PHASE3 IQ WITH HANDS-ON TASKS

Note - These Questions have been shared from your 6.0 Students Community

1)What are the things that you check for approval of PR requests on Github?
2)Before triggering the pipeline and checking the sonarqube report for analysis, at code or git level what does the pipeline check?
3)Possible security measures that can be taken before CI pipeline gets triggered?
4)How would you optimise CI/CD pipeline for large-scale apps to reduce build and deployment time?
5) Is the webhook trigger on git applied for all branches or for all pushes done by the developer?
6) OS and its troubleshooting
7) Networking and its troubleshooting

Jenkins

1) How do you set up a jenkins server in your organisation, can you walk me through?
2)What is the memory capacity and size of servers for jenkins master, slave, sonarqube, maven,docker, kubernetes, Prometheus, Grafana?
3)what database you use in your organisation, what work you do with database, how to
4)secure database, what disaster recovery strategy do you follow?
What strategy follows for Jenkins master slave architecture in your organisation?
5)Where do you store credentials for jenkins in your organisation?
6)What type of authentication is used for Jenkins in your organisation?
7)What is Jenkins' single sign on ?
8)how to troubleshoot and resolve failed builds or jobs in Jenkins

Github :

1)Can you explain git merge, rebase, and where do you use it in your organisation?
2)Where do you face merge conflicts? What is the approach you follow?
3)Where are GitHub credentials stored in the organisation?
4)discuss git tags and their significance in a release management process?
5)How do you apply tags to projects on GitHub?
6)how to manage secure sensitive information such as API keys credentials in a github repository explain process of Rolling back changes in a github repository especially using github

Maven:

1)Strategies for managing and resolving dependencies in maven

2)discuss strategies for optimising maven build performance purpose of maven settings.xml file how to configure it

Sonarqube:

1)what are the challenges you face when implementing sonarqube in a complex project
2)what are sonarqube Matrix how does sonarqube measure them
discuss significance of code smells, how Sonarqube identify and address them?
3)sonarqube different security levels used and their implications
explain purpose of solar cube quality gates how they contribute to development process

Docker:

1)In docker there are 5 containers, how do you identify which container has an issue?
2)How docker is used for micro services in your project?

kubernetes:

1)Difference between persistent volume and persistent volume claim?
2)How do you identify the pod causing the issue from 10 pods?
3)How do you decide cpu, memory usage limit for each pod?

Eks:

1)explain integration of EKS with IAM
2)How EKS handle automatic node scaling and what are the considerations for scaling clusters
3)concept of worker notes in E K S how they are provisioned
4)how EKS handle updates and patches for K8s control plane and worker notes
5)discuss monitoring and login options available for eks clusters
6)what are the security best practices for Amazon eks including for security policies and network policies
7)how to optimise performance of applications running on EKS consider for multi region deployments with Amazon eks
8)how EKS handle rolling update and rollbacks of applications
9)explain use of Helm charts in context of eks deployments
10)troubleshoot and diagnose issues in eks cluster

AWS Troubleshooting:

1)What do you do if you get an error in the browser such as 403 forbidden, 404 not found, 500 internet server error, 400 bad requests, 502 bad gateway, 503 service unavailable, 416 range not satisfied?
2) Where is the ssl certificate placed in the architecture?
3)How to secure ec2 instances?
4)Where do you use systems manager, secrets manager, hashicorp vault.

ECR:
1)security features of ECR
2)what IAM permissions given to ECR

3)how to configure and setup access control for ECR repositories
process of pushing and pulling docker images to and from ECR
4)how ECR handle image vulnerability scanning
5)strategies to optimise performance of docker pull from ECR
concept of image replication in ECR images

6)how to monitor and track changes in ECR repositories
7)process of image tagging and versioning in ECR
8)migrate docker images from existing repository to ECR

Tasks-HandsON

1. Server Setup:

a. Purchase a server or use an existing server capable of running the required services.
b. Install the necessary operating system and configure any required network Settings.

2. Domain Setup:

a. Purchase a free domain (e.g., domain.com) from a domain registrar or use an existing one.
b. Configure DNS settings to associate the following subdomains with the server#39;s
IP address:
service1.domain.com: Service 1 - Node.js API Application (for storing data in MongoDB)
service2.domain.com: Service 2 - Node.js API Application (for retrieving data from MongoDB)
db.domain.com: MongoDB service

3. Containerization:
a. Choose between Kubernetes or Docker for containerization, and ensure the the chosen platform is set up and running on the server.
b. Create Docker images for each service (MongoDB, Node.js API Application for
storing data, and Node.js API Application for retrieving data) or use pre-existing
images from trusted sources.
c. Push the Docker images to a Docker registry (e.g., Docker Hub).
4. Deployment:
a. Create Kubernetes manifests or Docker Compose files to define the deployment
configuration for each service.
b. Deploy the services using the appropriate Kubernetes or Docker commands,
ensuring the services are accessible within the cluster.
5. Service Accessibility:

a. Verify that each service is accessible within the cluster by accessing them using

Task For Devops setup. 2

the internal IP addresses or service names.
b. Test the functionality of each service to ensure they are working as expected
within the cluster.
6. Ingress Setup:
a. Install and configure an Ingress controller (e.g., Nginx Ingress) on the Kubernetes
cluster or set up an HTTP reverse proxy (e.g., Nginx) on the server.
b. Create Ingress rules or Nginx server blocks to associate the subdomains with
their respective services.

c. Verify that the subdomains correctly route to the corresponding services.
7. Final Testing:
a. Perform end-to-end testing by accessing the services using the associated subdomains (service1.domain.com,service2.domain.com, db.domain.com) from a web browser or API client.
b. Ensure that data can be stored and retrieved from the MongoDB service via the

Node.js API applications

Questions:-
1. If the developer assigned the code then what are the approaches/ team/persons for the complete CI/CD Setup……
2. What are the daily responsibilities
3. What are challenges do you getting in daily tasks

Interview Question:

1) What are the tools and technologies used in your company

2) Explain the different services used in AWS.

