**Tell me about yourself?**

My name is Sharath Kumar, I am from Udupi,

I hold a Bachelor's degree in Business management and Master of Commerce,

I trained 6 months internship and received AWS solution Architect and DevOps engineer certificate from Microdegree

Where I have made projects and gained handson experience with a tools like version control tool – git, build tool – Maven, Continue integration and continue deployment orchestration tool – Jenkins, cloud platform – AWS, operating system - Linux, containerization tool – docker and containerize orchestration tool – Kubernetes, infrastructure tool – Terraform, Configuration service management tool – Ansible, Monitoring and visualization tool – Prometheus ad Grafana.

and I also completed job ready projects of 12 real time projects by Singam.

Also received certificate of achievement from Super Scholar for completing the Interview Fast Track Program for IT Professionals

Apart of that I spent 7+ years working in accounting, where I've had the opportunity to work in various accounting software like SAP, Tally ERP 9, Tally Prime and Microsoft office.

Throughout my career in accounting, I have developed strong collaboration skills, working effectively within teams to achieve common goals. I believe that the ability to communicate and collaborate with colleagues is vital in the world of DevOps and

I am an eager learner, constantly striving to improve my skill and stay ahead of the latest trends in software development, I am also a team player and enjoy collaborating on project.

**Tell me about yourself?**

**My name is Sharath Kumar, I am from Udupi**

**I have done Bachelors of Business Management and Master of Commerce**

**I have been working in the industry for over 7 years in Accounting where I was working in SAP.**

**While my background lies in Accounting, my passion for technology, problem-solving & my dedication to mastering the field of DevOps, it has motivated me to pursue this exciting career change.**

**Throughout my career in accounting, I have developed strong collaboration skills, working effectively within teams to achieve common goals. I believe that the ability to communicate and collaborate with colleagues is vital in the world of DevOps.**

**I am an eager learner, constantly striving to improve my skill and stay ahead of the latest trends in software development, I am also a team player and enjoy collaborating on project.**

**What are your top 3/5 areas of interest?**

**Sports and Fitness:** Love for playing cricket, work out, or following athletic events.

**Volunteering and Community Service**: Dedication to giving back to the community and helping others.

**Trekking**: I absolutely love trekking. It's all about going on adventures in nature. I enjoy exploring new trails, overcoming tough paths, and seeing amazing views. (kodachadri hills, Kudremukh, Agumbe)

**Film and Entertainment**: Love for movies, television shows, and entertainment industry news.

**Food and Beverage**: Interest in gourmet dining, wine tasting, or brewing beverages.

These are some of the things I'm really enthusiastic about outside of work, and they help me stay balanced and engaged in various aspects of life."

**What are the achievements that give you the greatest satisfaction?**

I am most satisfied by achievements that contribute to bettering work processes, mentoring others, and earning recognition

**Why should we hire you?**

Honestly, I possess all the skills and experience that you’re looking for.

I’m pretty confident that I am the best candidate for this job role.

I am a self-motivated person and I try to exceed my superior’s expectations with high-quality work.

Being a fast learner, I quickly pick up business knowledge related to my project.

Lastly, I would like to add that I work well both as an individual contributor and also as a team member.

Collectively, all these skills put together make me a complete package for this job.

It’s not just my background in past academic projects but also my people skills, which will be applicable

to this position.

**Describe a challenging situation at work and how you handled it.**

Answer: In a previous role, we faced a tight deadline and a sudden increase in workload. I reorganized the team, prioritized tasks, and ensured open communication to meet the deadline. It taught me the importance of adaptability, teamwork, and effective time management.

**Describe a situation where you had to work with a difficult team member.**

Answer: In a previous project, a team member had a differing opinion on how to approach a task. I actively listened to their concerns, proposed compromises, and eventually found a solution that aligned with both our perspectives. This experience reinforced my belief in the power of effective communication and collaboration

**How do you handle stress and pressure?**

Answer: I manage stress by breaking down complex tasks into smaller, achievable goals. I also ensure regular breaks to maintain focus and recharge. Additionally, maintaining a healthy work-life balance and engaging in physical activities like exercise and mindfulness helps me manage stress effectively.

.**Why did you change your career?**

While my background lies in Accounting, my passion for technology, problem-solving & my dedication to mastering the field of DevOps, it has motivated me to pursue this exciting career change.

**Why do you want to work for this company?**

Answer: I am impressed by this company's [values/mission/innovations] and the impact it has in the [industry]. I believe my skills align well with the organization's goals, and I am eager to contribute and grow within this dynamic and collaborative environment.

