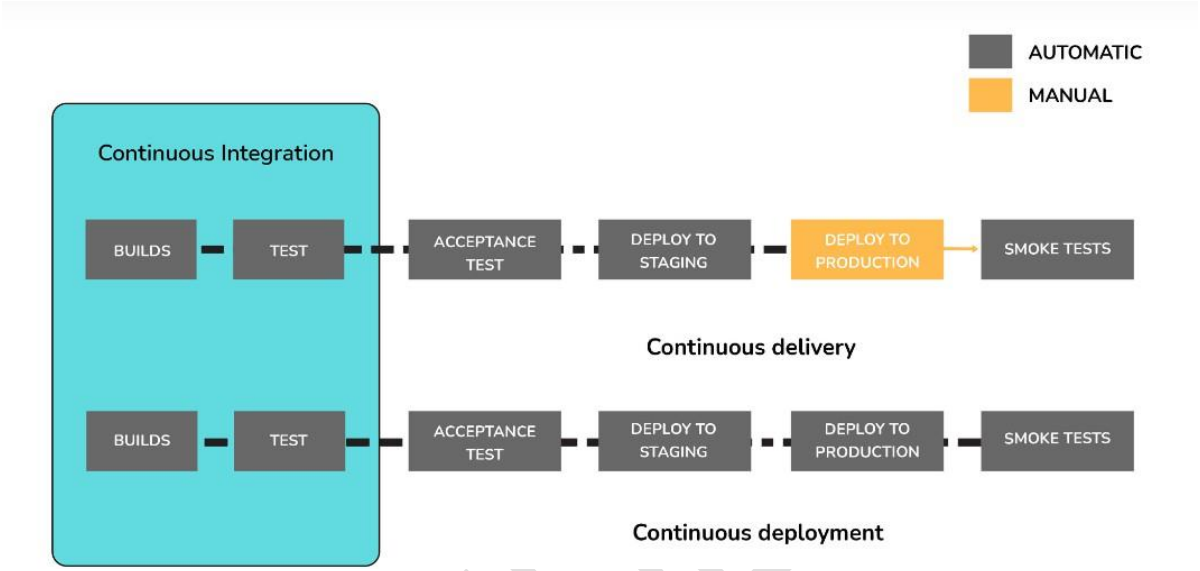


Q13. Differentiate between Continuous Deployment and Continuous Delivery?

Ans : The main difference between Continuous Deployment and Continuous Delivery are given below:

Continuous Deployment	Continuous Delivery
The deployment to the production environment is fully automated and does not require manual/ human intervention.	In this process, some amount of manual intervention with the manager ' s approval is needed for deployment to a production environment.

Continuous Deployment	Continuous Delivery
Here, the application is run by following the automated set of instructions, and no approvals are needed.	Here, the working of the application depends on the decision of the team.



Q 14. What can you say about antipatterns of DevOps?

Ans : A pattern is something that is most commonly followed by large masses of entities. If a pattern is adopted by an organization just because it is being followed by others without gauging the requirements of the organization, then it becomes an anti-pattern.

Similarly, there are multiple myths surrounding DevOps which can contribute to antipatterns, they are:

- DevOps is a process and not a culture.
- DevOps is nothing but Agile.
- There should be a separate DevOps group.
- DevOps solves every problem.
- DevOps equates to developers running a production environment.

- DevOps follows Development-driven management
- DevOps does not focus much on development.
- As we are a unique organization, we don't follow the masses and hence we won't implement DevOps.

Q 15. Can you tell me something about Memcached?

Ans : Memcached is an open-source and free in-memory object caching system that has high performance and is distributed and generic in nature. It is mainly used for speeding the dynamic web applications by reducing the database load.

Memcached can be used in the following cases:

- Profile caching in social networking domains like Facebook.
- Web page caching in the content aggregation domain.
- Profile tracking in Ad targeting domain.
- Session caching in e-commerce, gaming, and entertainment domain.
- Database query optimization and scaling in the Location-based services domain.

Benefits of Memcached:

- Using Memcached speeds up the application processes by reducing the hits to a database and reducing the I/O access.
- It helps in determining what steps are more frequently followed and helps in deciding what to cache.

Some of the drawbacks of using Memcached are:

- In case of failure, the data is lost as it is neither a persistent data store nor a database.
- It is not an application-specific cache.
- Large objects cannot be cached.

