1. DevOps Basics

1.1 What is DevOps?

- Practices and culture combining Development and Operations.
- Goal: faster, reliable delivery and better collaboration.

Notes (your project examples): _____

_____

1.2 Key principles of DevOps

- CI, CD, IaC, automation, monitoring/observability, collaboration.

Notes: _____

_____

2. CI / CD Fundamentals

2.1 What is Continuous Integration (CI)?

- Frequent commits to shared repo; automated build, tests, code analysis, scans.
- Catch issues early and keep code always buildable.

Notes: _____

_____

2.2 Continuous Delivery vs Continuous Deployment

- Delivery: always deployable, needs manual approval for prod.
- Deployment: auto deploy to prod after pipeline passes, no manual step.

Notes: _____

_____

2.3 When to choose Delivery vs Deployment

- Delivery: regulated/high-risk, strong change control.
- Deployment: SaaS/web apps, very mature automation & tests.

Notes: _____

_____

3. Infrastructure as Code (IaC) & Configuration Management

3.1 What is IaC and why it matters?

- Infra defined as code (YAML/HCL/JSON).
- Benefits: consistency, version control, reuse, automation.

Notes: _____

_____

3.2 Terraform vs CloudFormation / ARM

- CloudFormation/ARM: cloud-specific (AWS/Azure).
- Terraform: multi-cloud, rich modules, state mgmt.

Notes: _____

_____

3.3 Configuration management

- Tools: Ansible, Puppet, Chef; manage packages, configs, patches.

Notes: _____

_____

4. Version Control & CI/CD Tools

4.1 Why version control is critical in DevOps

- History, collaboration, rollback, single source of truth.

Notes: _____
_____

4.2 Common CI/CD tools

- Jenkins, GitHub Actions, GitLab CI, Azure Pipelines, AWS CodePipeline.

Notes: _____
_____

5. Containers & Orchestration

5.1 What is Docker?

- Container platform; packages app + dependencies; lightweight, portable.

Notes: _____
_____

5.2 What is Kubernetes and why use it?

- Orchestrates containers: deploy, scale, self-heal, rollouts/rollbacks.

Notes: _____
_____

5.3 Microservices & containers

- Microservices: small, independent services; containers are ideal runtime.

Notes: _____
_____

5.4 Docker Compose vs Docker Swarm vs Kubernetes

- Compose: local/single host multi-container.
- Swarm: simple orchestrator for small/medium setups.
- Kubernetes: full, production-grade orchestration.

Notes: _____
_____

6. Deployment Strategies

6.1 Blue-Green deployment

- Two environments: Blue (current), Green (new).
- Switch traffic when Green is validated; easy rollback.

Notes: _____
_____

6.2 Canary deployment

- Gradual rollout to small % of users, then increase if stable.
- Requires strong monitoring and rollback.

Notes: _____
_____

6.3 Blue-Green vs Canary

- Blue-Green: zero-downtime cutover, more infra.
- Canary: less infra, limited blast radius, gradual.

Notes: _____
_____

7. Monitoring, Logging & Observability

7.1 Monitoring approach

- Metrics for infra, containers, apps; alerts & dashboards.

- Tools: Prometheus+Grafana, DataDog, CloudWatch, Azure Monitor.
Notes: _____
_____

7.2 Logging approach
- Centralized logs: ELK/EFK, Cloud logs; structured JSON logs.
Notes: _____
_____

7.3 What is observability?
- Using metrics, logs, traces to understand system behavior.
Notes: _____
_____

8. DevSecOps & Security
8.1 What is DevSecOps?
- Integrate security into every stage of CI/CD and SDLC.
- SAST, DAST, dependency & image scanning, policy enforcement.
Notes: _____
_____

8.2 Secret management
- Tools: Vault, AWS Secrets Manager, Azure Key Vault, SSM.
- No secrets in code; enforce rotation & least privilege.
Notes: _____
_____

8.3 Securing Docker & Kubernetes
- Docker: minimal images, non-root, scan/sign images.
- K8s: RBAC, NetworkPolicies, Pod security, secrets, audits.
Notes: _____
_____

9. AWS DevOps Services
9.1 Core services
- CodeCommit, CodeBuild, CodeDeploy, CodePipeline.
- CloudFormation, Elastic Beanstalk, ECS, EKS, Lambda.
- CloudWatch, X-Ray, Systems Manager, Secrets Manager, GuardDuty, CloudTrail.
Notes: _____
_____

10. Azure DevOps & Azure Cloud
10.1 Azure DevOps components
- Azure Repos, Pipelines, Boards, Test Plans, Artifacts.
Notes: _____
_____

10.2 Azure DevOps vs traditional Jenkins setups
- Integrated suite vs separate tools; simpler governance, reporting.
Notes: _____
_____

11. Advanced Topics

11.1 What is GitOps?

- Git as single source of truth; tools (ArgoCD/Flux) reconcile state.

Notes: _____
_____

11.2 Handling multi-cloud

- Terraform for IaC; shared standards; central monitoring/logging.

Notes: _____
_____

11.3 Optimizing CI/CD pipelines

- Parallel stages, caching, incremental builds, artifact repos.

Notes: _____
_____

11.4 Incident management & SRE concepts

- Incidents, RCA, SLIs/SLOs, error budgets, post-mortems.

Notes: _____
_____

11.5 RTO & RPO (Disaster Recovery)

- RTO: how fast to restore service.
- RPO: acceptable data loss window.

Notes: _____
_____