HUSANPREET **SINGH**

DEVOPS ENGINEER

+91 9914795477

Husanpreet Singh

Husanpreetsingh9914@gmail.com

SUMMARY

Results-driven DevOps Engineer with expertise in implementing and managing infrastructure automation and continuous integration/continuous deployment (CI/CD) processes. Strong knowledge of AWS services, Jenkins, Kubernetes, Terraform, Docker, and Python scripting. Proficient in Linux environments and experienced in building scalable and highly available systems. Committed to optimizing software development processes for increased efficiency and productivity.

EDUCATION

Bachelor of Computer Application

Guru Nanak Dev University Amritsar, Punjab, India ,2020-2023

Senior Secondary School

Sant Giani Gurbachan Singh ji Khalsa Academy , Amritsar , 2018 – 2019

ADDITIONAL SKILLS

- Proficient in shell scripting and automation tools.
- Familiarity with configuration management tools like Ansible and .
- Strong understanding of networking and security principles in cloud environments.
- Experience with monitoring and logging tools such as CloudWatch.

SKILLS

- AWS (EC2, S3, Lambda, DynamoDB, CloudFormation, etc.)
- Jenkins
- AWS CodePipeline
- AWS CodeBuild
- Docker
- Kubernetes
- Helm
- Terraform
- Github
- Python
- Linux
- AWS (IaaS, SaaS, CLI)
- Fargate
- AWS Serverless
- Grafana

PROFESSIONAL EXPERIENCE

DEVOPS ENGINEER ASSOCIATE | Internship

The Entrepreneurship Network
1 Nov 2021 - 1 Feb 2022 | 3 Months

Project Summary: As a key member of the Angular Development Team, I spearheaded transformative initiatives aimed at optimizing development and deployment processes. This involved mastering AWS for hosting applications, refining Git workflows, and implementing robust automation practices.

Key Contributions:

- Implemented robust CI/CD pipelines using Jenkins, resulting in a 60% reduction in deployment times and significantly enhanced software release reliability.
- Leveraged Ansible for infrastructure provisioning, ensuring consistent deployment across environments and minimizing configuration errors.
- Employed Docker for containerization, simplifying dependency management and promoting uniformity between development and production setups.
- Orchestrated Docker containers using Kubernetes, enabling efficient scaling, load balancing, and bolstering application availability.

Achievements:

- Reduced deployment times by 60% through CI/CD automation.
- Ensured standardized deployments, mitigating production issues associated with configuration discrepancies.
- Improved application availability by effectively scaling resources using Kubernetes orchestration.

PROJECTS

- **Project: Two-tier Application Deployment for Scalability and Fault Tolerance**

  - Github , Docker , jenkins , K8s , Helm , EKS , Route-52,

  Situation: Led the deployment initiative for a robust two-tier application built on Flask and MySQL, aiming to handle a significant load of 10,000 concurrent users while adhering to top-tier DevOps methodologies.

  Task:
  - Orchestrated the Dockerization process to encapsulate the application, implementing best practices to ensure compatibility and version control. Images were systematically pushed to DockerHub Repository for meticulous versioning.
  - Automated the setup of a resilient Kubernetes cluster using Kubeadm initially and transitioned seamlessly to AWS EKS with eksctl, focusing on fault tolerance and high availability.
  - Utilized HELM to package intricate Kubernetes Manifest files, streamlining the deployment process onto the AWS EKS infrastructure. Ensured deployment efficiency by configuring a multi-node cluster and integrating Load Balancer functionalities.

  Action:
  - Leveraged Docker and Docker-compose to containerize the application, ensuring portability and encapsulation of dependencies.
  - Automated Kubernetes cluster establishment, initially through Kubeadm, then transitioned seamlessly to AWS EKS with eksctl, streamlining deployment and scaling efforts.
  - Employed HELM to package Kubernetes Manifest files, simplifying the deployment onto the AWS EKS infrastructure.
  - Configured a multi-node cluster setup, optimizing for high availability and resilience against failures. Implemented Load Balancer configurations for efficient traffic distribution.
  - Established a continuous integration and continuous deployment (CI/CD) pipeline utilizing Jenkins and GitHub webhooks. Automated the fetching of code from GitHub and orchestrated the building and deployment of Docker containers using Kubernetes HELM on AWS EKS.

  Result:
  - Achieved a significant enhancement in the application's scalability, successfully accommodating 10,000 concurrent users seamlessly.
  - Slashed downtime by an impressive 60% by leveraging AWS Managed Elastic Kubernetes Service (EKS), ensuring robustness and reliability in the deployment architecture.

- **Project: Serverless Deployment on AWS with Fargate and RDS**

  - AWS, Fargate , RDS

  Situation: Tasked with designing and implementing a serverless project on AWS, leveraging Fargate for container orchestration and RDS for deploying a PostgreSQL database instance.

  Task: Led the end-to-end implementation of a serverless architecture:
  - AWS Infrastructure Design: Strategized and architected a serverless solution on AWS, employing Fargate for container management and RDS for hosting a PostgreSQL database, ensuring optimal performance and scalability.
  - Fargate Container Orchestration: Implemented containerization with AWS Fargate, orchestrating application deployment without the need to manage the underlying infrastructure, enhancing scalability and efficiency.
  - RDS PostgreSQL Deployment: Established and configured a PostgreSQL database instance using Amazon RDS, ensuring data integrity, security, and high availability.

  Action:
  - Configured Fargate task definitions and clusters to efficiently manage and deploy containers, streamlining application scalability and resource utilization.
  - Implemented RDS parameters and settings, ensuring the PostgreSQL database operated optimally in alignment with project requirements.

  Result: Achieved remarkable outcomes through the serverless deployment:
  - Scalable and Efficient Infrastructure: Successfully deployed a serverless architecture on AWS, leveraging Fargate for containerized applications and RDS for a PostgreSQL database, enabling seamless scalability and resource efficiency.
  - Optimized Data Management: Established a robust PostgreSQL database instance on RDS, ensuring data integrity, security, and high availability while meeting project objectives.

  This project showcased expertise in architecting serverless solutions on AWS, effectively utilizing Fargate and RDS for streamlined deployment and optimal performance.

- **Docker infrastructure AWS using Terraform**

  -Terraform, AWS

  - Utilized Terraform to automate the creation of Virtual Private Clouds (VPCs) on cloud platforms like AWS .
  - Defined network architecture, including subnets, route tables, and security groups, ensuring modularity and scalability.
  - Employed Terraform modules to standardize VPC setups across multiple environments and regions.
  - Orchestrated network configurations, such as IP address ranges and connectivity, using Terraform's declarative syntax.
  - Streamlined network provisioning, enabling infrastructure setups and facilitating easy maintenance and updates.