Q 16. What are the various branching strategies used in the version control system?

Ans : Branching is a very important concept in version control systems like git which facilitates team collaboration.

Some of the most commonly used branching types are:

➢ Feature branching

- This branching type ensures that a particular feature of a project is maintained in a branch.
- Once the feature is fully validated, the branch is then merged into the main branch.

➢ Task branching

- Here, each task is maintained in its own branch with the task key being the branch name.
- Naming the branch name as a task name makes it easy to identify what task is getting covered in what branch.

➢ Release branching

- This type of branching is done once a set of features meant for a release are completed, they can be cloned into a branch called the release branch. Any further features will not be added to this branch.
- Only bug fixes, documentation, and release-related activities are done in a release branch.
- Once the things are ready, the releases get merged into the main branch and are tagged with the release version number.
- These changes also need to be pushed into the develop branch which would have progressed with new feature development.

The branching strategies followed would vary from company to company based on their requirements and strategies.

Q 17. Can you list down certain KPIs which are used for gauging the success of DevOps?

Ans : KPIs stands for Key Performance Indicators. Some of the popular KPIs used for gauging the success of DevOps are:

- Application usage, performance, and traffic
- Automated Test Case Pass Percentage.
- Application Availability
- Change volume requests
- Customer tickets
- Successful deployment frequency and time
- Error/Failure rates
- Failed deployments
- Meantime to detection (MTTD)
- Meantime to recovery (MTTR)

Q 18. What is CBD in DevOps?

Ans : CBD stands for Component-Based Development. It is a unique way for approaching product development. Here, developers keep looking for existing well-defined, tested, and verified components of code and relieve the developer of developing from scratch.

Q 19. What is Resilience Testing?

Ans : Resilience Testing is a software process that tests the application for its behavior under uncontrolled and chaotic scenarios. It also ensures that the data and functionality are not lost after encountering a failure.

Q 20. Can you differentiate between continuous testing and automation testing?

Ans : The difference between continuous testing and automation testing is given below:

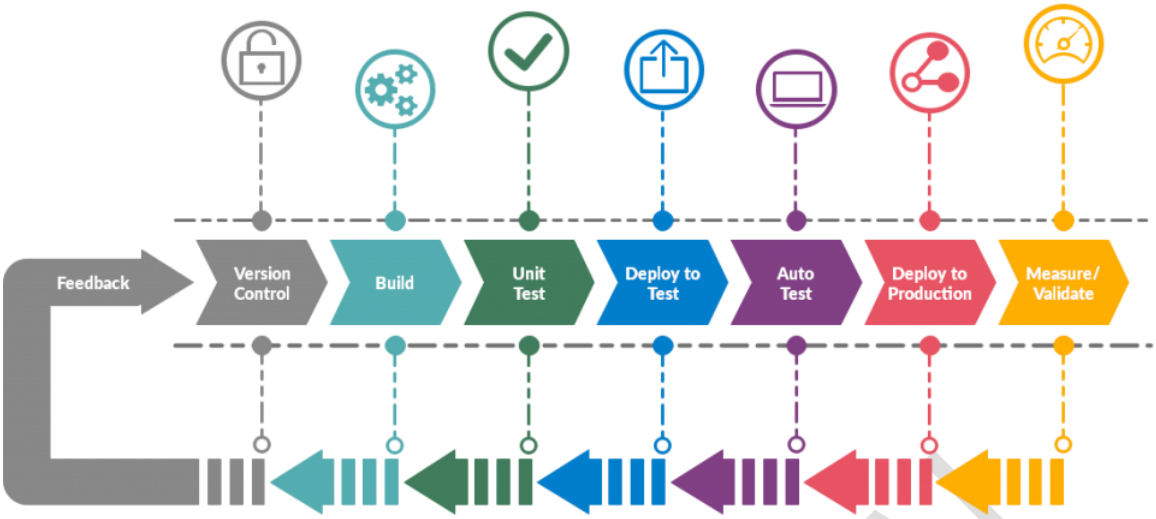
Continuous Testing	Automation Testing
This is the process of executing all the automated test cases and is done as part of the delivery process.	This is a process that replaces manual testing by helping the developers create test cases that can be run multiple times without manual intervention.
This process focuses on the business risks associated with releasing software as early as possible.	This process helps the developer to know whether the features they have developed are bug-free or not by having set of pass/fail points as a reference.

