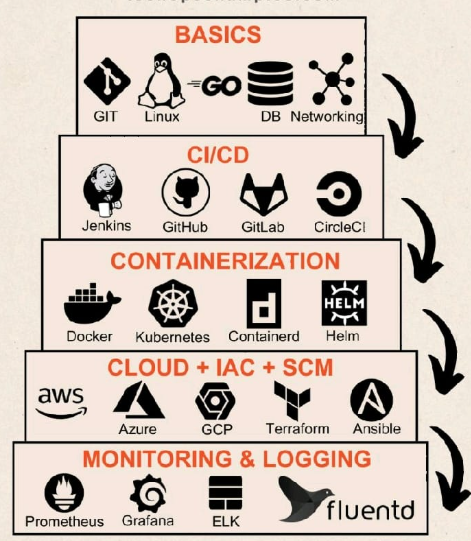
**Roadmap/Guide for DevOps/SRE/Prompt Engineer Career**

Embarking on a career in DevOps, SRE, or Prompt Engineering requires familiarity with a range of tools, technologies, and best practices. Here's a structured roadmap, covering essential skills, tools, and areas of learning, along with references to enhance your understanding.



**1. Agile Methodologies & Tools (Jira, Trello, etc.)**

Agile methodologies focus on iterative development, adaptability, and team collaboration, crucial for both DevOps and SRE roles. You should be familiar with Agile concepts and tools to ensure smooth project management and communication in the development lifecycle.

**Key Areas:**

* **Agile Frameworks**: Scrum, Kanban
* **Project Management Tools**: Jira, Trello, Asana
* **Version Control**: Git, GitHub, GitLab

**Learning Resources:**

* **Video**: [Agile Introduction by Scrum Training Series](https://www.youtube.com/watch?v=3SSOGyWxMek)
* **Article**: Atlassian Agile Coach
* **Tool**: Jira Tutorials

**2. Documentation Tools (Confluence, Markdown, etc.)**

In DevOps, keeping documentation clear and organized is crucial for project transparency and knowledge sharing. Learn how to document processes, decisions, and system architecture.

**Key Areas:**

* **Confluence**: Team collaboration, documentation of processes and decisions
* **Markdown**: Writing simple and efficient documentation
* **Diagrams**: Draw.io, Lucidchart for system design diagrams

**Learning Resources:**

* **Video**: [Confluence Overview by Atlassian](https://www.youtube.com/watch?v=aTq2QkURVtY)
* **Markdown Documentation**: [Markdown Guide](https://www.markdownguide.org/)
* **Tool**: Confluence Learning Resources

**3. Cloud Technologies**

Cloud is at the core of DevOps and SRE roles, with expertise required in managing infrastructure, deploying apps, and automating processes across platforms like AWS, Azure, or GCP.

**Key Areas:**

* **Cloud Providers**: AWS, Azure, Google Cloud Platform
* **Infrastructure as Code (IaC)**: Terraform, CloudFormation
* **Containers & Orchestration**: Docker, Kubernetes

**Learning Resources:**

* **Video**: [AWS Certified Solutions Architect - Full Course](https://www.youtube.com/watch?v=Ia-UEYYR44s)
* **Article**: [AWS Getting Started](https://aws.amazon.com/training/)
* **Tool**: Terraform Learn

**4. Linux/Unix Systems & Commands**

Linux-based systems are prevalent in DevOps environments. You need to be comfortable with system administration tasks, command-line utilities, and shell scripting.

**Key Areas:**

* **Basic Linux Commands**: Navigating the terminal, file management
* **Scripting**: Shell scripting, bash scripting for automation
* **Package Management**: apt, yum, dnf, rpm
* **Monitoring and Logging**: System logs, netstat, top, vmstat, syslog, journald

**Learning Resources:**

* **Video**: [Linux Command Line Basics by Udacity](https://www.youtube.com/watch?v=oxuRxtrO2Ag)
* **Article**: The Linux Command Line
* **Tool**: [Linux Journey](https://linuxjourney.com/)

**5. Programming Languages (Python, Go, Shell, etc.)**

A good understanding of programming is necessary for automating tasks, creating custom solutions, and debugging code. Python and Go are common languages in DevOps, and shell scripting is essential for Linux-based environments.

**Key Areas:**

* **Python**: Automation, scripting, and cloud interactions
* **Go**: Building microservices and efficient applications
* **Bash/Shell Scripting**: Automation and system management

**Learning Resources:**

* **Video**: [Python for DevOps by TechWorld with Nana](https://www.youtube.com/watch?v=QeJ9C8r7fzM)
* **Go Programming Tutorial**: [Go Programming Language Docs](https://golang.org/doc/tutorial/create-module)
* **Shell Scripting Tutorial**: Shell Scripting by TutorialsPoint

**6. Understanding Client Requirements & Stakeholder Communication**

In DevOps and SRE, understanding the client’s business goals and technical requirements is essential for effective project delivery. Effective communication is key for aligning team efforts with client expectations.

