Strecht Simulados · Guia de preparação



Guia de preparação ISTQB CT-GenAI

Tudo o que a prova cobra, organizado por domínio: conceitos-chave, comparações que confundem, pegadinhas reais, gatilhos do enunciado, questões comentadas e um plano para chegar pronto ao exame.

5
capítulos por domínio

32
questões comentadas

65%
nota de corte

6
provas completas no curso

ANTES DE COMEÇAR

Como usar este guia

Este guia foi feito para ser estudado junto com as 6 provas completas do curso ISTQB CT-GenAI: o guia explica o que cai e onde estão as armadilhas; as provas treinam o ritmo e mostram o que você ainda não domina.

Quando	O que fazer
Semana 1	Faça o teste diagnóstico (página seguinte) e leia os capítulos 1 e 2. No fim da semana, faça a Prova 1 do curso no tempo do exame.
Semana 2	Leia os demais capítulos, começando pelos domínios em que você errou mais no diagnóstico. Faça as provas 2 e 3.
Semana 3	Releia os alertas de pegadinha e as tabelas de comparação. Faça as provas 4 e 5 e revise cada erro pela explicação.
Semana 4	Estude a folha de cola, faça a Prova 6 e refaça as provas abaixo de 80%. Agende o exame quando estiver acima de 80% em todas (a nota de corte é 65%).

Mapa do exame

Cada prova do curso tem 40 questões e 60 minutos, no formato do exame. Referência de aprovação: 65%.

Domínio (capítulo)	Peso	Neste guia
1. Engenharia de Prompts para Testes de Software Eficientes	27,5%	18 conceitos · 14 pegadinhas
2. Gerenciando os Riscos da AI Generativa em Testes de Software	25%	18 conceitos · 15 pegadinhas
3. Implementação e Integração da AI Generativa em Organizações de Teste	17,5%	16 conceitos · 13 pegadinhas
4. Infraestrutura de teste com tecnologia LLM para Teste de Software	12,5%	16 conceitos · 14 pegadinhas
5. Introdução à AI Generativa para Teste de Software	17,5%	18 conceitos · 14 pegadinhas

Como ler os capítulos: os **tópicos-chave** são o que mais cai; a **tabela de memorização** separa o que a prova costuma confundir; os **alertas de prova** mostram as armadilhas das alternativas erradas; os **gatilhos** ligam palavras do enunciado à resposta; e o **checklist** diz se você está pronto para o domínio.

Estude um pouco todo dia, de graça

Instagram @strecht.simulados

Um post de prova por dia: questão comentada, pegadinhas e o “não confunda” de cada certificação.

Seguir no Instagram →

YouTube Strecht Simulados

Shorts diários com pegadinhas de prova e, aos sábados, um simulado comentado.

Inscrever-se no canal →

DIAGNÓSTICO

Você já sabe isso?

Responda às 12 questões sem consultar nada. As respostas, com a explicação e o capítulo para revisar, estão no fim do guia. Comece o estudo pelos capítulos em que errou.

D1 · Engenharia de Prompts para Testes de Software Eficientes

Um analista pede ao LLM que aponte ambiguidades em uma história de usuário, corrige a lista, envia a versão limpa pedindo a avaliação da testabilidade e, por fim, pede a verificação da completude dos critérios de aceite. Que técnica de prompting ele aplicou?

A. Encadeamento de prompts
B. Prompting de um disparo
C. Meta prompting
D. Prompting de poucos disparos

D2 · Engenharia de Prompts para Testes de Software Eficientes

Em um prompt estruturado para análise de requisitos, qual linha corresponde ao componente instrução?

A. 'Apresente o resultado em tabela com colunas ID, trecho e gravidade.'
B. 'Você é um analista de teste com experiência em sistemas de saúde.'
C. 'Requisitos: RF-01 a RF-12, anexados no arquivo enviado nesta conversa.'
D. 'Identifique ambiguidades nos requisitos abaixo e classifique cada uma.'

D3 · Engenharia de Prompts para Testes de Software Eficientes

O ciclo de teste de uma versão terminou. O gerente quer usar um LLM para apoiar o encerramento das atividades com base nos dados do ciclo. Qual pedido aplica corretamente a AI generativa a essa tarefa?

A. Gerar o plano de teste da próxima versão sem considerar os dados do ciclo encerrado
B. Gerar o relatório de conclusão de teste com sucessos e lições aprendidas do ciclo
C. Gerar um ranking de produtividade dos testadores para anexar ao fechamento do ciclo
D. Gerar novos casos de teste para as funcionalidades que passaram nos testes do ciclo

D4 · Gerenciando os Riscos da AI Generativa em Testes de Software

Uma equipe vai usar um LLM para analisar arquivos de log enviados por sistemas de terceiros. Qual risco a manipulação de solicitações representa nesse uso?

A. O modelo pode gerar código com uma backdoor escondida para o invasor
B. Os dados de treinamento do modelo podem ser alterados por avaliações falsas
C. O modelo pode revelar trechos dos dados usados no seu treinamento
D. Entradas adulteradas podem desviar o resultado da análise feita pela AI

D5 · Gerenciando os Riscos da AI Generativa em Testes de Software

Qual associação entre referência citada no syllabus e seu tipo está correta?

A. ISO/IEC 42001:2023 é norma; Lei de AI da UE é regulamento; NIST AI RMF é estrutura
B. ISO/IEC 42001:2023 é norma; Lei de AI da UE é estrutura; NIST AI RMF é regulamento
C. ISO/IEC 42001:2023 é estrutura; Lei de AI da UE é norma; NIST AI RMF é regulamento
D. ISO/IEC 42001:2023 é regulamento; Lei de AI da UE é norma; NIST AI RMF é estrutura

D6 · Gerenciando os Riscos da AI Generativa em Testes de Software

Segundo o syllabus, o que influencia o consumo de energia de uma tarefa executada com AI generativa no teste de software?

A. A ferramenta de gerenciamento de teste em que o resultado é armazenado
B. O idioma do prompt, já que textos em português consomem mais energia
C. A complexidade da tarefa e os recursos de computação que ela exige
D. A quantidade de testadores que revisa o resultado depois da geração

D7 · Implementação e Integração da AI Generativa em Organizações de Teste

Como as comunidades internas de prática promovem a melhoria contínua no uso de GenAI pelas equipes de teste, segundo o syllabus?

A. Centralizando os prompts em uma pessoa, que os libera conforme a demanda
B. Definindo um único LLM obrigatório e proibindo testes com outros modelos
C. Substituindo o treinamento prático por leituras individuais de artigos
D. Compartilhando bibliotecas de padrões de prompt e lições aprendidas

D8 · Implementação e Integração da AI Generativa em Organizações de Teste

Quais competências principais o syllabus aponta para testadores que trabalham com AI generativa nos processos de teste?

A. Treinar LLMs do zero, ajustar hiperparâmetros e publicar modelos em produção
B. Avaliar recursos do LLM, refinar prompts e avaliar testware gerado por AI
C. Programar redes neurais, rotular grandes bases e otimizar o uso de GPUs
D. Negociar licenças de LLM, gerenciar contratos e aprovar orçamentos de AI

D9 · Implementação e Integração da AI Generativa em Organizações de Teste

Em qual fase da adoção de GenAI a organização de teste treina as equipes nos conceitos de GenAI, dá acesso aos LLMs e experimenta os primeiros casos de uso para criar confiança?

A. Fase de descoberta
B. Fase de monitoramento
C. Fase de utilização e iteração
D. Fase de iniciação e definição de uso

D10 · Infraestrutura de teste com tecnologia LLM para Teste de Software

Uma empresa treina adicionalmente um LLM com pares de histórias de usuário e os respectivos casos de teste aprovados, para que o modelo produza casos no formato e na terminologia da organização. Que prática é essa?

A. Ajuste fino do modelo
B. Meta prompting
C. Geração aumentada por recuperação
D. Prompting de poucos disparos

D11 · Infraestrutura de teste com tecnologia LLM para Teste de Software

Uma organização vai colocar em produção uma ferramenta interna de teste baseada em LLM e precisa de práticas, ferramentas e processos para desenvolver, implantar e manter esse modelo. Como se chama esse conjunto de práticas?

A. Ajuste fino de LLMs
B. Geração aumentada por recuperação
C. Operações de LLM (LLMOps)
D. Engenharia de prompts

D12 · Infraestrutura de teste com tecnologia LLM para Teste de Software

Segundo o syllabus, que mudança os assistentes de AI com agentes baseados em LLM, integrados às ferramentas de teste, trazem para a automação de testes?