Q 21. Can you say something about the DevOps pipeline?

Ans : A pipeline, in general, is a set of automated tasks/processes defined and followed by the software engineering team.

DevOps pipeline is a pipeline which allows the DevOps engineers and the software developers to efficiently and reliably compile, build and deploy the software code to the production environments in a hassle free manner.

Following image shows an example of an effective DevOps pipeline for deployment.



The flow is as follows:

- Developer works on completing a functionality.
- Developer deploys his code to the test environment.
- Testers work on validating the feature. Business team can intervene and provide feedback too.
- Developers work on the test and business feedback in continuous collaboration manner.
- The code is then released to the production and validated again.

Q 22. Tell me something about Ansible work in DevOps

Ans : It is a DevOps open-source automation tool which helps in modernizing the development and deployment process of applications in faster manner. It has gained popularity due to simplicity in understanding, using, and adopting it which largely helped people across the globe to work in a collaborative manner.

Ansible	Developers	Operations	QA	Business/Clients
Challenges	Developers tend to focus a lot of time on tooling rather than delivering the results.	Operations team would require uniform technology that can be used by different skillset groups easily.	Quality Assurance team would require to keep track of what has been changed in the feature and when it has been changed.	Clients worry about getting the products to the market as soon as possible.
Need	Developers need to respond to new features/bugs and scale the efforts based on the demand.	Operation team need a central governing tool to monitor different systems and its workloads.	Quality Assurance team need to focus on reducing human error risk as much as possible for bug-free product.	Clients need to create a competitive advantage for their products in the market.
How does Ansible help?	Helps developers to discover bugs at an earlier phase, and assists them to perform faster deployments in a reliable manner.	Helps the Operations team to reduce their efforts on shadowing IT people and reduce the times taken for deployment. Also, Ansible assists them to perform automated	Helps QA team to establish automated test cases irrespective of the environments for achieving more reliable and accurate results. Helps to define identical security	Helps the Business team to ensure the IT team is on the right track. Also helps them to optimize the time taken for project innovation and strategising. Helps teams to collaborate in an

Ansible	Developers	Operations	QA	Business/Clients
		patching.	baselines and helps them reduce the burden of following traditional documentation.	effective manner.

Q 23. How does Ansible work?

Ans : Ansible has two types of servers categorized as:

- Controlling machines
- Nodes

For this to work, Ansible is installed on controlling machine using which the nodes are managed by means of using SSH. The location of the nodes would be specified and configured in the inventories of the controlling machine.

Ansible does not require any installations on the remote node servers due its nature of being agentless. Hence, no background process needs to be executed while managing any remote nodes.

Ansible can manage lots of nodes from a single controlling system my making use of Ansible Playbooks through SSH connection. Playbooks are of the YAML format and are capable to perform multiple tasks.

Q 24. How does AWS contribute to DevOps?

Ans : AWS stands for Amazon Web Services and it is a well known cloud provider. AWS helps DevOps by providing the below benefits:

1) Flexible Resources: AWS provides ready-to-use flexible resources for usage.
2) Scaling: Thousands of machines can be deployed on AWS by making use of unlimited storage and computation power.
3) Automation: Lots of tasks can be automated by using various services provided by AWS.
4) Security: AWS is secure and using its various security options provided under the hood of Identity and Access Management (IAM), the application deployments and builds can be secured.

Q 25. What can be a preparatory approach for developing a project using the DevOps methodology?

Ans : The project can be developed by following the below stages by making use of DevOps:

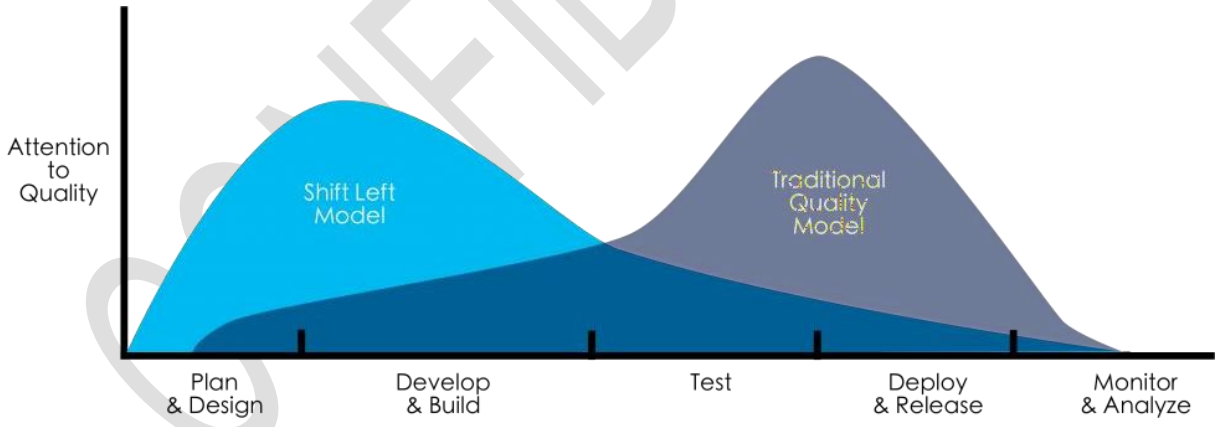
- Stage 1: Plan: Plan and come up with a roadmap for implementation by performing a thorough assessment of the already existing processes to identify the areas of improvement and the blindspots.
- Stage 2: PoC: Come up with a proof of concept (PoC) just to get an idea regarding the complexities involved. Once the PoC is approved, the actual implementation work of the project would start.
- Stage 3: Follow DevOps: Once the project is ready for implementation, actual DevOps culture could be followed by making use of its phases like version control, continuous integration, continuous testing, continuous deployment, continuous delivery, and continuous monitoring.

Q 26. Can you explain the "Shift left to reduce failure" concept inDevOps?

Ans : In order to understand what this means, we first need to know how the traditional SDLC cycle works. In the traditional cycle, there are 2 main sides -

- The left side of the cycle consists of the planning, design, and development phase
- The right side of the cycle includes stress testing, production staging, and user acceptance.

In DevOps, shifting left simply means taking up as many tasks that usually take place at the end of the application development process as possible into the earlier stages of application development. From the below graph, we can see that if the shift left operations are followed, the chances of errors faced during the later stages of application development would greatly reduce as it would have been identified and solved in the earlier stages itself.



The most popular ways of accomplishing shift left in DevOps is to:

- Work side by side with the development team while creating the deployment and test case automation. This is the first and the obvious step in achieving shift left. This is done because of the well-known fact that the failures that get notices in the production environment are not seen earlier quite often. These failures can be linked directly to:

    1. Different deployment procedures used by the development team while developing their features.
    2. Production deployment procedures sometimes tend to be way different than the development procedure. There can be differences in tooling and sometimes the process might also be manual.

- Both the dev team and the operations teams are expected to take ownership to develop and maintain standard procedures for deployment by making use of the cloud and the pattern capabilities. This aids in giving the confidence that the production deployments would be successful.
- Usage of pattern capabilities to avoid configurational level inconsistencies in the different environments being used. This would require the dev team and the operation team to come together and work in developing a standard process that guides developers to test their application in the development environment in the same way as they test in the production environment.

Q 27. Do you know about post mortem meetings in DevOps?

Ans : Post Mortem meetings are those that are arranged to discuss if certain things go wrong while implementing the DevOps methodology. When this meeting is conducted, it is expected that the team has to arrive at steps that need to be taken in order to avoid the failure(s) in the future.