**Key Areas:**

* **Requirement Gathering**: Client interaction, business goals
* **Stakeholder Communication**: Regular updates, setting expectations, feedback loops
* **Documentation**: Translating technical details into understandable language for non-technical stakeholders

**Learning Resources:**

* **Video**: [How to Gather Requirements for Software Projects](https://www.youtube.com/watch?v=6JQsqNJ-UMg)
* **Article**: Effective Communication with Stakeholders

**Additional Resources to Explore**

* **CI/CD Tools**: Jenkins, GitLab CI, CircleCI – Automating deployment pipelines.
  + CI/CD Introduction
  + Jenkins Documentation
* **Monitoring and Logging**: Prometheus, Grafana, ELK Stack, Datadog, New Relic.
  + [Prometheus & Grafana Tutorial](https://www.youtube.com/watch?v=h4vQlm3W_Go)
  + [ELK Stack Documentation](https://www.elastic.co/guide/index.html)
* **Security & Compliance**: DevSecOps, penetration testing, security automation.
  + OWASP Top Ten Security Practices
  + [DevSecOps Training](https://www.devsecops.org/)
* **Practice**: Start building your own projects, contribute to open-source, and experiment with the technologies you've learned.
* **Networking**: Join DevOps/SRE communities like Reddit, Stack Overflow, or LinkedIn to stay updated and ask for help.
* **Certifications**: If possible, aim for certifications like AWS Certified DevOps Engineer, Google Professional Cloud DevOps Engineer, or Azure DevOps Solutions Expert.

**List of Tools and Tech Stack we need to be learn or understand**

**1. Agile & Project Management Tools**

These tools help manage tasks, user stories, sprints, and track the development progress.

* **Jira**: Task and issue tracking, project management
* **Trello**: Visual project management with boards and cards
* **Asana**: Task management and project tracking
* **Monday.com**: Work management and project collaboration
* **Confluence**: Documentation, knowledge sharing, and collaboration

**2. Documentation & Collaboration Tools**

You’ll need tools to document projects, processes, and collaborate with the team effectively.

* **Confluence**: Documentation, knowledge sharing (integrates with Jira)
* **Google Docs/Sheets**: Collaborative document creation and management
* **Notion**: Project management, documentation, and notes
* **Markdown**: For creating simple and readable documentation (GitHub Flavored Markdown)

**3. Cloud Platforms & Services**

Cloud technologies are a cornerstone of modern DevOps practices for deploying, managing, and scaling applications.

**Popular Cloud Platforms:**

* **Amazon Web Services (AWS)**: Elastic Compute Cloud (EC2), S3, Lambda, IAM, etc.
* **Microsoft Azure**: Azure App Services, Azure Functions, Azure Kubernetes Service
* **Google Cloud Platform (GCP)**: Compute Engine, Google Kubernetes Engine (GKE), Cloud Functions
* **IBM Cloud**: Cloud Foundry, Kubernetes, Cloud Functions

**Cloud Management Tools:**

* **Terraform**: Infrastructure as Code (IaC) for provisioning and managing cloud resources
* **AWS CloudFormation**: IaC for AWS services
* **Pulumi**: Infrastructure as code using general-purpose programming languages like TypeScript, Python, Go, etc.

**Containerization & Orchestration:**

* **Docker**: Containerization tool to package apps and dependencies
* **Kubernetes**: Container orchestration for automating the deployment, scaling, and management of containerized applications
* **Helm**: Kubernetes package manager
* **Docker Compose**: Defining and running multi-container Docker applications

**4. Version Control & CI/CD**

Version control and Continuous Integration/Continuous Deployment (CI/CD) tools are vital for automating the software delivery pipeline.

**Version Control:**

* **Git**: Distributed version control system
* **GitHub**: Hosting Git repositories and version control with collaboration
* **GitLab**: Git repository management and CI/CD automation

**CI/CD Tools:**

* **Jenkins**: Open-source automation server to implement CI/CD pipelines
* **GitLab CI**: GitLab's integrated CI/CD tool
* **CircleCI**: Continuous integration and delivery tool
* **Travis CI**: Continuous integration service used to build and test code
* **ArgoCD**: Declarative GitOps continuous delivery tool for Kubernetes

**5. Monitoring & Logging**

These tools are essential for maintaining system reliability, visibility, and troubleshooting issues.