3) Types of ELB and what are the situation we choose type of ELB's

4) Why are you using Network Load balancer, Can you explain in detail about NLB&ALB

5) Explain any flow of CI used in your company

6) GIT HUB Web HOOK vs Poll SCM

7) Maven build

8) Plugins used in Jenkins

9) what is Docker file and explain all the instructions

10) Architecture of Kubernetes

11) Why Auto scaling is very important and how you implemented

12) Why Auto scaling is preferred

13) How master and worker nodes communicate in Kubernetes

14) Explain a few objects in Kubernetes

15) What's the branching strategy using,

1) What is Ingress Controller and why it's used.

2) How do you write your manifest files

3) What is Auto Scaling

4) How the developer is notified upon build failure I said slack changes and mail integration

5) ITIR for 192.168.0.0 --> How do you divide it for 2 subnets Asked me to explain in detail 2*32-n

6) Docker file instructions in detail

7) Apart from Jenkins any other CI tools knowledge

8) EFS - When we go for

9) Docker swarm vs Kubernetes Orchestration

10) How you implemented auto scaling using AWS Services I explained about EKS Service with Kubernetes Cluster

11) Why we need to use Helm charts

12) Helm 3.0 vs 2.0 and explanation about architecture

13) Monitoring tools used in your project

HELM:

🔍 What is Helm, and why is it used in Kubernetes?

🔍How do you install Helm and initialise a Helm chart repository?

🔍What is a Helm chart?

🔍How do you create a new Helm chart?

🔍What is the difference between a Helm release and a Helm chart? 🔍How do you deploy a Helm chart to a Kubernetes cluster?

🔍What are Helm values and how are they used?

🔍What is a Helm template, and how does it work?

🔍How do you upgrade a Helm release to a new version of a chart?

🔍What is a Helm hook, and when might you use one?

🔍Explain the difference between Helm 2 and Helm 3.

🔍What are Helm repositories, and how do you add or remove them?

🔍How do you rollback a Helm release to a previous version?

🔍What is Tiller, and why was it removed in Helm 3?
🔍How can you secure Helm deployments in a Kubernetes cluster?

🔍How do you manage dependencies in Helm charts?

🔍What is the purpose of Helm plugins, and can you name a few useful Helm plugins?

🔍How do you perform linting and testing of Helm charts?

Scenario Based Questions:

1. Scenario:

Question: Describe how you would design the deployment architecture for the microservices-based application using Docker to ensure high availability and scalability.

Answer: I would utilise Docker Swarm or Kubernetes for container orchestration to manage the deployment of microservices across multiple nodes. I would design the application to have multiple replicas of each service, distributed across different nodes for fault tolerance. Additionally, I would implement auto-scaling based on resource usage metrics to handle varying levels of traffic efficiently.

2. Scenario: You are tasked with deploying a web application that requires both a frontend and a backend service. The backend service relies on a database. How would you set up the Docker environment to run this application?

Answer: I would create separate Docker containers for the frontend, backend, and database services. Each container would be based on appropriate base images (e.g., Node.js for the frontend, Node.js or another suitable runtime for the backend, and a database image like MySQL or PostgreSQL for the database). I would then use Docker Compose to define the services, their dependencies, and network configurations in a YAML file, allowing easy orchestration and deployment of the entire application stack.

3. Scenario: Your team is developing a CI/CD pipeline for a project that utilizes Docker containers. How would you integrate Docker into the CI/CD process to ensure efficient and reliable delivery of the application?

Answer: I would start by creating Dockerfiles for each component of the application to ensure consistency across different environments. In the CI phase, I would build Docker images for each component, run unit tests within Docker containers, and push the images to a registry. In the CD phase, I would use Docker images in the deployment process, pulling the latest images from the registry, spinning up containers, and running integration tests before promoting the images to production.

4. Scenario: You are deploying a legacy application that requires specific dependencies and configurations onto Docker containers. How would you containerize this application while ensuring compatibility and stability?

Answer: I would start by analysing the dependencies and configurations of the legacy application and identifying any potential conflicts or compatibility issues with Docker. I would then create a Dockerfile that includes instructions to install the necessary dependencies and configure the environment according to the requirements of the legacy application. Additionally, I would use Docker volumes or bind mounts to manage data persistence and ensure that any required external resources are accessible from within the container.

MANAGERIAL ROUND QUESTIONS

1. Can you describe a difficult interaction you've had with a customer? How did you deal with it? Thinking back, what would you do differently?

2. Give me an example of when you did more than what was required from you.

3. Share with me about a time you exceeded expectations.

4. Tell me about a time when you were working on an initiative and saw an opportunity to do something bigger or better than the initial focus. Also how did you convince your team

5. Can you give me an example of a calculated risk where speed was critical?

6. Tell me about your most significant career failure and what you learned from it.

7. Tell me a situation where you went out of your comfort zone to learn and deliver something

8. Tell me about a time you couldn’t meet your own expectations on a project.

9. Tell me about a time when you did not effectively manage your project, and something did not get completed on time?

10. Tell us about a time when you had to solve a complex task under a strict timeline? What was your approach, and how did you solve it?

11. Tell me about a time you received negative feedback from your manager. How did you deal with it?

12. Tell me about a time you solved a complex problem by looking into the details.