Deep Sharma

Phoenix, AZ, USA • (623) 297-7216 • dshar112@asu.edu • sharmacodes.com • linkedin.com/in/deepsharma993

EDUCATION

Master of Science in Computer Science - Arizona State University, Tempe, Arizona **(GPA: 4.0) May 2026**

- *Relevant coursework*: Cloud Computing, Data Processing at Scale, Semantic Web Mining, Advanced Algorithms

Bachelor of Technology in Computer Science Engineering - SRM Institute of Science and Technology, India **(GPA: 3.5) May 2021**

TECHNICAL SKILLS

Programming Languages: Java, Python, C, C++, JavaScript, Kotlin, Bash
Cloud & DevOps: AWS, Azure, GitHub Actions, Terraform, Kubernetes, Jenkins, Ansible, Docker, Git
Web & Databases: React.js, Redux, Node.js, FastAPI, HTML/CSS, Tailwind CSS, MySQL, PostgreSQL, MongoDB
Certifications: Microsoft Certified: Azure Fundamentals (AZ-900), Azure Administrator Associate (AZ-104), Azure Developer Associate (AZ-204), DevOps Engineer Expert (AZ-400), HashiCorp Certified: Terraform Associate (002)

EXPERIENCE

Software Engineer 2 **July 2022 - July 2024**
Capgemini, Pune, India

- Developed a one-click solution using ServiceNow, Python, Terraform, and Azure DevOps to automate cloud resource provisioning, decreasing deployment time by 50%.
- Led a team of two developers for Terraform code remediation project, designed solutions, planned sprints, and reviewed and closed backlogs following project timeline.
- Automated 100 EC2 spin-ups/turn-offs utilizing AWS Lambda, cutting down monthly costs by approximately 40%.
- Deployed Bitbucket Datacenter on AWS leveraging a Blue-Green approach, ensuring 99.99% availability, with AWS CloudFormation, Docker, and EKS (Kubernetes) for RDS.
- Recognized with Capgemini Propeller Award' twice (2021, 2022) for outstanding efforts in upskilling multiple teams in DevOps practices and tools.

Software Engineer **June 2021 - July 2022**
Capgemini, Pune, India

- Designed Azure and AWS Terraform modules to streamline cloud infrastructure setup, accelerating DevOps adoption across organization and cutting time of fresh provisioning by 30%.
- Implemented AWS Lambda to create a ServiceNow incident whenever websites went down, alerting the Ops team to take quick action and reducing website downtime by 25%.
- Established approval gates and policies in 20+ Azure DevOps repositories, enhancing code quality and diminishing errors.
- Documented code design changes, drafted 10+ SOPs, and provided 15+ knowledge transfers (KTs) to operations teams, improving adoption of DevOps practices and infrastructure as code across organization.

Software Engineering Intern **January 2021 - June 2021**
Capgemini, Mumbai, India

- Conducted an in-depth analysis of a countrywide traffic accident dataset covering 49 U.S. states.
- Leveraged Azure Blob Storage for efficient data management. Key contributions include ETL operations with Azure Data Factory and Databricks, and visualizations with Power BI.
- Unveiled trends include monthly accident-prone cities, state-wise accident statistics, and severity ratios based on weather.
- Visualized findings through data visualization widgets, contributing to informed decisions in road safety.

PROJECTS

TerraZure [github.com/hadessharma/terraform-react-node-Azure-deployment**]**

- Empowered users to deploy cloud resources via the Terraform framework, TerraZure offers a user-friendly GUI and forms, allowing for easy deployment management without requiring deep Terraform knowledge.
- Built frontend in React JS with Firebase authentication, and implemented cloud provider authentication (AWS and Azure) using service principals and IAM.
- Adopted NodeJS for backend with an Express server, leveraging Terraform modules, receiving variable input from a frontend form.

Elastic Face Recognition Service on AWS [https://github.com/hadessharma/Elastic-Face-Recognition-Service-on-AWS**]**

- Engineered a multi-tiered, elastic face recognition application on AWS, designed to dynamically scale computing resources based on real-time workload demands.
- Implemented a decoupled architecture using Python, with a web-tier on EC2 to handle HTTP requests, S3 for image storage, and SQS queues to manage communication between the web and application tiers.
- Developed a custom autoscaling controller that dynamically scaled the application tier from 0 to 15 EC2 instances based on request queue depth, achieving processing of 100 concurrent requests in approximately 96 seconds.
- Built the application tier from a custom EC2 AMI equipped with a PyTorch deep learning model to perform model inference for facial recognition.