* **Prometheus**: Open-source monitoring system with time-series database
* **Grafana**: Visualization tool often paired with Prometheus for visualizing metrics
* **ELK Stack**: Elasticsearch, Logstash, and Kibana for search, logging, and visualization
* **Datadog**: Cloud monitoring and security platform
* **New Relic**: Full-stack observability platform
* **Zabbix**: Enterprise-level monitoring solution
* **Nagios**: IT infrastructure monitoring

**6. Configuration Management & Automation**

Automation and configuration management tools help with infrastructure provisioning, system setup, and management.

* **Ansible**: Configuration management and deployment automation
* **Puppet**: Configuration management tool for automating system configuration
* **Chef**: Infrastructure as code and automation for managing configurations
* **SaltStack**: Configuration management and orchestration tool
* **Vagrant**: Tool for building and managing virtualized development environments

**7. Security & Compliance (DevSecOps)**

Security practices integrated into the DevOps pipeline (DevSecOps) are crucial for maintaining a secure environment.

* **OWASP ZAP**: Open-source security testing for finding vulnerabilities
* **SonarQube**: Continuous inspection of code quality and security vulnerabilities
* **HashiCorp Vault**: Secrets management and data protection
* **Aqua Security**: Container security
* **Snyk**: Security vulnerability scanning for dependencies, Docker images, and Kubernetes

**8. Container Orchestration & Virtualization**

To manage and scale applications, you’ll need knowledge of container orchestration and virtualization technologies.

* **Docker**: Container platform
* **Kubernetes**: Automated container orchestration platform
* **OpenShift**: Kubernetes-based container orchestration platform by Red Hat
* **Docker Swarm**: Native clustering tool for Docker containers
* **VirtualBox**: Open-source virtualization software
* **VMware**: Enterprise-level virtualization

**9. Scripting & Automation Languages**

These languages are essential for automating tasks, system management, and building pipelines.

* **Python**: Popular for automation, scripting, and interacting with cloud APIs
* **Bash**: Shell scripting for automating Linux/Unix tasks
* **Go (Golang)**: For building highly concurrent and efficient microservices
* **Ruby**: Often used with Chef and Puppet for configuration management
* **PowerShell**: Automation scripting for Windows environments

**10. Networking Tools & Technologies**

Understanding networking is essential for configuring infrastructure, security, and communication between services.

* **TCP/IP**: Knowledge of the core networking protocol
* **Nginx**: Web server and reverse proxy for load balancing
* **HAProxy**: High-performance load balancer
* **iptables**: Linux firewall tool for configuring network security
* **Wireshark**: Network protocol analyzer for debugging network issues
* **Consul**: Service discovery and configuration management tool

**11. Database Management & Data Systems**

As a DevOps/SRE, you may need to manage and optimize databases, backup strategies, and scaling solutions.

* **MySQL**: Open-source relational database
* **PostgreSQL**: Advanced open-source relational database
* **MongoDB**: NoSQL database for scalable applications
* **Redis**: In-memory key-value store, often used for caching
* **Elasticsearch**: Full-text search and analytics engine
* **MariaDB**: Fork of MySQL, often used as a drop-in replacement

**12. Logging & Tracing**

Tracing and logging tools help to troubleshoot issues and understand the behavior of applications in production.

* **Elasticsearch**: For storing and searching logs
* **Logstash**: Collects logs and sends them to Elasticsearch
* **Fluentd**: Open-source data collector for unified logging
* **Jaeger**: Distributed tracing for microservices
* **Zipkin**: Distributed tracing system for microservices

**13. Service Mesh & Microservices**

Service meshes help manage microservice-to-microservice communications.

* **Istio**: Service mesh for managing microservices
* **Linkerd**: Lightweight service mesh for Kubernetes
* **Consul**: Service mesh with advanced networking features

**14. Site Reliability Engineering (SRE) Tools**

SRE focuses on maintaining reliability, availability, and performance at scale.

* **PagerDuty**: Incident management and on-call scheduling
* **Opsgenie**: Incident response and alert management
* **VictorOps**: Incident management for DevOps teams
* **ServiceNow**: IT service management (ITSM) platform

**15. Cloud-Native & Serverless Computing**

Cloud-native computing enables flexibility in modern application architectures.

* **AWS Lambda**: Serverless compute service
* **Azure Functions**: Serverless compute service
* **Google Cloud Functions**: Serverless computing on GCP
* **Kubernetes**: Container orchestration for running cloud-native apps
* **Cloud Run (GCP)**: Managed service for running containers in a serverless environment