**What are your strengths and weaknesses?**

Answer: One of my strengths is my ability to adapt quickly to new situations and technologies. However, sometimes I tend to get overly involved in a project, which can affect my work-life balance. I'm working on managing this by setting clear boundaries and prioritizing self-care

5**. Describe a situation where you had to work with a difficult team member**. Answer: In a previous project, a team member had a differing opinion on how to approach a task. I actively listened to their concerns, proposed compromises, and eventually found a solution that aligned with both our perspectives. This experience reinforced my belief in the power of effective communication and collaboration.

**6. How do you handle a situation where there's a sudden change in project requirements?** *Answer:* I adapt to sudden changes by quickly assessing the impact on the project, reassessing priorities, and reallocating resources accordingly. I communicate with the team to provide updated instructions and support, ensuring a seamless transition and successful adjustment to the new project requirements

**CICD**

CICD, which stands for Continuous Integration, Continuous Delivery and Continuous Deployment, represents software development and release practices which most of the company's have adopted according to their requirement. Before they used to follow waterfall model and Agile methodology etc Which had an issues

Incase of water model, there will not continuous feedback

Incase of Agile, we have continuous feedback but there is no continuous integration testing

Continuous integration is a process where developer frequently merge the code changes into a shared repository than there will be automated build and unit test

Continuous delivery is a extension of continuous integration where we will mimic few things before production deployment like build, test performance only even everything working fine than there will be a manual approval deploy to production environment

Continuous deployment is a process where it validate code changes or stable using automated test, build and deploy to production without any explicit approval so changes will be directly to deploy on production server in this case.

**The Amazon Relational Database Service (RDS AWS)** is a web service that makes it easier to set up, operate, and scale a relational database in the cloud. It provides cost-efficient, re-sizable capacity in an industry-standard relational database and manages common database administration tasks.

So people often develop a misconception, when they confuse **RDS with a database.**

**RDS is** not **a database, it’s a service that manages databases**, having said that, let’s discuss the databases that RDS can manage as of now:

* o Amazon aurora
* o Mysql
* o PostgreSQL
* o SQL Server
* o Oracle
* o MariaDB

Kubernetes

\*DevOps engineers looking to learn Kubernetes, understanding the following foundational concepts will give you a solid start:\*  
  
1. \*Containers and Microservices\*: Before diving into Kubernetes, ensure you understand what containers are (e.g., Docker) and the principles of microservices.  
  
2. \*Kubernetes Architecture\*:  
  
3. \*Kubernetes Objects\*:  
- Pods: The smallest deployable units in Kubernetes that can be created, scheduled, and managed. Pods can house one or more containers.  
- Deployments: Define the desired state for an application.  
- ReplicaSets: Ensure a specified number of pod replicas are running.  
- Services: An abstract way to expose an application running on a set of pods.  
- ConfigMaps and Secrets: Allow environment-specific configuration separation from application artifacts.  
- Persistent Volumes (PVs) and Persistent Volume Claims (PVCs): Provide storage resources in a cluster.  
  
4. \*Kubernetes Control Plane\*:  
- API Server: Gateway to the Kubernetes control plane.  
- etcd: Consistent and highly available key-value store for all cluster data.  
- Kube-controller-manager: Runs controller processes.  
- Kube-scheduler: Assigns your newly created pods to nodes.  
  
5. \*Kubectl\*: The command-line tool for interacting with a Kubernetes cluster. You'll use this extensively for deploying and managing applications, querying the system state, and more.  
  
6. \*Networking\*:  
- Service Network: Ensures that traffic gets routed to the right pods.  
- Pod Network: Every pod gets its own IP address.  
- Network Policies: Define how pods communicate with each other.  
  
7. \*Storage\*: How Kubernetes allows persistent storage to work with containers which are typically ephemeral.  
  
8. \*Namespaces\*: Used to support multiple virtual clusters within the same physical cluster.  
9. \*Logs & Monitoring\*: Tools like Prometheus for monitoring and Fluentd or ELK stack for logging.  
10. \*Autoscaling\*: Automatically adjust the number of pods or nodes based on traffic or load.  
11. \*Helm\*: A package manager for Kubernetes that helps define, install, and upgrade complex Kubernetes applications.  
12. \*Security\*:  
- Role-Based Access Control (RBAC): Defines what actions a user or system can perform.  
- Network Policies: Control communication between pods.  
- Pod Security Policies: Control sensitive aspects of the pod specification.  
13. \*Ingress and Ingress Controllers\*: Manage external access to services in a cluster, typically HTTP.  
  
