**🔹 Cheat Sheet 1 — Docker**

| **Concept** | **Key Points / Commands** | **Interview Tip** |
| --- | --- | --- |
| What is Docker? | Containerization platform using container runtime (Moby). | Difference vs VM: Lightweight, shares kernel. |
| Image vs Container | **Image** = blueprint, **Container** = running instance. | Expect a scenario question: “If container fails, how to restart?” |
| Basic Commands | docker run, docker ps, docker stop <id>, docker rm <id> | Remember -d = detached mode. |
| Build & Tag | docker build -t app:v1 . | Always tag images properly. |
| Volumes | docker run -v /host:/container | Used for persistence (DB, configs). |
| Networking | docker network create mynet | Expect: bridge vs host vs overlay. |
| Registry | docker push myrepo/app:v1 | Mention **Docker Hub**, **Harbor**, **ECR**. |
| Dockerfile Basics | FROM, RUN, COPY, CMD | Interviewers like multi-stage builds. |
| Orchestration | Single container = Docker, Multiple = K8s/Swarm. | Be clear Docker Swarm ≠ Kubernetes. |

**🔹 Cheat Sheet 2 — Jenkins**

| **Concept** | **Key Points / Commands** | **Interview Tip** |
| --- | --- | --- |
| What is Jenkins? | CI/CD tool, open-source. | Key = Automates build → test → deploy. |
| Pipeline Types | Scripted (Groovy), Declarative (YAML-like). | Declarative is more common now. |
| Basic Job | SCM (Git) → Build → Test → Deploy | They may ask you to **draw flow**. |
| Plugins | Git, Docker, K8s, SonarQube. | Plugins are Jenkins’ superpower. |
| Jenkinsfile | Example: pipeline { stages { stage('Build') { steps { sh 'mvn clean package' } } } } | Always version Jenkinsfile with code. |
| Triggers | Poll SCM, Webhook, Timer. | GitHub webhook is common. |
| Distributed Build | Master → Agents. | Key for scaling CI/CD. |
| Security | Role-based access, secrets in credentials plugin. | Expect secrets handling Q. |
| Alternatives | GitHub Actions, GitLab CI, ArgoCD. | Show awareness. |

**🔹 Cheat Sheet 3 — Git**

| **Concept** | **Key Points / Commands** | **Interview Tip** |
| --- | --- | --- |
| What is Git? | Distributed version control. | Repo = Local + Remote. |
| Common Commands | git clone, git add ., git commit -m, git push | Be confident with flow. |
| Branching | git checkout -b feature1 | Expect branching strategy Q. |
| Merge vs Rebase | Merge = history preserved, Rebase = cleaner history. | Know difference well. |
| Stash | git stash → temp save changes. | Useful in real-time. |
| Reset vs Revert | Reset = discard history, Revert = new commit undoing changes. | Common tricky Q. |
| Gitflow | Feature → Dev → QA → Prod. | Mention PRs & Code Review. |
| Conflict Resolution | Manual editing → git add → commit. | Share personal experience if asked. |

**🔹 Cheat Sheet 4 — Terraform**

| **Concept** | **Key Points / Commands** | **Interview Tip** |
| --- | --- | --- |
| What is Terraform? | IaC tool (HCL language). | State file is heart of Terraform. |
| Providers | AWS, Azure, GCP, K8s, etc. | Plugin-based architecture. |
| Workflow | init → plan → apply → destroy | Interviewers love lifecycle Q. |
| Variables | variable "region" {} → call with var.region | Useful in modularization. |
| State File | terraform.tfstate tracks infra. | Expect: “What if state is lost?” |
| Remote State | Stored in S3, GCS, etc. | For team collaboration. |
| Modules | Reusable infra components. | Interviewer may ask you to explain your module. |
| Provisioners | remote-exec, local-exec. | Best practice: avoid if possible. |
| Drift Detection | terraform plan shows drift. | Mention it in real-world scenario. |

**🔹 Cheat Sheet 5 — Kubernetes (K8s)**

| **Concept** | **Key Points / Commands** | **Interview Tip** |
| --- | --- | --- |
| What is K8s? | Container orchestration tool. | Declarative vs Imperative. |
| Pod | Smallest deployable unit. | One Pod = one/multiple containers. |
| Deployment | kubectl apply -f deploy.yaml | Rolling updates, rollbacks. |
| Service | ClusterIP, NodePort, LoadBalancer. | Common Q: difference between them. |
| ConfigMap/Secret | Externalize configs. | Secrets base64 encoded, not encrypted. |
| Namespace | Logical isolation in cluster. | Useful in multi-team env. |
| Scaling | kubectl scale deployment myapp --replicas=5 | HPA (Horizontal Pod Autoscaler). |
| Ingress | Controls external traffic to services. | Often asked in interviews. |
| Monitoring | Prometheus + Grafana. | Know basics of observability. |

**🔹 Cheat Sheet 6 — Prometheus & Grafana**

| **Concept** | **Key Points / Commands** | **Interview Tip** |
| --- | --- | --- |
| Prometheus | Metrics collection & alerting. | Pull-based model via exporters. |
| Exporters | Node Exporter, cAdvisor, Blackbox. | Be ready to explain use cases. |
| TSDB | Stores metrics as time-series data. | Interviewers like “time-series DB” Q. |
| Alerts | Alertmanager handles notifications. | Alert → Email/Slack/Webhook. |
| Grafana | Visualization tool (dashboards). | Often used with Prometheus. |
| Data Sources | Prometheus, Loki, Elastic, Influx. | Mention flexible integrations. |
| Query Language | PromQL for Prometheus. | Example: rate(http\_requests\_total[5m]). |
| Use Cases | CPU, Memory, Disk, App performance. | Be ready for monitoring scenario Q. |

**🔹 Cheat Sheet 7 — Ansible**

| **Concept** | **Key Points / Commands** | **Interview Tip** |
| --- | --- | --- |
| What is Ansible? | Config mgmt & automation tool (agentless). | Works via SSH & YAML playbooks. |
| Inventory | Static (hosts.ini) / Dynamic (cloud). | Expect Q on difference. |
| Playbook | Written in YAML. | Idempotent execution is key. |
| Modules | Units of work (package, copy, service). | Example: apt, yum, copy. |
| Roles | Organized way to reuse tasks. | Good practice in big projects. |
| Handlers | Triggered tasks (restart service). | Expect scenario-based Q. |
| Ad-hoc Commands | ansible all -m ping | Quick for troubleshooting. |
| Ansible Vault | Encrypt secrets. | Very common interview Q. |
| Tags | Run specific tasks only. | Use --tags and --skip-tags. |
| Comparison | Terraform = Infra, Ansible = Config. | They often ask this. |