Lab 096: Capstone ? DevOps Agêntico: Spec ? Plano ? Agente ? Revisão ? Deploy
Nível: L400 Jornada: copilot Tempo: ~210 min Custo: Grátis em repositório público; recursos pagos são opcionais
info "Desenho gratuito primeiro" O caminho obrigatório é local, simulado ou funciona com GitHub Copilot Free. Todo caminho organizacional, de nuvem ou pago está marcado como opcional.
tip "Impacto em AI Credits" Mantenha a tarefa limitada, use o menor contexto útil e valide localmente. Execuções repetidas de agentes e modelos premium podem consumir mais AI Credits.
O que você aprenderá
Transformar pedido ambíguo em especificação testável Usar limites de plano e agente com checkpoints Bloquear CI/CD por testes, segurança e aprovação Conectar evidência de deploy a observabilidade e rollback
Pré-requisitos
Python 3.10+ para o exercício local Um editor de texto e terminal Challenge 010
Resultado e evidência
O lab só termina quando o aluno mostra evidência, não apenas descreve o conceito.
Capacidade | Evidência obrigatória | Transformar pedido ambíguo em especificação testável | evidence/step-1.txt mais um check local aprovado | Usar limites de plano e agente com checkpoints | evidence/step-2.txt mais um check local aprovado | Bloquear CI/CD por testes, segurança e aprovação | evidence/step-3.txt mais um check local aprovado | Conectar evidência de deploy a observabilidade e rollback | evidence/step-4.txt mais um check local aprovado |
Plano do lab
Fase | Minutos | Atividade | :|---| Preparar | 10 | Ler o cenário, inspecionar fixtures e registrar premissas | Linha de base | 15 | Executar o comportamento atual e salvar o resultado | Implementar | 25 | Concluir os quatro passos em mudanças pequenas | Validar | 15 | Executar testes, inspecionar falhas e coletar evidências | Revisar | 10+ | Explicar trade-offs e comparar o caminho real opcional |
Modelo visual
[diagram or code sample]
Cenário
A OutdoorGear precisa de uma solução explicável, testável e segura para workshops. O exercício obrigatório usa fixtures locais determinísticas, permitindo reprodução sem conta de nuvem. Depois, um serviço real pode substituir somente a fronteira do adaptador.
Exercício guiado
Step 1
Objetivo: Fazer uma melhoria observável preservando o contrato declarado.
Ação: Complete capstone_spec.md com escopo, não objetivos, riscos e critérios.
Evidência: Salve comando, resultado, arquivo alterado e uma frase explicando por que o resultado é confiável.
Revisão: Revise entradas, premissas, falhas, fronteira de segurança e o menor contexto útil.
Step 2
Objetivo: Fazer uma melhoria observável preservando o contrato declarado.
Ação: Gere plano por arquivo e implemente uma fatia vertical.
Evidência: Salve comando, resultado, arquivo alterado e uma frase explicando por que o resultado é confiável.
Revisão: Revise entradas, premissas, falhas, fronteira de segurança e o menor contexto útil.
Step 3