14. \*StatefulSets and DaemonSets\*: For workloads that need stable hostnames, persistent storage, or run on every node, respectively.  
15. \*Liveness and Readiness Probes\*: Check the health of your applications and ensure traffic doesn't go to unhealthy instances.  
\_Once familiar with these foundational concepts, you can explore more advanced topics like multi-cluster setups, service meshes (like Istio), and continuous deployment/continuous integration (CI/CD) integrations for Kubernetes  
[Amazon Web Services (AWS)](https://www.linkedin.com/company/amazon-web-services/) [DevOps](https://www.linkedin.com/company/skills-devops/) [Kubernetes Architect](https://www.linkedin.com/company/kubernetes-architect/)

Why Kubernetes?

Kubernetes came into the picture after the Software development teams started

switching from monolithic and microservices architecture to containerization because

of scalability and deployment issues. Containerization does solve the issue of

scalability, downtime and dependency management quite efficiently however there are

still some issues that remain unsolved.

**1] what is the architecture of Kubernetes?**

Kubernetes work on Master-Slaves Architecture. There is a node also called controller node on which k8s is running is called a master node and the container are running on the other node called slaves node.

One of the most asked question is about the Kubernetes architecture and I always tell the answer with multiple ways of explanations like the 4 group theory or the home theory.  
  
The Kubernetes architecture consists of a Master node and the worker nodes as per our design where the master node takes the request coming from the user ( API SERVER ) and process the request and then send the data to ( ETCD )DATABASE , here the brain of the Master is ( CONTROL MANAGER ) which controls all the functions inside the master node.The SCHEDULER is again a component which schedules the deploymet of PODS( APP INSIDE CONATINERS ) based on the availability of the worker nodes.  
  
The Worker node has four components ( KUBELET ) Which talks with the master node and sets the communication between both master and worker node, The ( KUBE PROXY ) which sets all the proxy and network related things and the ( CONTAINER RUNTIME ) which runs and manages data on a host of operating system. The final component is POD ( The three layers POD - CONTAINER - APP ) which is the smallest unit in Kubernetes that helps us to run applications.

**2] What does Control manager? etcd? API Server? Scheduler?**

**Control manager -** The controller manager, runs all the controllers on the Kubernetes Cluster. Although each controller, is a separate process, but to reduce complexity, all the controllers are compiled into a single process. They are as follows: Node Controller, Replication Controller, Endpoints Controller, Service Accounts and Token Controllers

**Etcd** - It is a highly available distributed key value store, which is used to store cluster wide secrets. It is only accessible by Kubernetes API server, as it has sensitive information

**API Server -** It exposes the Kubernetes API. The Kubernetes API is the front-end for Kubernetes Control Plane, and is used to deploy and execute all operations in Kubernetes

**Scheduler -** The scheduler takes care of scheduling of all the processes, Dynamic Resource Management and manages present and future events on the cluster

**5] What is Node port and Cluster IP?**

ClusterIP provides internal service access within the cluster,

NodePort enables access to services from outside the cluster,

**9] What is ingress controller?**

Ingress is a Kubernetes resource that manages external access to services within a cluster. It acts as an API gateway, routing external traffic to specific services based on defined rules. Ingress simplifies the management of HTTP and HTTPS routes, enabling more efficient traffic handling

**10] Explain different types of services in Kubernetes**

Kubernetes offers different types of services,including ClusterIP, NodePort, and LoadBalancer. ClusterIP provides internal service access within the cluster, NodePort enables access to services from outside the cluster, and LoadBalancer exposes services to external traffic, typically in cloud environments.

**14] What is daemon sets?**

A daemon set ensures that all the nodes run a copy of a pod. As nodes are added to the cluster, pods are added to them. As nodes are removed from the cluster, those pods are garbage collected.

Deleting a daemon set will cleanup the pods it created even if in the running node.

**16] What is a namespace in Kubernetes? What if we don't specify any namespace?**

A namespace is a logical space within a clusterthat enables the partitioning of resources. It helps organize and isolate various objects and resources, preventing naming conflicts. If not specified, a pod takes the 'default' namespace by default.