A. Retorno à execução manual, pois os agentes substituem as ferramentas de automação atuais
B. Passagem da execução baseada em scripts para uma automação orientada por objetivos
C. Fim da necessidade de verificar resultados, pois os agentes não cometem erros
D. Troca dos casos de teste por respostas de chatbot, sem artefatos de teste gerados

01

CAPÍTULO 1 · 27,5% DA PROVA

Engenharia de Prompts para Testes de Software Eficientes

É o capítulo que mais pesa: 11 questões e 16 pontos, porque os LOs GenAI-2.2.1 a 2.2.5 são K3 e valem 2 pontos cada, e o LO dos componentes do prompt (2.1.1) costuma cair duas vezes. Preciso reconhecer os seis componentes do prompt estruturado, distinguir encadeamento, zero, um e poucos disparos e meta prompting, separar prompt do sistema e do usuário e, principalmente, saber aplicar a GenAI em cada atividade: análise, projeto e implementação, regressão automatizada e monitoramento e controle. Fecha com as sete métricas de avaliação e as técnicas de refinamento. Onde mais erro: confundir contexto com dados de entrada, precisão com recuperação e teste A/B com modificação iterativa, e escolher alternativa que pertence a outra atividade do processo de teste.

Neste capítulo

18 tópicos-chave · 14 pegadinhas · 8 macetes · 14 gatilhos · 4 questões comentadas

Tópicos-chave

Prompt estruturado (seis componentes)

Função, contexto, instrução, dados de entrada, restrições e formato de saída. Cada componente contribui para a clareza e a precisão do pedido. A prova mostra trechos e pede o componente de cada um.

Função

Define a persona que o LLM adota, como testador, gerente de teste ou engenheiro de automação ('Atue como...', 'Você é...'). Serve para o modelo determinar suas responsabilidades e adotar tom e abordagem adequados.

Contexto

Informações básicas sobre o objeto de teste, a funcionalidade a testar e outros dados relevantes para o LLM determinar as condições de teste ('O objeto de teste é o módulo de reembolso...'). Se faltar, a saída fica genérica.

Instrução

Descreve a tarefa de forma clara, imperativa e concisa, com os requisitos relevantes para executá-la ('Identifique ambiguidades nos requisitos abaixo e classifique cada uma'). Redação vaga e hesitante não é boa instrução.

Dados de entrada

As informações a serem processadas: histórias de usuário, critérios de aceite, capturas de tela, casos de teste existentes. Dados detalhados e estruturados geram resultados mais precisos e sensíveis ao contexto.

Restrições

Especificam como as instruções devem ser aplicadas aos dados de entrada: limitações ou considerações especiais ('Não inclua testes de desempenho e use somente dados fictícios', 'Use apenas os campos da especificação').

Formato de saída

Define a estrutura e as características da resposta: tabela com colunas, JSON com campos definidos. Se a resposta vem em parágrafos e precisa ir para planilha, é este componente que se ajusta.

Encadeamento de prompts

Divide a tarefa em etapas intermediárias; a saída de cada etapa é verificada manual ou automaticamente e refinada antes da próxima. Útil em tarefas complicadas que exigem decomposição e checagem sistemática, como analisar ambiguidades, depois testabilidade e depois completude dos critérios.

Zero, um e poucos disparos

Zero disparo não traz exemplos e usa o conhecimento prévio do modelo; um disparo traz um único exemplo; poucos disparos traz mais de um exemplo para consolidar o comportamento desejado. Disparo é sinônimo de exemplo no prompt.

Meta prompting

O LLM gera ou refina o próprio prompt, que o testador avalia e refina em ciclo iterativo. Útil quando não se sabe criar um prompt eficaz; reduz o esforço manual de criar prompts. Não dispensa a avaliação humana e não elimina o não determinismo.

Prompt do sistema x prompt do usuário

O prompt do sistema define comportamento, persona, regras e parâmetros operacionais para toda a sessão; é configurado uma vez e costuma ficar oculto ao usuário. O prompt do usuário é a entrada de cada interação. O LLM considera os dois juntos em cada resposta. Divisão típica: sistema com função, contexto geral e restrições; usuário com o pedido específico e o formato de saída.

GenAI na análise de teste (GenAI-2.2.1)

Identificar defeitos na base de teste (ambiguidades, inconsistências, informação ausente, com perguntas aos stakeholders); gerar condições de teste decompondo requisitos vagos em declarações mensuráveis e testáveis; priorizar por risco fornecendo probabilidade e impacto de falha, aspectos regulatórios e histórico de defeitos; analisar cobertura mapeando requisitos às condições; sugerir técnicas como particionamento de equivalência e análise de valor limite. Prompts multimodais com história, wireframe e regras dos campos geram critérios de aceite.

GenAI no projeto e implementação (GenAI-2.2.2)

Rascunhos de casos de teste a partir de requisitos funcionais e não funcionais; Gherkin com alguns pares aprovados de história e cenário como exemplos; scripts a partir de casos estruturados, informando a ferramenta e as convenções e revisando o resultado; dados sintéticos a partir do esquema e das regras, com valores extremos e sem dados reais; cronograma de execução com dependências, riscos e recursos, verificando o raciocínio; tabela que mostra qual critério de aceite cada caso cobre.

GenAI no teste de regressão automatizado (GenAI-2.2.3)

Análise de impacto com as alterações de código e a rastreabilidade para focar a regressão; scripts de GUI adaptados a novos localizadores, com verificação conforme o risco; regressão de API adaptada pela especificação atualizada; automação orientada por palavras-chave com a documentação da biblioteca e exemplos; relatórios de defeito com registros, capturas e dados do ambiente. Na análise de resultados: comparar com a especificação, agrupar defeitos semelhantes, manter lista de anomalias conhecidas e fazer verificação cruzada antes da conclusão.

GenAI no monitoramento e controle (GenAI-2.2.4)

Sintetizar dados da ferramenta de gerenciamento, inclusive comentários livres, em progresso, tendências, cobertura e riscos, conferindo os números; alertar desvios exige o plano de teste com metas e marcos; no controle, sugerir repriorização, ajuste de cronograma e realocação para o gerente decidir; painéis e resumos em linguagem natural para cada público; ao fim do ciclo, relatório de conclusão com sucessos e lições aprendidas.

Escolha da técnica de prompting (GenAI-2.2.5)

Encadeamento para tarefas complexas que exigem precisão com verificação humana em cada etapa, como análise e projeto de teste; poucos disparos para saídas repetitivas ou de formato restrito (Gherkin, palavras-chave, relatórios em modelo padrão); meta prompting para tarefas flexíveis e novas, como criar o prompt de detecção de anomalias. As técnicas podem ser combinadas no mesmo caso de uso.

Métricas de avaliação (GenAI-2.3.1)

Acurácia: correção geral frente a padrões de especialistas (abrange todos os requisitos). Precisão: correção frente a um objetivo específico (identifica corretamente anomalias). Recuperação: acha todas as instâncias relevantes (partições válidas e inválidas). Relevância e ajuste contextual: aplicável ao contexto (base de teste e regras do domínio). Diversidade: variedade de entradas e casos extremos, sem repetição. Taxa de sucesso na execução: artefatos que rodam como estão. Eficiência de tempo: tempo da AI frente ao esforço manual. Avalie com dados estatisticamente relevantes, manualmente ou automatizando (comparar com referência predefinida).

Técnicas de refinamento (GenAI-2.3.2)

Modificação iterativa (parte de um prompt simples e ajusta a cada rodada), teste A/B (compara versões com métricas predefinidas), análise de resultados (examina a saída e agrupa os tipos de erro), integração do feedback do usuário (opinião dos testadores sobre utilidade e clareza) e ajuste de comprimento e especificidade. Sessões de avaliação e bibliotecas de prompts compartilhadas padronizam a qualidade e evitam erros repetidos.

Macetes para memorizar

💡 Seis componentes

Quem sou, onde estou, o que faço, com o quê, com que limites, como entrego: Função, Contexto, Instrução, Dados de entrada, Restrições, Formato de saída.

💡 Escolha da técnica

Padrão → poucos disparos. Etapas → encadeamento. Novidade → meta prompting. E dá para combinar os três.

💡 Disparos

Disparo = exemplo: zero, um, mais de um.

💡 Sistema x usuário

Sistema é a regra da casa; usuário é o pedido da vez.

💡 Métricas

Acurácia acerta tudo frente ao especialista; Precisão acerta o alvo (anomalias); Recuperação pega todas as partições; Relevância cabe no domínio; Diversidade varia; Taxa de sucesso roda; Eficiência economiza tempo.

💡 Refinamento

Muda aos poucos (iterativa), compara versões (A/B), estuda erros (análise), ouve testadores (feedback), mexe no tamanho (comprimento e especificidade).

💡 Análise de resultados de regressão

Compara, Agrupa, Conhece anomalias, Cruza antes de concluir.

💡 Monitoramento e controle

A GenAI sintetiza e sugere; o gerente confere e decide.