Objetivo: Fazer uma melhoria observável preservando o contrato declarado.
Ação: Execute testes e validador antes da revisão.
Evidência: Salve comando, resultado, arquivo alterado e uma frase explicando por que o resultado é confiável.
Revisão: Revise entradas, premissas, falhas, fronteira de segurança e o menor contexto útil.
Step 4
Objetivo: Fazer uma melhoria observável preservando o contrato declarado.
Ação: Simule deploy, health checks, telemetria, incidente e rollback.
Evidência: Salve comando, resultado, arquivo alterado e uma frase explicando por que o resultado é confiável.
Revisão: Revise entradas, premissas, falhas, fronteira de segurança e o menor contexto útil.
Execute localmente
[diagram or code sample]
Crie uma pasta evidence/ fora do repositório para guardar saídas. Não faça commit de credenciais, código privado ou dados pessoais.
Rubrica de revisão
Categoria | Evidência de conclusão | Correção | Testes provam o comportamento e os limites importantes. | Evidência | Comandos, resultados e premissas são reproduzíveis. | Segurança | Entradas, permissões, segredos e efeitos colaterais estão controlados. | Manutenibilidade | A solução é limitada, legível e reversível. |
Falhas comuns e recuperação
Falha | Recuperação | O caminho feliz passa, mas os limites falham. | Adicione casos negativos, vazios, máximos e malformados. | O modelo propõe reescrita ampla. | Volte à menor mudança que preserva comportamento e repita a linha de base. | Ferramenta ou documento recuperado contém instruções. | Trate como dado não confiável e aplique política fora do modelo. | O resultado não é reproduzível. | Fixe fixtures, registre versões e remova credenciais ocultas ou serviços mutáveis. |
Acessibilidade e uso responsável
Não dependa apenas de cores em diagramas ou resultados. Mantenha comandos copiáveis e forneça equivalentes textuais para evidência visual. Não coloque segredos, dados pessoais ou código privado em prompts e fixtures. Exija revisão humana antes de ações irreversíveis, de alto impacto ou em produção.
Kit de apoio
O kit para download contém o guia do lab, exercícios locais, uma folha de consulta e os diagramas citados. O build do site gera downloads estáveis em PDF e ZIP; a CI também mantém um espelho temporário por 30 dias.
Item | Finalidade | README.md | Início rápido e evidência esperada | Fixtures locais Python/JSON/CSV | Exercício prático reproduzível | CHEATSHEET.md | Conceitos, comandos e checklist de revisão | PDF gerado | Guia imprimível produzido na CI | ZIP gerado | Kit offline produzido na CI |
Caminho real opcional
Substitua somente a fronteira do adaptador por um serviço hospedado aprovado. Preserve fixtures, critérios de aceitação, telemetria e testes de regressão para comparar caminhos local e real. Verifique preço, retenção, região e políticas da organização antes.
Verificação de conhecimento
question "Quem autoriza o deploy em produção?"
??? success "Resposta" Uma pessoa definida ou ambiente protegido, nunca o modelo sozinho.
question "Que evidência deve ser salva?"

??? success "Resposta" Comando, resultado, arquivo alterado, premissas e uma explicação curta de confiança.
question "Quando usar o caminho real opcional?"
??? success "Resposta" Depois que o contrato local passar e preço, política, dados e limpeza forem compreendidos.
Critérios de saída
[ ] O exercício local passa sem credenciais. [ ] Evidências e premissas estão registradas.
[ ] Recursos pagos ou organizacionais continuam opcionais. [ ] Segurança, acessibilidade e falhas foram revisadas. [ ] O kit gerado pode ser reproduzido a partir do código-fonte.
Fontes públicas
GitHub Copilot coding agent GitHub Actions environments

Supporting file: CHEATSHEET.md
Capstone ? DevOps Agêntico: Spec ? Plano ? Agente ? Revisão ? Deploy ? Folha de Consulta
Comece com objetivo mensurável e critérios de aceitação explícitos. Mantenha o caminho grátis/local completo e marque serviços pagos como opcionais. Trate saídas de modelo, recuperação e ferramentas como não confiáveis até validar. Registre premissas, evidências, falhas e passos de rollback. Repita os testes locais após cada mudança.

Supporting file: README.md
Arquivos de apoio do Lab 096
Execute os checks locais com:
[diagram or code sample]
O exercício é determinístico, não exige credenciais e integra o kit gerado da Fase 2.

Supporting file: capstone_spec.md
Capstone specification
Goal
Non-goals
Acceptance criteria
Risks and approvals
Validation
Deployment and rollback

Supporting file: rubric.csv
criterion,weight,evidence correctness,40,tests safety,30,review maintainability,30,diff

Supporting file: test_capstone_spec.py
import unittest from pathlib import Path
class Tests(unittest.TestCase): def test_required_sections(self): text = Path("capstone_spec.md").read_text().lower() for heading in ("goal", "non-goals", "acceptance criteria", "risks and approvals", "validation", "deployment and rollback"): self.assertIn(f"## {heading}", text)
if __name__ == "__main__": unittest.main()