1.Show the journey of devops from traditional waterfall model

1 waterfall is a traditional practice for developing systems. it is a practice of developing in stages Gather and analyze

software requirements, design, develop, test and deploy into operations.

The output of one stage is required to initiate the next stage.

2 The Agile methodology allows for exploration of new ideas and quicker determinations about which of those ideas are viable.

Agile methods are designed to adapt to changing business needs during development.

3 DevOps is an engineering culture that aims to unify development and operations in ways that lead to more efficient development.

It is not a standard or framework, but instead an organizational collaboration that has given rise to a set of best practices

through continuous integration, continuous delivery and continuous testing.

2. Advantages of devops over agile development

DevOps is a practice of bringing development and operations teams together whereas Agile is an iterative approach that focuses on collaboration, customer feedback and small rapid releases.

DevOps focuses on constant testing and delivery while the Agile process focuses on constant changes.

DevOps requires relatively a large team while Agile requires a small team.

DevOps leverages both shifts left and right principles, on the other hand, Agile leverage shift-left principle.

The target area of Agile is Software development whereas the Target area of DevOps is to give end-to-end business solutions and fast delivery.

DevOps focuses more on operational and business readiness whereas Agile focuses on functional and non-function readiness.

3. Benefits of using the devops

1. Ensure faster deployment

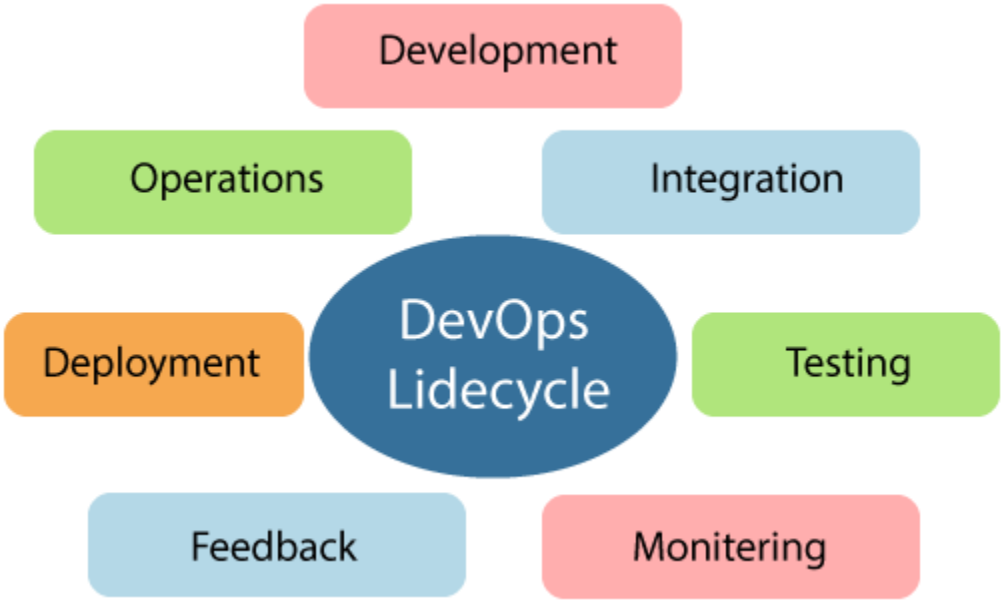
2.Stabilize work environment

3.Continuous delivery of software

4.Fast and reliable problem-solving techniques

5.Minimal cost of production

4.Explain lifecycle of devops



- **Continuous Development –** Continuous development involves planning, outlining, and introducing new code. The aim of continuous development is to optimize the procedure of code-building and to reduce the time between development and deployment.
- **Continuous Integration (CI) –** This practice of DevOps implementation involves the integration of developed code into a central repository where configuration management (CM) tools are integrated with test & development tools to track the code development status. CI also includes quick feedback between testing and development to be able to identify and resolve various code issues that might arise during the process.
- **Continuous Testing** – The aim of continuous testing is to speed up the delivery of code to production. This phase of DevOps involves simultaneous running of pre-scheduled and automated code tests as application code is being updated.
- **Continuous Delivery** – Continuous delivery is aimed at quick and sustainable delivery of updates and changes ready to be deployed in

the production environment. Continuous delivery ensures that even with frequent changes by developers, code is always in the deployable state.

- **Continuous Deployment (CD) –** This practice also automates the release of new or changed code into production similar to continuous delivery. The use of various container technology tools such as Docker and Kubernetes allow continuous deployment as they play a key role in maintaining code consistency across various deployment environments.

- **Continuous Monitoring** – It involves ongoing monitoring of the operational code and the underlying infrastructure supporting it. Changes/application deployed in the production environment is continuously monitored to ensure stability and best performance of the application.

5.Various tools used for each phase of devops

Planning:- Jira , confluence , Slack
Coding:- *Stash, GitHub, GitLab*
Software build:- *Docker, Puppet, Chef, Ansible, Gradle.*
Testing:- *Vagrant, Selenium, JUnit, Codeception, BlazeMeter, TestNG*
Deployment:- *Jenkins, Kubernetes, Docker, OpenShift, OpenStack, Jira.*
Monitoring:- *Nagios, Splunk, Slack, New Relic, Datadog, Wireshark*

6.Difference between DVCS and CVCS

Centralized Version Control	Distributed Version Control
In CVS, a client need to get local copy of source from server, do the changes and commit those changes to centeral source on server.	In DVS, each client can have a local branch as well and have a complete history on it. Client need to push the changes to branch which will then be pushed to server repository.
CVS systems are easy to learn and set up.	DVS systems are difficult for beginners. Multiple commands needs to be remembered.
Working on branches in difficult in CVS. Developer often faces merge conflicts.	Working on branches in easier in DVS. Developer faces lesser conflicts.
CVS system do not provide offline access.	DVD systems are workable offline as a client copies the entire repository on their local machine.
CVS is slower as every command need to communicate with server.	DVS is faster as mostly user deals with local copy without hitting server everytime.
If CVS Server is down, developers cannot work.	If DVS server is down, developer can work using their local copies.