Tabela de memorização: não confunda

Quando é...	...e quando é
Contexto Descreve o objeto de teste e a funcionalidade ('O sistema é um portal de matrículas').	**Dados de entrada** São as informações a processar: história US-87 com critérios, captura da tela, casos existentes.
Restrições Limites para executar a tarefa: 'não inclua', 'use apenas', 'somente dados fictícios'.	**Formato de saída** Forma da resposta: tabela com ID, passos e resultado esperado; JSON com campos.
Prompt do sistema Regras, persona e tom válidos para toda a sessão, definidos uma vez e geralmente ocultos.	**Prompt do usuário** Pergunta ou pedido digitado a cada interação, que muda o tempo todo.
Encadeamento de prompts Já sei decompor a tarefa e quero verificar cada etapa antes da próxima.	**Meta prompting** Não sei como escrever o prompt; peço ao LLM que proponha e eu avalio e refino.
Poucos disparos Muitos artefatos no mesmo padrão (Gherkin, palavras-chave, relatório modelo da empresa).	**Encadeamento de prompts** Tarefa complexa com revisão por especialista a cada resultado intermediário.
Precisão Correção frente a um objetivo específico: os casos identificam corretamente as anomalias.	**Recuperação** Capacidade de achar todas as instâncias relevantes: os casos abrangem as partições válidas e inválidas.
Teste A/B de prompts Várias versões do prompt avaliadas entre si com métricas predefinidas, para ver qual fraseado ou estrutura dá o melhor resultado.	**Modificação iterativa** Um prompt simples melhorado rodada a rodada, observando os resultados.

Números que caem

Item	Valor
Peso no exame	**11 questões e 16 pontos (27,5% das questões)**
LOs K3 do capítulo (2 pontos cada)	**GenAI-2.2.1 a GenAI-2.2.5**
LO com 2 questões por prova	**GenAI-2.1.1 (estrutura do prompt)**
Tempo de estudo do capítulo 2 no syllabus	**365 minutos**
Componentes do prompt estruturado	**6**
Métricas de avaliação listadas	**7 (acurácia, precisão, recuperação, relevância e ajuste contextual, diversidade, taxa de sucesso na execução, eficiência de tempo)**

Alertas de prova (pegadinhas)

⚠ **Alerta de prova:** 'Atue como engenheiro de automação' é contexto, e 'O sistema é um portal de matrículas' é função.
Como não cair: Persona é função; descrição do objeto de teste é contexto. A inversão função/contexto é um distrator clássico nas associações.

⚠ **Alerta de prova:** Um bloco com história, critérios de aceite, captura de tela e casos existentes é contexto.
Como não cair: Se é a informação que o LLM vai processar, são dados de entrada. Contexto só descreve o objeto e a funcionalidade.

⚠ **Alerta de prova:** 'Não inclua testes de desempenho' pertence ao formato de saída ou à instrução.
Como não cair: Excluir tipos de teste ou limitar dados é restrição. A instrução é a tarefa (gerar casos) e o formato é a forma da resposta.

⚠ **Alerta de prova:** Regras de tom formal ficam no prompt do usuário repetido a cada pergunta, ou nos dados de entrada.
Como não cair: O que vale para toda a conversa vai no prompt do sistema, definido uma vez; repetir no prompt do usuário é justamente o que se quer evitar.

⚠ **Alerta de prova:** O prompt do sistema vale só para a primeira resposta, ou é reescrito a cada pergunta, ou as regras ocultas vêm de ajuste fino feito a cada pergunta.
Como não cair: O prompt do sistema permanece constante na sessão e o LLM o considera junto com o prompt do usuário atual. Ajuste fino é treinamento adicional, não acontece por pergunta.

⚠ **Alerta de prova:** Meta prompting dispensa avaliação humana ou garante respostas idênticas.
Como não cair: No meta prompting o testador avalia e refina os prompts gerados; ele não elimina o não determinismo.

⚠ **Alerta de prova:** Para converter 200 casos no mesmo padrão de palavras-chave, use encadeamento com revisão completa de cada script.
Como não cair: O desafio é repetir um formato: poucos disparos com exemplos da biblioteca. Encadeamento é para tarefas complexas com verificação por etapa.

⚠ **Alerta de prova:** Para uma tarefa nova sem prompt pronto, use poucos disparos com exemplos que a equipe ainda não tem.
Como não cair: Sem exemplos e sem saber estruturar a tarefa, comece por meta prompting, descrevendo o objetivo e pedindo ao LLM que proponha o prompt.

⚠ **Alerta de prova:** No meio do ciclo, atrasado, peça o relatório de conclusão; ou deixe o LLM decidir e aplicar sozinho a nova ordem.
Como não cair: Controle de teste com GenAI é pedir sugestões justificadas de repriorização, cronograma e recursos para o gerente decidir. Relatório de conclusão é para o fim do ciclo, e priorização é propensa a erros de raciocínio.

⚠ **Alerta de prova:** Para economizar tokens, envie só os títulos dos casos; ou envie o repositório inteiro para o LLM escolher.
Como não cair: Títulos não têm etapas nem resultados esperados; o repositório inteiro pode exceder a janela e não define o padrão. Envie os casos estruturados ou alguns exemplos selecionados.

⚠ **Alerta de prova:** Para dados de teste, envie apólices reais trocando os nomes, ou peça registros aleatórios sem esquema para não enviesar.
Como não cair: A síntese correta descreve o esquema e as regras e pede registros sintéticos com valores extremos: realistas, relevantes e sem dados pessoais.

⚠ **Alerta de prova:** Peça ao LLM que execute a carga de 500 usuários ou rode o cálculo em produção e informe o resultado.
Como não cair: O LLM gera rascunhos de casos, condições e scripts; ele não executa o sistema. Requisito não funcional também vira caso de teste com pré-condições, carga, entradas e resultados esperados.

⚠ **Alerta de prova:** Testes de GUI falharam porque os localizadores mudaram: marque como defeito do produto ou remova os testes da regressão.
Como não cair: O comportamento está correto; a GenAI adapta os scripts aos novos localizadores, com verificação dos ajustes conforme o risco.

⚠ **Alerta de prova:** Uma única execução do prompt basta para calcular as métricas, e comparar a saída com uma referência predefinida deve ser evitado.
Como não cair: O LLM varia: baseie as métricas em dados estatisticamente relevantes. Comparar com referência predefinida é forma válida de automatizar a avaliação.

Como cai na prova

📝 **Como cai:** Cenário K3: 'Qual abordagem aplica melhor a AI generativa a essa tarefa?' com uma atividade do processo de teste.
Como resolver: 1) Identifique a atividade (análise, projeto e implementação, regressão, monitoramento e controle). 2) Procure a alternativa que dá ao LLM a entrada certa (base de teste, casos estruturados com ferramenta, especificação nova, plano com metas, alterações com rastreabilidade) e mantém a verificação humana. 3) Elimine as que fazem o LLM executar, decidir, aprovar, esconder dados ou que pertencem a outra atividade.

📝 **Como cai:** Associação de trechos de prompt (1-4) com componentes (A-D).
Como resolver: Classifique cada trecho pelos gatilhos ('Atue como', descrição do sistema, 'use apenas', 'responda em tabela'). Confira as inversões típicas: função com contexto e restrições com formato de saída.

📝 **Como cai:** Escolha da técnica de prompting para um cenário (K3, 2 pontos).
Como resolver: Procure a pista dominante: padrão repetitivo, revisão por etapa ou tarefa nova sem prompt. Se o cenário traz as três necessidades, a resposta é a combinação das técnicas.

📝 **Como cai:** Qual métrica corresponde à medição descrita.
Como resolver: Isole o objetivo da medição no enunciado e case com o exemplo do syllabus: anomalias = precisão, partições = recuperação, regras do domínio = relevância, casos extremos e variedade = diversidade, roda sem erro = taxa de sucesso.

📝 **Como cai:** Qual técnica de refinamento foi aplicada.
Como resolver: Veja como o prompt foi melhorado: versões em paralelo com métrica (A/B), rodada a rodada (iterativa), erros da saída agrupados (análise), opinião dos testadores (feedback), tamanho e detalhe (comprimento e especificidade). Consultar especialistas é detecção de alucinação, não refinamento.

📝 **Como cai:** Onde colocar uma orientação: prompt do sistema ou do usuário.
Como resolver: Vale para toda a conversa e não deve ser repetido = sistema. Muda a cada interação = usuário.

Gatilhos do enunciado

Se o enunciado trouxer...	Pense em
'Atue como...', 'Você é um...'	**Função**
'O objeto de teste é...', descrição do sistema e da funcionalidade	**Contexto**

Verbo no imperativo dizendo a tarefa	**Instrução**
'Não inclua', 'use apenas', 'somente dados fictícios'	**Restrições**
'Responda em tabela', 'entregue em JSON com os campos...'	**Formato de saída**
Casos genéricos que ignoram a funcionalidade específica	**Faltou contexto**
Texto definido uma vez, vale para a sessão toda, usuário não vê	**Prompt do sistema**
Etapas sucessivas com revisão a cada passo	**Encadeamento de prompts**
Mesmo padrão toda vez: Gherkin, palavras-chave, relatório modelo	**Poucos disparos**
Ninguém sabe escrever o prompt; tarefa nova como detecção de anomalias	**Meta prompting**
Código alterado e regressão longa demais no CI/CD	**Análise de impacto com alterações de código e rastreabilidade**
Alertar desvios do plano, prever riscos pelas tendências	**Fornecer o plano de teste com metas e marcos**
'Roda como veio', 'partições válidas e inválidas', 'identifica corretamente anomalias', 'regras do domínio', 'casos extremos'	**Taxa de sucesso na execução, recuperação, precisão, relevância e ajuste contextual, diversidade**
Três versões comparadas por métrica; ou opinião dos testadores; ou erros da saída agrupados	**Teste A/B; integração do feedback do usuário; análise de resultados**

🎯 Se você lembrar de só uma coisa deste capítulo...