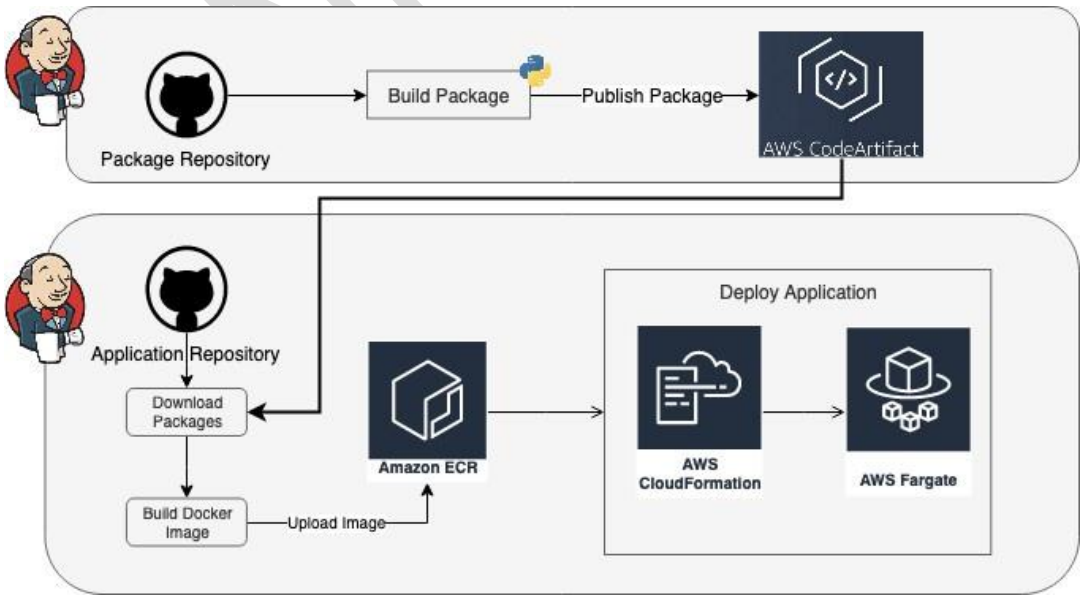
Q 28. What is the concept behind sudo in Linux OS?

Ans : Sudo stands for ` superuser do ' where the superuser is the root user of Linux. It is a program for Linux/Unix-based systems that gives provision to allow the users with superuser roles to use certain system commands at their root level.

Q 29. Can you explain the architecture of Jenkins?

Ans : Jenkins follows the master-slave architecture. The master pulls the latest code from the GitHub repository whenever there is a commitment made to the code. The master requests slaves to perform operations like build, test and run and produce test case reports. This workload is distributed to all the slaves in a uniform manner.

Jenkins also uses multiple slaves because there might be chances that require different test case suites to be run for different environments once the code commits are done.



Q 30. Can you explain the "infrastructure as code" (IaC) concept?

Ans : As the name indicates, IaC mainly relies on perceiving infrastructure in the same way as any code which is why it is commonly referred to as "programmable infrastructure" . It simply provides means to define and manage the IT infrastructure by using configuration files.

This concept came into prominence because of the limitations associated with the traditional way of managing the infrastructure. Traditionally, the infrastructure was managed manually and the dedicated people had to set up the servers physically. Only after this step was done, the application would have been deployed. Manual configuration and setup were constantly prone to human errors and inconsistencies.

This also involved increased cost in hiring and managing multiple people ranging from network engineers to hardware technicians to manage the infrastructural tasks. The major problem with the traditional approach was decreased scalability and application availability which impacted the speed of request processing. Manual configurations were also time-consuming and in case the application had a sudden spike in user usage, the administrators would desperately work on keeping the system available for a large load. This would impact the application availability.

IaC solved all the above problems. IaC can be implemented in 2 approaches:

1. Imperative approach: This approach "gives orders" and defines a sequence of instructions that can help the system in reaching the final output.
2. Declarative approach: This approach "declares" the desired outcome first based on which the infrastructure is built to reach the final result.

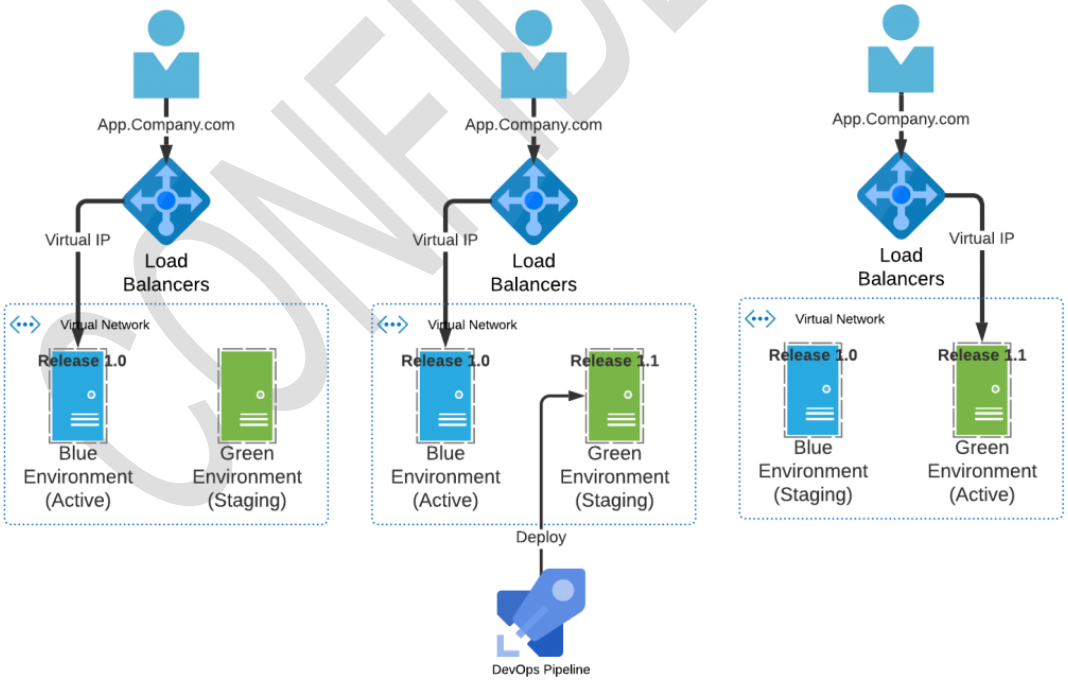
Q 31. What is `Pair Programming' ?

