

NARESH AGRAWAL

857-415-8953 | nareshagrawal316@gmail.com | www.nareshagrawal.com | www.linkedin.com/in/naresh-agrawal

EDUCATION

Northeastern University, Boston, USA

Expected August 2021

Master of Science in Information Systems

Shri G.S Institute of Technology and Science (SGSITS), Indore, India

April 2018

Bachelor of Engineering in Electronics & Telecommunication

TECHNICAL SKILLS

Programming languages	Java SE, Java EE (J2EE), Python, Shell Scripting, SQL
Web Technologies	HTML, JSP, CSS, SCSS, Bootstrap, JavaScript, React
Frameworks	Spring Boot, Spring MVC, Hibernate, RESTful APIs, Microservices, Swing, JUnit, Log4J
DevOps Tools	Kubernetes, Amazon Web Services (AWS), Azure, Helm Chart, KOPS, Docker, Scripting, Apache Kafka, Ansible, Terraform, Packer, Jenkins, GitHub Actions, CircleCI, Prometheus, Maven, Git
Databases & Server	MySQL, AWS RDS, MongoDB, DynamoDB, Apache Tomcat, Nginx
Version Control & Tools	GitHub, Bitbucket (Stash), Postman, Apache JMeter, Splunk, JIRA, ServiceNow, Confluence
Operating Systems	Linux, macOS, Windows

WORK EXPERIENCE

SS&C Intralinks, Waltham, USA

June 2020 – Jan 2021

Site Reliability Engineering Co-op (SRE)

- Created a 'Status Page', integrated with monitoring tool (**SaaS**) informing customer about downtime, incident, maintenance of the products and services, easily communicate real-time status to end-users
- Set up **monitoring, alerts**, handled overloads on server and **automated** tasks via **CI/CD** pipeline
- Deployed code updates on **Kubernetes**, worked to roll environment forward and performed release engineering
- Troubleshoot** and escalate **bugs** for Live server product, examining, investigating, and resolving problems to help smooth product **performance** and tracking progress through **Jira, ServiceNow** and Git Repositories
- Created, maintained documentation of systems and processes for existing and new systems on Confluence

PROJECTS

Weather Alert API (AWS, Kubernetes, Helm Chart, Docker, Jenkins, Ansible, Kafka, Prometheus, Java)

- Built **Microservices** based **REST** Spring Boot API, support 10,000 requests per second with an uptime of **99.99%**, deployed **containerized** applications on **Kubernetes** cluster using Helm charts, **Ansible** and **Jenkins** via **CI/CD** pipeline
- Designed Jenkins Pipeline to deploy application on each push to **GitHub** repository, build, push docker images to **Docker** Hub, and perform rolling update style deployments on the Kubernetes cluster
- Used **KOPS** for creating Kubernetes cluster on **AWS**, Apache **Kafka** for communication among microservices, implemented Cluster auto-scaling and Horizontal pod auto-scaling using Metrics Server to handle dynamic load
- Installed **Nginx** Ingress Controller to route traffic on **DNS** and Let's-Encrypt to issue SSL certificate for secure connection, **EFK** stack for logging and **Prometheus** for analyzing and monitoring, visualize it on **Grafana** dashboard

Uber (AWS, Azure, EKS, AKS, Helm charts, Jenkins, Ansible, Terraform, Docker, React, Spring Boot)

- Built Microservices based application, deployed **containerized** frontend app on **AKS** cluster and backend **API** on **EKS** cluster using **Helm** charts, and **Jenkins** via **CI/CD** pipeline
- Developed a dynamic, responsive frontend using **JavaScript** React library running behind Nginx server and backend with **Spring Boot** REST API, persisted data with **AWS RDS**, Integrated Google map API and handled user session with **JWT**
- Designed **CI/CD** pipeline using Jenkins to build, push docker image to **Docker** Hub and perform rolling update style deployments on both **AKS** and **EKS** cluster
- Configured a **distributed** system to support high levels of performance, availability and scalability with an ability to scale automatically to handle dynamic load

Online Book Store (AWS, Terraform, Packer, CircleCI, Spring Boot, Java, Hibernate, RDS)

- Engineered Spring Boot app based on MVC architecture, created **AWS** resource stack using **Terraform** script and deployed application on **EC2** via **CI/CD** pipeline using **CircleCI**
- Executed **Lambda** function to trigger **SES** when a message is published on SNS Topic and performed auto-scaling for EC2 instance using **CloudWatch** alarm, Incorporated **S3** buckets, **AWS RDS** to store object metadata with **SSL/TLS** connection
- Configured **Load Balancer** and security groups to route traffic on **DNS**, attached **SSL certificate** for secure connection, LogGroup for logging all user and system actions and metrics for analyzing and monitoring

MyOrganization (React, JavaScript, Node.JS, Express, MongoDB, Google API)

- Developed a dynamic, responsive frontend using **JavaScript** **React** library, following **MERN** stack
- Implemented features to **schedule**, sync meetings in real-time with google calendar, send out hangout meeting link email
- Incorporated chat feature to interact with other people in the organization, one to one chat or group chat using web socket and authorize user using Google Login (login with Gmail)