- Seis componentes: função, contexto, instrução, dados de entrada, restrições, formato de saída.
- Tabela de escolha: poucos disparos para padrão, encadeamento para etapas com revisão, meta prompting para tarefa nova; combinar é permitido.
- Prompt do sistema vale para a sessão inteira e é considerado junto com cada prompt do usuário.
- Em todas as atividades, a GenAI precisa da entrada certa e sua saída é verificada; ela sugere, não decide nem executa o sistema.
- Precisão = anomalias identificadas corretamente; recuperação = todas as partições válidas e inválidas.

Essenciais para a prova (checklist)

- ☐ Sei classificar qualquer trecho de prompt em um dos seis componentes.
- ☐ Sei diferenciar zero, um e poucos disparos, encadeamento e meta prompting por um exemplo.

- ☐ Sei o que vai no prompt do sistema e o que vai no prompt do usuário.
- ☐ Sei aplicar GenAI na análise de teste: defeitos na base, condições, priorização por risco, cobertura e técnicas.
- ☐ Sei aplicar GenAI em projeto e implementação: casos, Gherkin, scripts, dados sintéticos e cronograma.
- ☐ Sei aplicar GenAI na regressão: análise de impacto, localizadores, API, palavras-chave e relatórios de defeito.
- ☐ Sei aplicar GenAI em monitoramento, controle e conclusão do teste.
- ☐ Sei escolher a técnica de prompting adequada a um cenário.
- ☐ Sei reconhecer as sete métricas pelos exemplos do syllabus.
- ☐ Sei distinguir as cinco técnicas de refinamento de prompt.

Questões de revisão do capítulo

R1.1 · Engenharia de Prompts para Testes de Software Eficientes

Uma testadora escreveu três versões de um prompt para gerar casos de teste e quer descobrir qual delas produz os resultados mais precisos e relevantes, usando métricas definidas antes do experimento. Qual técnica de refinamento ela está aplicando?

A. Modificação iterativa do prompt
B. Integração do feedback do usuário
C. Teste A/B de prompts
D. Análise de resultados

R1.2 · Engenharia de Prompts para Testes de Software Eficientes

A história de usuário diz: 'Como cliente, quero cancelar meu pedido rapidamente para não ser cobrado.' Não há critérios de aceite nem prazo de cancelamento definido. A analista quer usar um LLM para identificar defeitos nessa base de teste antes de derivar condições de teste. Qual instrução aplica melhor a AI generativa a essa tarefa?

A. Crie critérios de aceite definitivos para o cancelamento e marque a história como pronta
B. Liste ambiguidades, inconsistências e informações ausentes e proponha perguntas aos stakeholders
C. Gere os scripts de teste automatizados do cancelamento e execute-os no ambiente de homologação
D. Estime o esforço de teste do cancelamento e distribua as tarefas entre os testadores da equipe

R1.3 · Engenharia de Prompts para Testes de Software Eficientes

Uma seguradora precisa de dados de teste para validar o cálculo de prêmios, mas a política interna proíbe copiar dados de produção. O testador vai usar um LLM para isso. Qual abordagem aplica corretamente a AI generativa nessa tarefa de implementação de teste?

A. Enviar uma amostra de apólices reais ao LLM e pedir que ele troque os nomes dos segurados
B. Descrever o esquema e as regras dos dados e pedir registros sintéticos, incluindo valores extremos
C. Pedir ao LLM que busque na internet registros públicos de segurados para usar como dados de teste
D. Solicitar registros aleatórios sem descrever o esquema, para não enviesar a geração dos dados

R1.4 · Engenharia de Prompts para Testes de Software Eficientes

A equipe tem 80 casos de teste manuais bem estruturados e quer gerar scripts automatizados para a ferramenta de automação que já usa. Qual abordagem aplica melhor a AI generativa a essa tarefa de implementação de teste?

A. Enviar os casos estruturados, a ferramenta e as convenções da equipe, e revisar os scripts gerados
B. Enviar só os títulos dos casos de teste, para que o prompt fique menor e mais rápido de ser processado
C. Pedir que o LLM escreva novos casos de teste do zero, descartando os 80 casos já estruturados
D. Pedir scripts genéricos sem informar a ferramenta, para não restringir a criatividade do modelo

Respostas comentadas

R1.1. C) Teste A/B de prompts

Por quê: O teste A/B de prompts compara versões de prompt com métricas predefinidas para descobrir o fraseado ou a estrutura que produz os resultados mais precisos (GenAI-2.3.2).

✗ A) Incorreta. A modificação iterativa parte de um prompt básico e o ajusta aos poucos, sem comparar versões em paralelo.
✗ B) Incorreta. Integrar o feedback do usuário é coletar a opinião dos testadores sobre utilidade e clareza.
✗ D) Incorreta. A análise de resultados examina a saída em busca de imprecisões para entender os tipos de erro.

> **Dica de prova:** Várias versões comparadas por métrica: teste A/B.

R1.2. B) Liste ambiguidades, inconsistências e informações ausentes e proponha perguntas aos stakeholders

Por quê: Na análise de teste, a GenAI pode identificar possíveis defeitos na base de teste, como ambiguidades e informações incompletas, antes de gerar e priorizar condições de teste (GenAI-2.2.1).

✗ A) Incorreta. Critérios definitivos dependem de acordo com os stakeholders; o LLM apoia, mas não decide que a história está pronta.
✗ C) Incorreta. Gerar e executar scripts é implementação e execução de teste, não a análise da base de teste pedida no cenário.
✗ D) Incorreta. Estimar esforço e distribuir tarefas é planejamento, não identificação de defeitos na base de teste.

> **Dica de prova:** Base de teste vaga: peça ambiguidades e perguntas.

R1.3. B) Descrever o esquema e as regras dos dados e pedir registros sintéticos, incluindo valores extremos

Por quê: A síntese de dados de teste com GenAI produz dados sintéticos que se assemelham aos de produção e preservam a privacidade; a qualidade da entrada fornecida ao LLM afeta diretamente o resultado (GenAI-2.2.2).

✗ A) Incorreta. Enviar dados reais ao LLM expõe informações confidenciais, e trocar nomes não remove os demais dados pessoais.

✗ C) Incorreta. Usar registros de pessoas reais obtidos fora da organização cria risco de privacidade e contraria a política citada.

✗ D) Incorreta. Sem o esquema e as regras, os dados gerados tendem a ser pouco relevantes para o cálculo testado.

> **Dica de prova:** Dados de teste com GenAI: sintéticos, realistas e sem dado pessoal.

R1.4. A) Enviar os casos estruturados, a ferramenta e as convenções da equipe, e revisar os scripts gerados

Por quê: Na geração automatizada de scripts, a GenAI interpreta as etapas de casos de teste estruturados e as traduz em código para várias estruturas de automação de teste (GenAI-2.2.2).

✗ B) Incorreta. Títulos não trazem as etapas e os resultados esperados necessários para o script.

✗ C) Incorreta. Os casos estruturados são a melhor base para gerar os scripts.

✗ D) Incorreta. Sem informar a ferramenta, o código gerado tende a não ser compatível com a estrutura usada.

> **Dica de prova:** Caso estruturado mais ferramenta: script gerado e revisado.

02

CAPÍTULO 2 · 25% DA PROVA

Gerenciando os Riscos da AI Generativa em Testes de Software

São 10 questões e 11 pontos: o LO GenAI-3.1.2 (identificar alucinação, erro de raciocínio e viés numa saída) é K3 e vale 2 pontos, e os vetores de ataque (GenAI-3.2.2) costumam cair duas vezes. Preciso separar os três problemas de saída, como detectar e mitigar cada um, o papel da temperatura e da semente, os riscos de privacidade e de segurança, os quatro vetores de ataque com o nome da v1.1 (manipulação do contexto), as medidas de mitigação, o consumo de energia e as quatro referências normativas com seus tipos. As alternativas erradas adoram sugerir 'aumentar a temperatura' ou 'retirar contexto' como solução. Onde mais erro: trocar manipulação do contexto por manipulação de solicitações e confundir ISO/IEC 42001 com ISO/IEC 23053.

Neste capítulo

18 tópicos-chave · 15 pegadinhas · 7 macetes · 14 gatilhos · 4 questões comentadas

Tópicos-chave

Alucinação

Conteúdo factualmente incorreto ou irrelevante para a tarefa. No teste aparece como casos fictícios ou irrelevantes, scripts que não funcionam e testes de critérios de aceite inexistentes, como verificar pagamento com criptomoeda quando o briefing só cita cartão.

Erro de raciocínio

Interpretar mal estruturas lógicas, como causa e efeito, lógica condicional ou solução passo a passo, chegando a conclusões incorretas. Ocorre porque o LLM se baseia em correspondência de padrões, sem raciocínio lógico verdadeiro; planejamento e priorização de testes são tarefas típicas (ordem que viola dependências informadas).

Viés

Resultado que favorece certos tipos de informação ou perspectiva por causa da composição dos dados de treinamento. Exemplos: LLM treinado principalmente em inglês que ignora formatos de nome e endereço de outros idiomas; dados sintéticos que não representam a população da estratégia; tipos de teste não funcionais sub-representados.

