Aqui está uma explicação detalhada de cada tópico para facilitar seus estudos para a avaliação.

**1. Compreensão de Conceitos e Cultura DevOps**

* **DevOps: Integração entre desenvolvimento e operações**  
  DevOps é uma abordagem que promove a colaboração entre equipes de desenvolvimento (Dev) e operações (Ops). O objetivo é entregar software de alta qualidade de forma mais rápida e eficiente.
* **Benefícios da automação, colaboração e melhoria contínua**  
  Automação reduz erros manuais, acelera processos e aumenta a eficiência. Colaboração melhora a comunicação entre equipes. Melhoria contínua garante a evolução constante dos processos e produtos.
* **Princípios como entrega de valor, medição e monitoramento**  
  Entrega de valor significa focar no que realmente traz benefícios ao cliente. Medição ajuda a identificar gargalos. Monitoramento assegura a estabilidade das aplicações e a detecção precoce de problemas.

**2. Automação e Integração Contínua**

* **Pipelines de CI/CD com Jenkins e Docker**  
  CI/CD (Continuous Integration/Continuous Deployment) são práticas que automatizam o ciclo de desenvolvimento e entrega de software. O Jenkins é usado para configurar pipelines que integram, testam e implantam software. Docker ajuda a criar ambientes consistentes e isolados.
* **Deploys automatizados com Docker e Docker Compose**  
  Docker cria containers para empacotar e executar aplicações de forma consistente. Docker Compose gerencia múltiplos containers, facilitando a automação de deploys.
* **Scripts em Shell para automação**  
  Shell scripts são usados para executar tarefas repetitivas, como backups, configurações ou deploys, reduzindo tempo e erros.

**3. Habilidades Práticas em Tecnologias DevOps**

* **Configuração de ambiente Linux, WSL, Docker e Git**  
  Aprender a instalar e configurar esses ambientes é essencial para criar um pipeline de desenvolvimento robusto.
* **Manipulação de repositórios Git e configuração de chaves SSH**  
  Git é a ferramenta de controle de versão mais usada. Aprender a criar branches, fazer commits e configurar chaves SSH garante segurança e agilidade no trabalho.
* **Desenvolvimento de aplicações com Flask e front-end básico**  
  Flask é um framework leve para criar APIs e aplicações web. Complementá-lo com HTML, CSS e JavaScript ajuda a criar interfaces de usuário funcionais.

**4. Monitoramento e Observabilidade**

* **Prometheus e Grafana para métricas e visualização**  
  Prometheus coleta métricas, como uso de CPU e memória. Grafana exibe essas métricas de forma gráfica, permitindo monitorar a saúde das aplicações.
* **Garantia de estabilidade em produção**  
  O monitoramento proativo detecta problemas antes que afetem os usuários finais, assegurando estabilidade e performance.

**5. Colaboração e Trabalho em Equipe**

* **Trabalho colaborativo entre desenvolvimento e operações**  
  Práticas como reuniões diárias, feedbacks rápidos e uso de ferramentas integradas como Slack ou Trello melhoram a comunicação entre as equipes.
* **Resolução de problemas de integração e comunicação**  
  É comum haver conflitos durante deploys. Aprenda a usar logs, monitoramento e comunicação eficiente para resolver problemas rapidamente.

**Principais Tópicos que serão cobrados**

**1. Conceitos Gerais**

* **Cultura e metodologia DevOps**  
  Entenda o foco em colaboração, automação e melhoria contínua.
* **Diferenças entre Waterfall, Agile e DevOps**  
  Waterfall segue etapas sequenciais (tradicional), Agile prioriza entregas rápidas e DevOps combina Agile com automação e integração.
* **Benefícios de automação e colaboração**  
  Menor tempo de entrega, redução de erros e maior eficiência no ciclo de vida do software.

**2. Ferramentas e Componentes DevOps**

* **Docker**  
  Ferramenta para criar containers que garantem consistência entre ambientes.
* **Jenkins**  
  Sistema de integração contínua para automação de builds, testes e deploys.
* **Prometheus e Grafana**  
  Soluções para coleta de métricas e visualização, respectivamente.
* **Git**  
  Controle de versão para gerenciar mudanças no código.

**3. Configuração de Ambientes**

* **WSL (Windows Subsystem for Linux)**  
  Permite rodar Linux no Windows. Útil para desenvolvimento DevOps.
* **Docker e Docker Compose**  
  Facilita a criação e gestão de containers e aplicações multicontainer.
* **Git**  
  Aprender comandos básicos como clone, commit, push e pull.

**4. Conceitos de Implementação Prática**

* **Desenvolvimento de APIs com Flask**  
  Flask ajuda a criar APIs para comunicação entre front-end e back-end.
* **Pipelines de CI/CD com Jenkins**  
  Configure pipelines para integrar, testar e entregar código automaticamente.
* **Docker Compose**  
  Gerencie múltiplos containers para aplicações complexas.

**5. Monitoramento e Testes Automatizados**

* **Grafana**  
  Crie dashboards para monitorar métricas do Prometheus.
* **Testes automatizados**  
  Automação de testes para reduzir falhas em produção e garantir qualidade do software.