Ans : Pair programming is an engineering practice where two programmers work on the same system, same design, and same code. They follow the rules of "Extreme Programming". Here, one programmer is termed as `` driver '' while the other acts as `` observer '' which continuously monitors the project progress to identify any further problems.

Q 32. What is Blue/Green Deployment Pattern?

Ans : A blue-green pattern is a type of continuous deployment, application release pattern which focuses on gradually transferring the user traffic from a previously working version of the software or service to an almost identical new release - both versions running on production.

The blue environment would indicate the old version of the application whereas the green environment would be the new version.



The production traffic would be moved gradually from blue to green environment and once it is fully transferred, the blue environment is kept on hold just in case of rollback necessity.

In this pattern, the team has to ensure two identical prod environments but only one of them would be LIVE at a given point of time. Since the blue environment is more steady, the LIVE one is usually the blue environment.

Q 33. What is Dogpile effect? How can it be prevented?

Ans : It is also referred to as cache stampede which can occur when huge parallel computing systems employing caching strategies are subjected to very high load.

It is referred to as that event that occurs when the cache expires (or invalidated) and multiple requests are hit to the website at the same time.

The most common way of preventing dogpiling is by implementing semaphore locks in the cache. When the cache expires in this system, the first process to acquire the lock would generate the new value to the cache.

Q 34. What are the steps to be undertaken to configure git repository so that it runs the code sanity checking tooks before any commits? How do you prevent it from happening again if the sanity testing fails?

Ans : Sanity testing, also known as smoke testing, is a process used to determine if it ' s reasonable to proceed to test.
Git repository provides a hook called pre-commit which gets triggered right before a commit happens. A simple script by making use of this hook can be written to achieve the smoke test.

The script can be used to run other tools like linters and perform sanity checks on the changes that would be committed into the repository.

The following snippet is an example of one such script:

```
#!/bin/sh
files=$(git diff –cached –name-only –diff-filter=ACM | grep
‘.py$’)
if [ -z files ]; then
exit 0
fi
unfmtd=$(pyfmt -l $files)
if [ -z unfmtd ]; then
exit 0
fi
echo “Some .py files are not properly fmt’d”
exit 1
```

The above script checks if any .py files which are to be committed are properly formatted by making use of the python formatting tool pyfmt.

If the files are not properly formatted, then the script prevents the changes to be committed to the repository by exiting with status 1.

`

Q 35. How can you ensure a script runs every time repository gets new commits through git push?

Ans : There are three means of setting up a script on the destination repository to get executed depending on when the script has to be triggered exactly. These means are called hooks and they are of three types:

- Pre-receive hook: This hook is invoked before the references are updated when commits are being pushed. This hook is useful in ensuring the scripts related to enforcing development policies are run.
- Update hook: This hook triggers the script to run before any updates are actually made. This hook is called once for every commit which has been pushed to the repository.
- Post-receive hook: This hook helps trigger the script after the updates or changes have been accepted by the destination repository. This hook is ideal for configuring deployment scripts, any continuous integration-based scripts or email notifications process to the team, etc.

Brij Mohan