Origem dos problemas de saída

Alucinações, erros de raciocínio e vieses resultam da natureza dos dados de treinamento e das limitações inerentes ao modelo transformador. Ficam mais prováveis com prompts mal projetados ou sem dados de entrada contextuais.

Detecção de alucinação

Verificação cruzada com documentação, requisitos e comportamento conhecido, consulta a especialistas no domínio e verificações de consistência (saídas consistentes entre si e com informação conhecida). A escolha da abordagem depende do nível de risco.

Detecção de erro de raciocínio e de viés

Erro de raciocínio: validação lógica e teste de saída, por exemplo executar os scripts gerados contra o objeto de teste e conferir os resultados. Viés: revisar a representatividade dos dados gerados em relação à estratégia de teste e a cobertura dos tipos de teste.

Técnicas de mitigação (GenAI-3.1.3)

Fornecer contexto completo e informações relevantes; dividir prompts em segmentos gerenciáveis (etapas encadeadas verificadas); usar formatos de dados claros e estruturados; selecionar o modelo de GenAI apropriado e treinado para a tarefa; avaliar o prompt em vários LLMs e comparar os resultados. RAG e ajuste fino, do capítulo 4, complementam essas técnicas.

Não determinismo, temperatura e semente (GenAI-3.1.4)

A amostragem na inferência é probabilística e saídas longas variam mais; a reprodutibilidade total não pode ser garantida. Diminuir a temperatura reduz a aleatoriedade, ao custo de menos criatividade e diversidade (resultados mais repetitivos). Definir uma semente aleatória, disponível em algumas implementações, faz o gerador usar a mesma sequência pseudoaleatória e melhora a reprodutibilidade. Também ajuda automatizar parte da verificação da saída com avaliação estruturada e consistente.

Riscos de privacidade de dados

Exposição não intencional (a saída revela dados confidenciais que não foram fornecidos), falta de controle sobre o uso de dados (a ferramenta armazena e processa sem consentimento ou controle) e riscos de conformidade (usar ferramentas sem cumprir normas como o GDPR, com possíveis disputas legais).

Riscos de segurança

Infraestrutura com LLM vulnerável a ataques (violação de dados, acesso não autorizado), exploração de vulnerabilidades do LLM por agentes mal-intencionados para alterar seu comportamento ou extrair informações, e entradas maliciosas intencionais para enganar o modelo e comprometer a acurácia.

Manipulação do contexto

Vetor de ataque que busca extrair dados de treinamento confidenciais, por exemplo com solicitações longas que excedem a janela de contexto ou perguntas elaboradas para obter casos de teste de outro cliente usados no treinamento. Na v1.1 do syllabus, é o novo nome do antigo vetor 'exfiltração de dados'.

Manipulação de solicitações

Introduzir dados que atrapalham o resultado da AI durante o uso. Exemplo do syllabus: imagens que atraem a AI para um contexto diferente e provocam alucinações sobre critérios de aceite; também trechos ocultos em documentos ou logs adulterados enviados para análise.

Envenenamento de dados

Manipulação dos dados de treinamento. Exemplo do syllabus: avaliações falsas ao classificar os resultados de relatórios de teste gerados por AI; também rótulos adulterados que marcam falhas graves como cosméticas ou histórico de defeitos falso usado no treino.

Geração de código malicioso

Manipular o LLM para gerar backdoors no código produzido durante o uso, como chamadas ocultas a comandos externos ou canal de comunicação com IP externo desconhecido em um script de teste.

Medidas de privacidade e segurança (GenAI-3.2.3)

Minimização de dados (evitar dados confidenciais, salvo permissão legal, e usar só o necessário), anonimização e pseudonimização, armazenamento e transmissão seguros (criptografia forte e controles de acesso) e treinamento de recursos (programas e políticas para uso responsável). Estratégias adicionais: revisão sistemática dos resultados, avaliação por comparação com outro LLM, ambiente operacional seguro (oferta comercial segura, nuvem segura ou infraestrutura própria), auditorias regulares de segurança e avaliações de vulnerabilidade, e manter-se atualizado com as práticas recomendadas de segurança. As medidas se complementam e devem ser combinadas; recomenda-se envolver engenheiros de segurança sênior, o jurídico, o CTO ou o CISO.

GDPR e GenAI

Os regulamentos de proteção de dados não restringem a GenAI de forma explícita, mas impõem salvaguardas como legalidade e finalidade. A origem do fornecedor não dispensa as salvaguardas.

Impacto ambiental (GenAI-3.3.1)

O consumo de energia cresce com o uso e com a complexidade da tarefa. Gerar uma imagem pode consumir tanta energia quanto carregar totalmente um smartphone; gerar texto consome pequena porcentagem disso. Usar LLM como serviço web também aumenta a carga em dispositivos, redes e data centers; o efeito cumulativo gera emissões significativas de CO2, mesmo com dados precisos difíceis de obter. Limitar interações desnecessárias entre modelos reduz o impacto.

Referências normativas (GenAI-3.4.1)

ISO/IEC 42001:2023: norma com requisitos para o sistema de gerenciamento de AI na organização (consistência e confiabilidade). ISO/IEC 23053:2022: norma (o syllabus em português a lista como 'Padrão') que dá estrutura para sistemas de AI com machine learning (ciclo de vida, tolerância a falhas, transparência, qualidade de dados). Lei de AI da UE: regulamento que classifica aplicações por nível de risco (transparência, responsabilidade, mitigação de viés). NIST AI RMF 1.0: estrutura dos Estados Unidos para gerenciar riscos de AI (justiça, transparência, segurança, imparcialidade). As organizações devem acompanhar sua evolução.

Macetes para memorizar

💡 Três problemas de saída

Inventa (alucinação), erra a lógica (raciocínio), favorece um lado (viés).

💡 Detecção

Alucinação → cruza com a documentação. Raciocínio → valida a lógica e executa. Viés → confere a representatividade.

💡 Vetores de ataque

Extrair, Desviar, Envenenar, Plantar: manipulação do contexto, manipulação de solicitações, envenenamento de dados, geração de código malicioso. Treino adulterado = envenenamento; treino vazado = contexto.

💡 Riscos de privacidade

Expõe, Perde o controle, Descumpre a lei.

💡 Referências

ISO = norma, UE = regulamento, NIST = estrutura. '42 gerencia a organização; 23 cuida do ciclo de vida do ML'.

💡 Temperatura

Fria repete, quente varia. Nunca esquente para corrigir alucinação.

💡 Energia

Imagem = smartphone cheio; texto = pequena porcentagem disso; milhões de usos = muito CO2.

Tabela de memorização: não confunda

Quando é...	...e quando é
Alucinação Algo inventado ou irrelevante: recurso, critério ou funcionalidade que não existe.	**Erro de raciocínio** Os itens existem, mas a lógica falha: ordem que viola dependências, causa e efeito, cálculo.
Viés Composição desequilibrada vinda do treino: dados de um só país, só testes funcionais quando a estratégia pede outros tipos.	**Alucinação** Conteúdo fictício que não está no briefing ou nos requisitos.
Manipulação do contexto O atacante quer EXTRAIR dados de treinamento confidenciais.	**Envenenamento de dados** O atacante ADULTERA os dados de treinamento (avaliações falsas, rótulos trocados).
Manipulação de solicitações Entrada adulterada durante o uso para DESVIAR o resultado (imagem, trecho oculto, log).	**Manipulação do contexto** Pedidos elaborados para VAZAR o que estava no treinamento.
Exposição não intencional A saída revela dados confidenciais que o usuário não forneceu.	**Falta de controle sobre o uso de dados** A ferramenta guarda e processa os prompts sem consentimento ou controle da empresa.
ISO/IEC 42001:2023 Requisitos de sistema de gerenciamento de AI na organização; governança, consistência e confiabilidade.	**ISO/IEC 23053:2022** Estrutura para sistemas de AI com machine learning; ciclo de vida, tolerância a falhas e transparência.
Lei de AI da UE Regulamento europeu que classifica aplicações por nível de risco.	**NIST AI RMF 1.0** Estrutura (não regulamento) dos EUA para gerenciar riscos de AI, focada em justiça, transparência e segurança.

Números que caem

Item	Valor
Peso no exame	**10 questões e 11 pontos (25% das questões)**
LO K3 do capítulo (2 pontos)	**GenAI-3.1.2 (identificar alucinações, erros de raciocínio e vieses)**
LO com 2 questões por prova	**GenAI-3.2.2 (vetores de ataque)**
Tempo de estudo do capítulo 3 no syllabus	**160 minutos**
Vetores de ataque na v1.1	**4 (manipulação do contexto, manipulação de solicitações, envenenamento de dados, geração de código malicioso)**
Referências normativas citadas	**ISO/IEC 42001:2023, ISO/IEC 23053:2022, Lei de AI da UE, NIST AI RMF 1.0**

Energia para gerar uma imagem	**Pode equivaler a carregar totalmente um smartphone**

Alertas de prova (pegadinhas)

⚠ **Alerta de prova:** Aumentar a temperatura para o modelo 'explorar mais possibilidades' reduz alucinações e erros.
Como não cair: Temperatura mais alta aumenta a aleatoriedade e a variação, e tende a piorar a invenção de regras. Nenhuma técnica de mitigação do syllabus sobe a temperatura.