**4. Describe a challenging project where you integrated various DevOps tools.**

* **Situation:** In a project for Mercedes Benz, we onboarded 16 microservices into a CICD pipeline, integrating multiple DevOps tools.
* **Task:** We aimed to achieve end-to-end integration of tools for security, builds, Docker, Ansible, and Kubernetes deployment.
* **Action:** I designed automation scripts in shell and Python to check pipeline conditions, onboarded microservices to Sonarqube, automated GitLab/Sonar/Jenkins APIs, and maintained Jenkins build servers with Docker.
* **Result:** The project streamlined deployment, ensured code quality, and reduced operational overhead.

**5**. Can you explain your role in the Kubernetes Managed Metric System project for Palo Alto Networks?

* Situation: I worked on a project involving the installation of Prometheus on AWS EKS clusters and integrated Grafana for monitoring.
* Task: Our goal was to manage security-related applications and ensure continuous data extraction and monitoring.
* Action: We used Helm charts for management, integrated InfluxDB, and used Prometheus exporters to extract data. Splunk integration helped collect application logs.
* Result: This project enhanced security and monitoring capabilities for Palo Alto Networks' applications**.**

**1] Explain the architecture of Docker**

**Docker uses a client-server architecture**

** Docker Client is a service which runs a command. The command is translated using REST API and is sent to the Docker Daemon (server)**

** Docker Daemon accepts the request and interacts with the operating system in order to build Docker Images and run Docker containers**

** A Docker Image is a template of instruction which is used to create containers**

**2] What is the use of a Dockerfile?**

Is a simple text file, which is going to hold the declarative or instruction or docker commands which you want to dockerise the application & create it as the docker image

Docker File is used for creating Docker Images using the build command

**3] WHAT IS DOCKER?**

Docker is a opensource containerization platform that packages your application and all its dependencies together in the form of Containers to ensure that your application works seamlessly in any environment.

**4] Tell us something about Docker Compose.**

Docker Compose is a YAML file which contains details about the services, networks, and volumes for setting up the Docker application. So, you can use Docker Compose to create separate containers, host them and get them to communicate with each other. Each container will expose a port for communicating with other containers.

**5]. What is Docker Swarm?**

Docker Swarm is native clustering for Docker. It turns a pool of Docker hosts into a single, virtual Docker host. Docker Swarm serves the standard Docker API, any tool that already communicates with a Docker daemon can use Swarm to transparently scale to multiple hosts

**6] What is the lifecycle of a Docker Container?**

This is one of the most popular questions asked in Docker interviews. Docker containers have the following lifecycle:

Create a container

Run the container

Pause the container (optional)

Un-pause the container (optional)

Start the container

Stop the container

Restart the container

Kill the container

Destroy the container

**7] Docker Basic Commands**

The following command gives you information about Docker Client and Server versions:

**$ docker version**

How do you get the number of containers running, paused and stopped

**$ docker ps –a**

If you vaguely remember the command and you’d like to confirm it, how will you get help on that particular command**?**

**$ docker --help**

The above command lists all Docker commands. If you need help with one specific command, you can use the following syntax:

**$ docker <command> --help**

How to login into docker repository?

**$ docker login**

If you wish to use a base image and make modifications or personalize it, how do you do that?

**$ docker pull <image\_name>**

How do you create a docker container from an image?

Pull an image from docker repository with the above command and run it to create a container. Use the following command:

**$ docker run -it -d <image\_name>**

Suppose you have 3 containers running and out of these, you wish to access one of them. How do you access a running container?

**$ docker exec -it <container id> bash**

* 1. **what is the challenge u face as devops engineer how u solved and what steps u take .**

As a DevOps engineer, a significant challenge I faced was streamlining the deployment process. To address this, I implemented a continuous integration/continuous deployment (CI/CD) pipeline, automated testing, and collaborated with the development and operations teams. This reduced deployment time and increased overall system stability

**2 . Tell the project that u worked where your contribution made wider impact across multi departments.**

In a previous project, my DevOps contributions had a broad impact across multiple departments. By introducing a comprehensive monitoring system and optimizing resource utilization, we improved application performance. This positively influenced not only the development team but also operations, support, and even marketing, as the improved performance led to better user satisfaction and increased customer engagement.

* 1. **Tell about a time or situation where you had to influence and communicate to others your way of thinking — who were the audience and how did u chanelize the communication**

In a critical situation, I had to communicate a shift in our infrastructure strategy to a diverse audience, including developers, operations, and management. I organized clear and concise meetings, highlighting the benefits and addressing concerns. Through collaborative discussions and documentation, I successfully influenced the team to adopt the new approach, fostering a more efficient and resilient system.