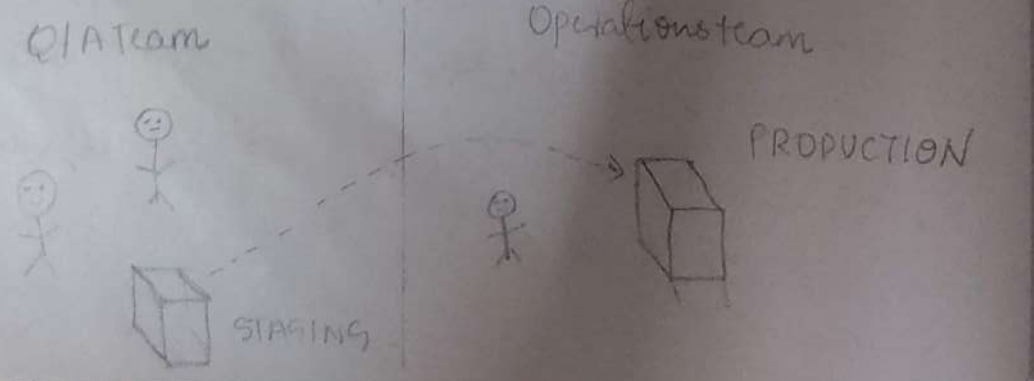
3. Explain How DevOps and ITIL coexist.

- DevOps and ITIL are mutually exclusive and are usually considered as alternatives to each other.
- DevOps focusses on communication between development and operations team to deliver s/w with high quality.
- ITIL (Information Technology infrastructure library) is a framework for IT service management that manages IT services approach in ITIL systematically
- Combining DevOps and ITIL can bring significant benefits to a company.
- It leads to reduced inefficiencies, security gaps and increases delivery speed and efficiency.

- Provides more efficiency, agile, customer focussed approach to IT service delivery.

4. Explain staging / production.

- Staging environments are the last line of test environment and interchangable.
- Production environment refers to the live, operational environment where is running and serving end users.
- staging and Production are important components in continuous delivery process.
- Changes are testing is validated in staging before being deployed for production.
- staging and production gives
  - → Improved quality.
  - → Increase in speed
  - → Reduad risk.



5. Delivery pipeline

- It is a set of automated process i.e used to build, test, deploy s/w changes from development to production.

STAGES

- Code development and version control using GIT.
- Continuous integration
- Continuous testing
- Deployment
- staging
- Release
- monitoring.

- Each stage is integrated with the other making rapid delivery of s/w changes.
- Provides transparency and accountability.
- Reduces risk and improves collabration btw development and operation teams.

...

⑧ Continuous Delivery and Monitoring

- Continuous delivery and monitoring are two key aspects in devOps that work together and improve speed and quality of s/w delivery.

- Continuous delivery focuses on automatic building, testing and deploying code

6. Wrapping up.

- It refers to bringing a s/w development project on a close.
- It involves different tasks depending on the specifics of project.
- It makes sure that all pieces of development, fit together for a smooth functioning system.
- These pieces are tested independently, mostly through automation during the process.
- It makes the process more efficient and streamlined.

→ Improve
→ Increase in speed
→ Reduce risk.

⑦ Identifying Bottlenecks.

- It is a process of identifying areas of pipeline where work is not flowing smoothly, causing delays or slowing down in delivery process.

Techniques

(i) Metrics and monitoring, by collecting and analyzing metrics like build time, test time, deployment time.

(ii) Process mapping, by identifying areas where work is getting slow (or) stuck.

(iii) Feedback from team members to get the insights of the pipeline.

- By using the above techniques, it helps improve work flow, increase speed, reduce risk.

⑧ Continuous Delivery and Monitoring

- Continuous delivery and monitoring are two key aspects in devOps that work together and improve speed and quality of s/w delivery.

- Continuous delivery focuses on automatic building, testing and deploying code

changes to production as soon as they are ready. The goal is to reduce the time and improve speed in delivery process.

- Continuous monitoring identify and resolves problem quickly through automated monitoring tools and logging systems. It reduces downtime, resolves issues quickly.
- It allows organizations to form a feedback look [i.e both from continuous monitoring & delivery] to improve delivery pipeline & production system.