⚠ **Alerta de prova:** Pedir ao mesmo LLM que confirme a própria resposta é a forma de achar os erros.
Como não cair: A técnica do syllabus é avaliar o prompt em vários LLMs e comparar os resultados.

⚠ **Alerta de prova:** Retirar o contexto ou encurtar o prompt evita respostas enviesadas e deixa o modelo mais objetivo.
Como não cair: Sem contexto, o risco de alucinação aumenta. A mitigação é contexto completo em formato claro e estruturado.

⚠ **Alerta de prova:** Respostas mais longas dão espaço para o modelo se corrigir.
Como não cair: Saídas longas variam mais e não corrigem o raciocínio; divida a tarefa em etapas verificadas.

⚠ **Alerta de prova:** Baixar a temperatura causa mais alucinações, janela menor ou necessidade de retreino.
Como não cair: O efeito colateral de temperatura baixa é menos criatividade e diversidade, com resultados mais repetitivos. A temperatura é ajustada na inferência.

⚠ **Alerta de prova:** Fixar temperatura e semente garante reprodutibilidade total com o mesmo prompt e modelo.
Como não cair: A reprodutibilidade total não é garantida; temperatura e semente apenas reduzem a variação.

⚠ **Alerta de prova:** Atacante que tenta fazer o modelo revelar dados do treino está fazendo envenenamento de dados.
Como não cair: Envenenamento adultera o treino; extrair dados do treino é manipulação do contexto (antes chamada exfiltração de dados).

⚠ **Alerta de prova:** Trechos ocultos num documento de requisitos enviado ao LLM são envenenamento de dados.
Como não cair: O documento é entrada do uso, não dado de treinamento: manipulação de solicitações.

⚠ **Alerta de prova:** LLM treinado principalmente em inglês que ignora formatos de outros idiomas é envenenamento ou alucinação.
Como não cair: Não há ação intencional nem item inventado: é viés vindo da composição dos dados de treinamento.

⚠ **Alerta de prova:** Ordem CT-03, CT-01, CT-07 quando CT-07 depende de CT-03 que depende de CT-01 é viés por preferir números baixos.
Como não cair: A sequência quebra dependências informadas: erro de raciocínio, detectado por validação lógica.

⚠ **Alerta de prova:** Revisar a representatividade dos dados em relação à estratégia detecta alucinações ou erros de raciocínio.
Como não cair: Representatividade é detecção de viés. Alucinação usa verificação cruzada com documentação; raciocínio usa validação lógica e execução dos scripts.

⚠ **Alerta de prova:** Minimização de dados resolve prompts guardados sem proteção e trafegando em texto aberto.
Como não cair: Para dados armazenados e em trânsito a medida é armazenamento e transmissão seguros, com criptografia forte e controle de acesso.

⚠ **Alerta de prova:** Um LLM público com prompt do sistema proibindo guardar dados, ou conta pessoal com histórico apagado, atende alta confidencialidade.
Como não cair: Instrução em prompt não é controle de segurança e conta pessoal é Shadow AI. As opções são oferta comercial segura, nuvem segura ou infraestrutura própria.

⚠ **Alerta de prova:** Cada consulta de texto gasta a energia de carregar um smartphone, ou o consumo acontece só no treinamento, ou usar como serviço web elimina o consumo.
Como não cair: É a geração de imagem que pode equivaler a uma carga de smartphone; o uso também consome energia em dispositivos, redes e data centers, e o efeito cumulativo é grande.

⚠ **Alerta de prova:** NIST AI RMF é regulamento e a Lei de AI da UE é estrutura; ou ISO/IEC 42001 trata do ciclo de vida de ML.
Como não cair: ISO é norma (na tradução, a 23053 aparece como 'Padrão', mesmo tipo), UE é regulamento, NIST é estrutura. Ciclo de vida de AI com ML é a ISO/IEC 23053; sistema de gerenciamento de AI é a ISO/IEC 42001.

Como cai na prova

📝 **Como cai:** K3 (2 pontos): o enunciado mostra a saída de um LLM e pergunta o que o testador deve identificar.
Como resolver: Compare a saída com o que foi informado: item inventado = alucinação; itens certos em ordem ou lógica errada = erro de raciocínio; composição desequilibrada frente à estratégia = viés. Desconfie da alternativa 'saída correta' e das que citam envenenamento sem atacante.

📝 **Como cai:** Cenário de ataque: qual vetor foi usado?
Como resolver: Pergunte duas coisas: o alvo são os dados de treinamento ou a entrada enviada durante o uso? O objetivo é extrair, adulterar, desviar ou plantar backdoor? Pedidos para extrair dados do treinamento = manipulação do contexto; adulterar os dados de treinamento = envenenamento de dados; entrada adulterada no uso para desviar o resultado = manipulação de solicitações; levar o LLM a gerar backdoor no código = geração de código malicioso.

📝 **Como cai:** Qual técnica de mitigação trata a causa descrita (ou 'Selecione DUAS').
Como resolver: Case a causa com a técnica: prompt sem detalhes → contexto completo; plano grande num prompt só → segmentos verificados; modelo genérico em linguagem incomum → modelo apropriado; incerteza sobre a resposta → comparar entre LLMs. Elimine temperatura alta, respostas longas, retirar contexto ou exemplos.

📝 **Como cai:** Qual risco de privacidade ou qual medida de proteção se aplica.
Como resolver: Identifique onde está o dado: na saída (exposição), na ferramenta sem consentimento (falta de controle), contra a lei (conformidade). Medidas: menos dados (minimização), dados mascarados (anonimização/pseudonimização), dados guardados e trafegando (criptografia e controle de acesso).

📝 **Como cai:** Qual referência normativa e qual o seu tipo.
Como resolver: Primeiro o tipo (norma, regulamento, estrutura), depois o foco: gerenciamento (42001), ciclo de vida de ML (23053), níveis de risco (UE), justiça e transparência nos EUA (NIST).

Gatilhos do enunciado

Se o enunciado trouxer...	Pense em
Caso de teste de recurso que não está nos requisitos ou no briefing	**Alucinação**
Ordem que quebra dependências, causa e efeito mal interpretados	**Erro de raciocínio (validação lógica, executar e conferir)**
Dados que não representam o público, tipo de teste ausente, treino concentrado em um idioma	**Viés (revisar representatividade frente à estratégia)**

Menor variação possível entre execuções do mesmo prompt	**Diminuir a temperatura**
Reproduzir a geração da semana passada	**Definir semente aleatória**
Solicitações longas que excedem a janela ou perguntas para vazar dados do treino	**Manipulação do contexto**
Imagem, documento ou log adulterado que desvia a análise	**Manipulação de solicitações**
Avaliações falsas em massa, rótulos de treino alterados	**Envenenamento de dados**
Chamada oculta a comando externo ou IP desconhecido no script gerado	**Geração de código malicioso**
Resposta traz dados de clientes que ninguém forneceu	**Exposição não intencional de dados**
Ferramenta gratuita guarda e processa os prompts sem consentimento	**Falta de controle sobre o uso de dados**
Dados pessoais processados sem verificar o GDPR	**Risco de conformidade, com disputas legais**
Três LLMs trocando mensagens mesmo após aprovação	**Limitar interações desnecessárias entre modelos**
Norma de sistema de gerenciamento de AI; regulamento por nível de risco; estrutura americana de risco	**ISO/IEC 42001:2023; Lei de AI da UE; NIST AI RMF 1.0**

🎯 Se você lembrar de só uma coisa deste capítulo...

- Alucinação inventa, erro de raciocínio erra a lógica, viés vem da composição dos dados de treinamento.
- Temperatura menor e semente reduzem a variação, mas a reprodutibilidade total nunca é garantida.
- Na v1.1, extrair dados de treinamento se chama manipulação do contexto (antes exfiltração de dados).
- As medidas de privacidade e segurança são complementares e devem ser combinadas; o GDPR não proíbe GenAI.
- ISO/IEC 42001 = gerenciamento; ISO/IEC 23053 = ciclo de vida de ML; Lei da UE = regulamento por risco; NIST AI RMF = estrutura.

Essenciais para a prova (checklist)

- ☐ Sei classificar uma saída de LLM como alucinação, erro de raciocínio ou viés.
- ☐ Sei a forma de detecção adequada a cada um dos três problemas.
- ☐ Sei as cinco técnicas de mitigação de 3.1.3 e que RAG e ajuste fino as complementam.

- ☐ Sei o efeito da temperatura e da semente e o efeito colateral de baixar a temperatura.
- ☐ Sei distinguir os três riscos de privacidade e os riscos de segurança.
- ☐ Sei associar os quatro vetores de ataque a exemplos, usando o nome da v1.1.
- ☐ Sei as medidas de privacidade e as estratégias adicionais, incluindo as opções de ambiente operacional seguro.
- ☐ Sei o que influencia o consumo de energia e a comparação entre gerar imagem e texto.
- ☐ Sei o tipo e o foco de cada uma das quatro referências normativas.

