Ally Financial Training Questions

Jacob Cantrell

Can you explain your understanding of DevOps and how it differs from traditional software development methodologies?

DevOps is a software development approach that emphasizes collaboration and communication between software development & IT operations teams. It aims to bring the two sides closer and automate many processes involved.

DevOps differs from traditional software development methodologies in ways such as emphasizing a cultural shift with development and operations teams working more closely together; relying heavily on automation, which reduces the time it takes code to get from development to production; and traditional software development methodologies tend to be more siloed and less reliant on automation, while not necessarily prioritizing CI/CD.

How do you use tools such as Git, Jenkins, and Ansible to automate the software development and deployment process?

Git is a distributed version control system that allows developers to manage their source code and collaborate with others, enabling teams to track changes, revert to previous versions, and merge changes, automating version control and collaboration.

Jenkins is an open-source automation server which can be configured to automate building, testing, and deployment everytime changes are pushed to a Git repository.

Ansible can be used to automate provisioning, configuration, and deployment of software. For instance, Ansible can automate the deployment of a new version of an application or the configuration of servers.

Can you walk me through a recent project you managed using DevOps practices and the challenges you faced?

In a recent DevOps project, a team used Git for version control & collaboration, Jenkins for CI/CD, and Ansible for infrastructure as code.

One challenge faced was ensuring the CI/CD pipeline was fast, reliable, and consistent. To overcome this, automated testing & code quality checks at every stage of the pipeline can be implemented. Automatic rollback mechanism in case of failures, which helped minimize downtime and reduce risk of human error, can also be set up.

Another challenge was ensuring the security of infrastructure and applications. To overcome this, security measures such as encryption, network security, and continuous monitoring & logging of infrastructure and applications can be implemented.

How do you use containerization and orchestration technologies, such as Docker & Kubernetes, to improve scalability and reliability?

Docker & Kubernetes improve scalability and reliability by:

1. Easy Deployment - Containers can be easily deployed on any system, making it easy to scale application horizontally by adding more containers.
2. Improved Resource Utilization - Containers share the host system's resources, allowing multiple containers to run on a single host.
3. High Availability - Kubernetes provides automatic failover & self-healing to ensure apps continue to run even if a host or container fails
4. Environment Consistency - Containers provide a consistent environment, making it easy to move apps from development to production
5. Automated Scaling - Kubernetes can automatically scale apps based on demand

Can you explain how you use monitoring and logging tools, such as Prometheus & ELK, to improve visibility & troubleshoot issues in production environment?

Prometheus & ELK improve visibility & troubleshoot issues by:

1. Real-time Monitoring - Prometheus provides real-time monitoring & alerting making it easy to detect & respond to issues in a timely manner
2. Root Cause Analysis - Prometheus provides numerous performance metrics that can identify the root cause of an issue
3. Improved Collaboration - ELK provides centralized location for log data, making it easy for multiple teams to collaborate on troubleshooting
4. Automated Alerting - Prometheus provides automated alerting, making it easy to set up alerts for specific conditions

How do you handle & manage infrastructure as code using tools such as Terraform or CloudFormation?

Terraform & CloudFormation help handle & manage IaC in the following ways:

1. Version Control - they use version control systems to store & manage configuration files that define the infrastructure
2. Repeatable & Automated Deployments - provide a declarative syntax that allows developers to describe desired state of their infrastructure
3. Infrastructure Testing - allow devs to write test to validate the infrastructure before it is deployed to production
4. Infrastructure Management - provides centralized location for managing infrastructure, making it easy to view & manage the entire infrastructure
5. Cost Optimization - provide way to manage infrastructure costs by automating provisioning & scaling of resources

Can you discuss your experience with CI/CD pipelines?

CI/CD pipelines help organizations improve the speed, quality, and reliability of software delivery. They also provide a way to reduce the risk of human error and make it easier to track changes, making it easier to maintain software over time. They also provide source code management and monitoring & feedback.

How do you use tools, such as JIRA or Trello, to manage & track work in a DevOps environment?

JIRA & Trello help track & manage work in a DevOps environment by:

1. Task Management - allow teams to create & track tasks, bugs, and features, and assign them to team members
2. Workflow Management - provide a way to manage & visualize the workflow of tasks, bugs, and features
3. Collaboration - provide centralized location for team members to work on projects in real-time
4. Reporting - provides a way to generate reports & dashboard to help the team track progress, identify trends, and make informed decisions

Can you explain how you use A/B testing and canary releases to minimize risk and improve the release process?

A/B Testing allows the testing of two or more variations of a product or feature with a small subset of users to determine which one performs better, reducing the risk of releasing a faulty product.

Canary releases releases new software to a select small subset of users, which allows organizations to validate the new software in a production-like environment and catch & resolve issues before releasing to a broader customer base.