(9) DevOps S/W architecture.

- It refers to the design and organization of tools, system and processes used to implent devOps pratices.

COMPONENTS

- SOURCE code management is to store and manage code base.
- Continuous Integration such as Jenkins to automate building, testing & deployment of code.

- Monitoring tools such as Nagios used to monitor the performance of production system.
- Containers such as Rocket can be used to deploy applications.

- The main goal of devops architecture (s/w) is to increase the speed and efficiency of s/w delivery.

(10) Monolithic scenario.

- To understand monolithic scenario. Lets suppose we have large web application with different functions.
- It also contains a static website inside the application.
- Even a single mistake in website, we need to rebuild the entire web application and deploy it again.

ISSUES

- Need to change the entire code base.
- Make the bizarre and correct the spelling.
- Built a new abstract with correction.
- Deploy new attribute to productions.

(11) Automation in DevOps is a result of utilization of cloud computing.

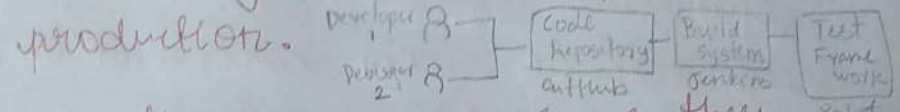
- Infrastructure as a service that allows organization to quickly manage computing resources as needed without maintaining physical infrastructure that allows improving speed for s/w delivery.
- It provides tools and services such as scripts, API's, configuration management tool.
- Continuous integration and delivery provides tools & services such as build & deploy pipelines.
- Monitoring and logging to check the performance metrics improving the visibility & reliability of their applications.

(12) Benefits of CI/CD continuous

- Faster s/w delivery
- Improved quality
- Increased collaboration
- Enhanced security
- Better resource allocation.
- Faster feedback

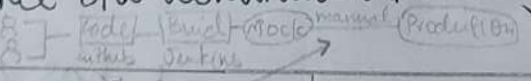
(13) Continuous Integration and continuous delivery are names of same processes or not.

- Continuous Integration refers to the practice of automatically building, testing and integrating code.
- Continuous Delivery refers to the practice of automating the release of code changes to production.



- They are the same process as they common goal is to make release faster, reliable and less error.
- CI focusses on continuous integration of code and CD focusses on continuous delivery of code changes to production.

(14) Difference b/w Continuous Delivery & Deployment



Continuous Delivery	Continuous Deployment
• A s/w engineering practice code changes are automatically prepared for deployment. • Improved developer productivity. • Bugs are found and addressed earlier. • New updates released.	• A s/w engineering pratice code changes to automatically release into production environment. • Faster release process. • Fixing problem easier. • No human intervention.

(15) Scrum is a methodology to exhibit in Agile. How does Scrum comply with Agile manifesto principles.

- Scrum emphasis on strong collaboration and communication between team members.
- It prioritizes the delivery of working s/w with regular releases.
- It requires closer collaboration btw development team & customer.
- Designed to be flexible and adaptable.
- Encourages technical excellence and good design.
- Regular and continuous improvement of process in sustainable development practice in SCRUM.

(16) Software Control Management has a workflow. Explain in detail capturing all activities in relation to DevOps Release Management Plan & Agile Sprint planning.

- Keeping track of code changes and maintaining a history of all versions through version control system.

(17)

- Creating separate branches for different features or bug fixes allowing parallel development and testing without affecting the main codebase.
- Integrating code changes from different branches into main codebase.
- Changes in building and code changes s/w automatically.
- Deployment of s/w to different environments.

(17) Write in detail about shared authentication & different GIT server application.

- GITLab offers built-in issue tracking, code review, project management tools.
- GITHub provides user friendly interface for managing code with features for code review.
- GIT Bucket uses web based interface for managing Git Based repositories.
- Gogs is a Git server that is designed to be easy set up and use.

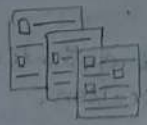
(18) Explain 7C's of DevOps life cycle for business facility.

Continuous Planning

Planning for new delivery features continuously.

Continuous Feedback.

- Improving by analyzing the result operations from the s/w.