Questões de revisão do capítulo

R2.1 · Gerenciando os Riscos da AI Generativa em Testes de Software

Um atacante envia a uma ferramenta de teste com LLM solicitações extremamente longas, que excedem a janela de contexto, para tentar fazer o modelo revelar trechos dos seus dados de treinamento. Qual vetor de ataque é esse?

A. Envenenamento de dados
B. Manipulação de solicitações
C. Geração de código malicioso
D. Manipulação do contexto

R2.2 · Gerenciando os Riscos da AI Generativa em Testes de Software

A estratégia de teste de uma API bancária exige testes funcionais, de desempenho e de segurança. O LLM gerou 60 casos de teste, todos funcionais. O que o testador deve identificar nessa saída?

A. Uma saída completa, pois testes funcionais cobrem a segurança
B. Uma alucinação, pois testes funcionais não existem para APIs
C. Um erro de raciocínio, pois o total de casos deveria ser par
D. Um viés, com tipos de teste não funcionais sub-representados

R2.3 · Gerenciando os Riscos da AI Generativa em Testes de Software

Durante a revisão de scripts de automação gerados por um LLM, a equipe nota uma chamada escondida que baixa e executa um comando externo a cada execução da suíte. Considerando que o modelo foi manipulado, qual vetor de ataque isso indica?

A. Manipulação do contexto
B. Geração de código malicioso
C. Manipulação de solicitações
D. Envenenamento de dados

R2.4 · Gerenciando os Riscos da AI Generativa em Testes de Software

Uma equipe europeia usa uma ferramenta de GenAI para processar dados pessoais de clientes em testes, sem verificar as exigências do Regulamento Geral de Proteção de Dados (GDPR). Qual risco de privacidade de dados está mais evidente?

A. Risco de alucinação nos casos de teste gerados
B. Risco de erro de raciocínio na priorização
C. Risco de envenenamento dos dados de treinamento
D. Risco de conformidade, que pode levar a disputas legais

Respostas comentadas

R2.1. D) Manipulação do contexto

Por quê: Na versão 1.1 do syllabus, o vetor que busca extrair dados de treinamento confidenciais chama-se manipulação do contexto (GenAI-3.2.2).

✗ A) Incorreta. O envenenamento de dados manipula os dados de treinamento; aqui o atacante tenta extraí-los.
✗ B) Incorreta. A manipulação de solicitações introduz dados que atrapalham o resultado da AI, como imagens que desviam o contexto.
✗ C) Incorreta. A geração de código malicioso induz o LLM a produzir backdoors.

> **Dica de prova:** Sobrecarregar a janela para vazar treino: manipulação do contexto.

R2.2. D) Um viés, com tipos de teste não funcionais sub-representados

Por quê: Para detectar vieses, revisa-se a representatividade do material gerado em relação à estratégia de teste e a cobertura dos tipos de teste (GenAI-3.1.2).

✗ A) Incorreta. Desempenho e segurança são características não funcionais que precisam de testes próprios.
✗ B) Incorreta. Os casos funcionais são válidos; o problema é a ausência dos demais tipos.
✗ C) Incorreta. Não há regra sobre a quantidade de casos ser par.

> **Dica de prova:** Faltou tipo de teste exigido: viés.

R2.3. B) Geração de código malicioso

Por quê: O syllabus exemplifica a geração de código malicioso com backdoors, como chamadas de comandos externos, inseridas no código gerado durante o uso (GenAI-3.2.2).

✗ A) Incorreta. A manipulação do contexto busca extrair dados de treinamento.
✗ C) Incorreta. A manipulação de solicitações atrapalha o resultado com entradas adulteradas, sem foco em backdoors.
✗ D) Incorreta. O envenenamento manipula dados de treinamento; o sintoma descrito é um backdoor no código gerado.

> **Dica de prova:** Comando externo escondido: geração de código malicioso.

R2.4. D) Risco de conformidade, que pode levar a disputas legais

Por quê: As preocupações de privacidade citadas são exposição não intencional, falta de controle sobre o uso de dados e riscos de conformidade, como o descumprimento do GDPR (GenAI-3.2.1).

✗ A) Incorreta. Alucinação é um problema de qualidade da saída, não de conformidade com proteção de dados.

✗ B) Incorreta. Erro de raciocínio é um defeito lógico da saída.

✗ C) Incorreta. Envenenamento é um ataque aos dados de treinamento.

> **Dica de prova:** Ignorou o GDPR: risco de conformidade.

03

CAPÍTULO 3 · 17,5% DA PROVA

Implementação e Integração da AI Generativa em Organizações de Teste

São 7 questões e 7 pontos sobre o lado organizacional: riscos da Shadow AI, aspectos da estratégia de GenAI, critérios de seleção de LLMs e SLMs, as três fases de adoção, as habilidades essenciais, como criar capacidade nas equipes e como mudam os papéis de testadores e gerentes de teste. É conteúdo de listas, e a prova testa se eu reconheço o item certo pela pista do cenário. Os distratores costumam ser extremos (proibir, substituir pessoas, treinar modelos do zero, decidir por popularidade) ou nomes de fase que não existem. Onde mais erro: confundir as fases descoberta e iniciação e definição de uso, e trocar custo recorrente por comunidade e suporte.

Neste capítulo
16 tópicos-chave · 13 pegadinhas · 7 macetes · 14 gatilhos · 4 questões comentadas

Tópicos-chave

Shadow AI

Uso de GenAI sem aprovação ou supervisão formal da organização, como testadores usando contas pessoais de ferramentas de AI no trabalho.

Riscos da Shadow AI

Fragilidades de segurança e privacidade (violação dos dados enviados a ferramentas sem segurança robusta), problemas de conformidade e regulamentação (possíveis consequências legais) e propriedade intelectual vaga (licenças pouco claras, risco de disputas).

Como evitar a Shadow AI

Definir uma estratégia e etapas para integrar e implantar a GenAI. O syllabus propõe adoção estruturada, não proibição nem liberdade total de escolha.

Aspectos da estratégia de GenAI (GenAI-5.1.2)

Começa por objetivos de teste mensuráveis, como aumento da produtividade, redução dos ciclos de teste e melhoria da qualidade do teste; seleção de LLMs alinhada aos objetivos, compatível com a infraestrutura e escalável; qualidade dos dados de entrada (precisos, relevantes e protegidos); treinamento técnico e ético das equipes; métricas sobre os resultados; diretrizes de processo.

Diretrizes de processo

Regras para dados confidenciais, obrigações de transparência sobre o que foi gerado com GenAI e portas de qualidade com revisão do testware gerado. Servem para garantir conformidade regulatória e diretrizes éticas.

Critérios de seleção de LLMs/SLMs (GenAI-5.1.3)

Desempenho do modelo, potencial de ajuste fino, custo recorrente e comunidade e suporte. Como poucos benchmarks públicos são voltados a teste de software, a seleção exige avaliação cuidadosa; diferenças de recursos funcionais, como suporte a entrada multimodal, também pesam.

Desempenho do modelo

Avaliado nas tarefas de teste visadas, com métricas, frente aos padrões de referência da organização. Não é posição em benchmark público de geração de código, popularidade nem número de parâmetros.

Custo recorrente, potencial de ajuste fino e comunidade e suporte

Custo recorrente: taxas de licenciamento e despesas operacionais (na estimativa, tokens de entrada e saída e frequência da tarefa), dentro do orçamento da organização. Potencial de ajuste fino: se vale adaptar o modelo com dados do domínio (terminologia específica). Comunidade e suporte: documentação e ajuda para implementar e resolver problemas.

Fases de adoção (GenAI-5.1.4)

Descoberta: conscientização e desenvolvimento de capacidades, com treinamento das equipes, acesso aos LLMs/SLMs e experimentos com casos de uso iniciais para criar confiança. Iniciação e definição de uso: identificar e priorizar casos de uso práticos, avaliar a infraestrutura de teste com LLM, desenvolver conhecimento especializado e garantir alinhamento com as necessidades da organização. Utilização e iteração: integração total aos processos, monitoramento contínuo, medição e gestão da transformação para benefícios sustentáveis e escalabilidade.

Fases em paralelo e gestão de mudanças

As fases podem ocorrer em paralelo para casos de uso diferentes: a análise de relatórios de teste pode estar mais adiantada no roteiro enquanto a automação de teste está nas fases iniciais. Preocupações como o medo de perder o emprego devem ser reconhecidas e tratadas durante a adoção.

Habilidades essenciais (GenAI-5.2.1)

Avaliar os recursos do LLM, refinar prompts e avaliar o testware gerado por AI, com domínio das técnicas de prompting, compreensão das janelas de contexto e métodos de revisão. Essas habilidades se combinam com experiência em teste e no domínio para avaliar criticamente casos e dados gerados.

Conhecimentos essenciais de segurança e ambiente

Conhecer os riscos da GenAI e as estratégias de mitigação, as implicações de segurança de compartilhar testware com LLMs, higienizar os dados (remover ou mascarar informações sensíveis e pessoais antes de enviar) e aplicar engenharia de prompts que preserva a privacidade. Na dimensão ambiental: modelos do tamanho certo para cada tarefa e padrões de uso otimizados.