Continuous Delivery & Monitoring

Delivery :- change in functional / config. changes

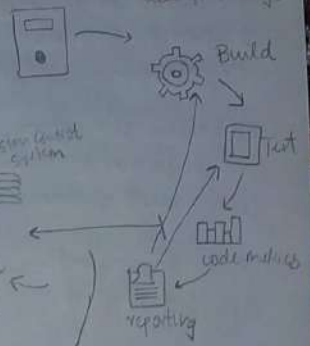
Monitoring - Health of the application, performance, security.

Continuous Development

This phase involves planning and coding of s/w

Continuous Integration

Nearly always lifecycle. Developers require to commit changes in source code frequently.



Continuous Testing.

The developed s/w is continuously checked for bugs.

Cut environment

QA environment

staging environment

Continuous Deployment

Code is deployed in the production servers.

RR RRR Public Jobs

Notifications

(19) Difference between DevOps and Agile Methodology

DevOps	Agile.
• Emphasis on collaboration & productivity.	• Emphasis on incremental changes.
• Focuses on functional & non functional readiness.	• Focuses on operational & business readiness.
• Addresses the gap btw customer & development teams.	• Addresses the gap btw development & Operation team.
• Large team with different skillset.	• Small team with advanced skillset.
• Exhsive documentation	• Light on documentation.
• Received via customers	• Received internally.

(20) Types of automated testing & Pros, cons.

(i) Unit Testing :- It is done to individual classes or function.

(ii) Integration Testing :- It makes sure the multiple modules of applications work

well with each other.

(iii) System Testing :- It tests the entire system in an environment replicates the real production environment.

(iv) UI Testing :- It makes sure that the application works great as per the user perspective.

PROS AND CONS

PROS	CONS
• Speed & Efficiency	• Time consuming & expensive.
• Consistency	• Maintainence cost.
• Increased Confidence.	• Debugging.
• Improved test coverage	• False results.
• Improved Return of investment	• Dependency of code quality.

(21) Javascript Testing

- It ensures the quality and reliability of Javascript code in web applications.

TOOLS AND FRAMEWORK

- Jest is popular Javascript testing framework that provides easy to use API
- Mocha supports both synchronous & asynchronous testing.

- Karma :- It runs test in real browsers making possible to test code in same environment as end user.
- Chai :- Simple & readable syntax for writing tests
- AVA :- Fast & modern Javascript testing framework that supports parallel testing.

(22) Test driven development

- It emphasizes the importance of writing automated tests before writing code.
- It can be used to validate the functionality & quality of s/w changes as part of CI/CD pipeline.

Benefits

- Improved quality code
- Faster and frequent releases
- Better collaboration with development & operations teams.
- Reduce the risks of bugs.
- Valuable tool for organizations that are looking to adopt a DevOps approach to s/w development.

(23)

(24)

(23) REPL driven development.

- REPL (Read Eval Print Loop) is a REPL interface interface to write and test code incrementally.
- Developers write small pieces of code, evaluate and immediately see the results.
- It can be used to quickly test and validate code changes without need for full build & deployment.

BENEFITS

- Faster & iterative development
- Improved collaboration
- Reduced risk.

(24) Deployment systems & Tools.

- Deployment system and tools are crucial components as they allow organization to automate the same process of deploying and managing s/w applications.

TOOLS

(i) Chef is a ruby based deployment system from Opscode. To install chef we need a command 'sudo' to proceed.

(iii) Salt Stack :- It is a python based deployment solution. It consists of a salt master and salt minion.

(iii) Docker :- Allows organization to package and deploy applications and services as containers i.e portable lightweight.

(25) Ansible

- It is a deployment sol^n that favours simplicity.
- It works on push-based model i.e it pushes tasks and configuration to the targeted systems Rather than pulling them.
- It requires no special skills or knowledge to get started.
- Popular choice for organization.
- It provides high degree of flexibity & control over target system.

(26) Puppet master and agent.

- Puppet master is responsible for distributing configuration info to the puppet agents
- Puppet agent is responsible for applying configurational info received from the master to the target system.
- Key benefits of puppet architecture is flexibility.
  - Ease of use.
  - Puppet language is simple & intuitive.
  - High level abstraction that makes easy to manage complex systems.