Criação de capacidade (GenAI-5.2.2)

Abordagem prática e gradual: exercícios orientados, aprendizado entre pares e integração aos poucos nas tarefas diárias; caminhos de aprendizado estruturados; prática com vários LLMs/SLMs. A equipe progride de prompts básicos para técnicas focadas, como prompts específicos de teste.

Padrão de prompt e comunidade de prática

Padrão de prompt é um modelo reutilizável para criar prompts com resultados consistentes. Comunidades internas de prática fazem reuniões regulares para mostrar aplicações bem-sucedidas, discutir desafios e refinar práticas, compartilhando bibliotecas de padrões de prompt e lições aprendidas.

Evolução do papel do testador (GenAI-5.2.3)

Torna-se especialista em teste assistido por AI, orientando e verificando o testware gerado. Novas tarefas: revisar a saída da AI, refinar prompts e manter bibliotecas de prompts específicas de teste. A experiência em técnicas de teste continua essencial.

Evolução do papel do gerente de teste

Estratégia de teste baseada em AI, gerenciamento de risco com AI e monitoramento e controle dos processos com AI; equilibrar recursos humanos e de AI; estabelecer governança de AI; supervisionar equipes híbridas de pessoas e ferramentas ou agentes de GenAI; garantir competências tradicionais e alfabetização em AI.

Macetes para memorizar

💡 Riscos da Shadow AI

SCP: Segurança e privacidade, Conformidade, Propriedade intelectual.

💡 Fases de adoção

Descobre, Define, Usa: descoberta → iniciação e definição de uso → utilização e iteração.

💡 Critérios de seleção

DACC: Desempenho, Ajuste fino, Custo recorrente, Comunidade e suporte.

💡 Estratégia de GenAI

Objetivo → LLM → Dados → Treino → Métrica → Diretriz, nesta ordem de raciocínio.

💡 Diretrizes de processo

Confidencial, Transparente, Porta de qualidade.

💡 Papéis

Testador orienta e verifica; gerente equilibra e governa.

💡 Capacitação

Do básico ao específico, em grupo e aos poucos.

Tabela de memorização: não confunda

Quando é...	...e quando é
Fase de descoberta Treinar as equipes, dar acesso aos LLMs, primeiros experimentos para criar confiança.	**Fase de iniciação e definição de uso** Identificar e priorizar casos de uso práticos e avaliar a infraestrutura.
Iniciação e definição de uso Escolher e priorizar onde usar a GenAI.	**Utilização e iteração** GenAI integrada aos processos, com monitoramento contínuo e gestão da transformação.
Custo recorrente Tokens de entrada e saída, frequência de uso e licenças.	**Comunidade e suporte** Documentação e ajuda para implementar e resolver problemas.
Desempenho do modelo Resultado nas tarefas de teste visadas, medido com métricas frente às referências da organização.	**Potencial de ajuste fino** Possibilidade de adaptar o modelo com dados e terminologia do domínio.
Portas de qualidade Impedir que testware gerado siga para uso sem revisão.	**Obrigações de transparência** Saber quais casos e relatórios foram gerados com GenAI, por exemplo para auditores.
Papel do testador Orientar e verificar: revisar saída, refinar prompts, manter bibliotecas de prompts.	**Papel do gerente de teste** Equilibrar e governar: estratégia, risco, controle, governança e equipes híbridas.

Risco de segurança e privacidade (Shadow AI) Ferramenta pessoal com dados de clientes: possível violação de dados.	**Risco de propriedade intelectual vaga** Licença pouco clara e documentos protegidos por direitos autorais: risco de disputas.

Números que caem

Item	Valor
Peso no exame	**7 questões e 7 pontos (17,5% das questões)**
Tempo de estudo do capítulo 5 no syllabus	**80 minutos**
Fases de adoção	**3**
Critérios de seleção de LLMs/SLMs	**4**
Riscos da Shadow AI	**3**

Alertas de prova (pegadinhas)

⚠ **Alerta de prova:** Para evitar Shadow AI, proíba ferramentas de AI ou deixe cada testador escolher a sua.
Como não cair: O caminho é definir estratégia e etapas de integração; proibir não é o proposto e escolha individual sem aprovação é a própria Shadow AI.

⚠ **Alerta de prova:** Usar só ferramentas gratuitas reduz riscos contratuais.
Como não cair: Ferramentas gratuitas não aprovadas podem ter segurança frágil e licenças pouco claras.

⚠ **Alerta de prova:** Fases como 'monitoramento', 'experimentação piloto', 'consolidação e expansão' ou 'prova de conceito'.
Como não cair: Só existem três: descoberta; iniciação e definição de uso; utilização e iteração. Monitoramento acontece dentro da utilização e iteração.

⚠ **Alerta de prova:** Casos de uso em fases diferentes indicam falha, pois as fases valem para a organização inteira.
Como não cair: As fases podem correr em paralelo por caso de uso, sem exigência de sincronizar.

⚠ **Alerta de prova:** Escolher o modelo pelo ranking em benchmarks públicos de código, pela popularidade, pela data de lançamento mais recente ou pelo número de parâmetros.
Como não cair: Os critérios são desempenho nas tarefas de teste visadas (frente aos padrões da organização; poucos benchmarks públicos são de teste), potencial de ajuste fino, custo recorrente e comunidade e suporte (documentação e ajuda, não fama).

⚠ **Alerta de prova:** Tamanho da janela de contexto é o critério para adaptar o modelo à terminologia do setor.
Como não cair: Adaptar com dados do domínio é o potencial de ajuste fino.

⚠ **Alerta de prova:** Volume maior de dados de entrada gera resultados melhores, ou o LLM corrige dados ruins antes de responder.
Como não cair: O que importa é qualidade e relevância, com proteção; dados ruins tendem a produzir resultados ruins em todo uso.

⚠ **Alerta de prova:** Um segundo LLM pode substituir os revisores humanos nas portas de qualidade.
Como não cair: Comparar com outro LLM ajuda, mas não substitui a revisão do testware gerado prevista nas diretrizes.

⚠ **Alerta de prova:** Com GenAI o testador passa a treinar modelos, programar redes neurais ou deixa de aplicar técnicas de teste.
Como não cair: O testador avalia recursos do LLM, refina prompts e revisa o testware; as técnicas de teste continuam essenciais.

⚠ **Alerta de prova:** O gerente reduz a equipe na proporção da automação, delega a estratégia aos agentes ou deixa a escolha de ferramentas livre.
Como não cair: O gerente equilibra recursos humanos e de AI, desenvolve a estratégia baseada em AI e estabelece governança.

⚠ **Alerta de prova:** Diante do medo de perder o emprego, evite o tema ou adie a adoção até aprovação unânime.
Como não cair: Reconheça e trate essas preocupações durante a adoção.

⚠ **Alerta de prova:** Capacitar com um curso teórico único e integração imediata, ou com experimentação livre sem caminho definido.
Como não cair: A capacitação é prática e gradual, com exercícios orientados, aprendizado entre pares e caminhos estruturados.

⚠ **Alerta de prova:** Antes de colar logs num LLM externo, divida-os entre vários LLMs ou converta-os em imagem para proteger os dados.
Como não cair: Espalhar aumenta a exposição e imagem não remove nada; o certo é higienizar, removendo ou mascarando dados sensíveis e pessoais.

Como cai na prova

📝 **Como cai:** Descrição de atividades da organização e pergunta 'em qual fase da adoção ela está?'.
Como resolver: Procure o verbo: treinar, dar acesso e experimentar = descoberta; identificar e priorizar casos de uso, avaliar infraestrutura = iniciação e definição de uso; integrado e monitorado continuamente = utilização e iteração. Elimine nomes de fase inexistentes.

📝 **Como cai:** Qual critério de seleção foi avaliado ou negligenciado.
Como resolver: Case o sintoma: sem documentação = comunidade e suporte; tokens e licenças = custo recorrente; adaptar ao domínio = potencial de ajuste fino; resultado nas tarefas = desempenho.

📝 **Como cai:** Qual risco de Shadow AI está em evidência.
Como resolver: Dados sensíveis em ferramenta pessoal = segurança e privacidade; empresa regulada = conformidade; licença e direitos autorais = propriedade intelectual. Descarte riscos de outros capítulos (alucinação, energia, excesso de ajuste).

📝 **Como cai:** O que muda no papel do testador ou do gerente.
Como resolver: Escolha a alternativa equilibrada: testador revisa, refina e mantém prompts; gerente equilibra, governa e supervisiona equipes híbridas. Elimine extremos: substituir pessoas, treinar modelos, abandonar técnicas, delegar decisões à AI.

📝 **Como cai:** 'Selecione DUAS' sobre capacitação ou critérios de seleção.
Como resolver: Valide cada alternativa contra a lista do syllabus; as erradas trazem popularidade, data de lançamento, idiomas da interface, curso único ou experimentação sem estrutura.

Gatilhos do enunciado

Se o enunciado trouxer...	Pense em
Contas pessoais de GenAI sem aprovação nem supervisão formal	**Shadow AI**
Licença confusa e documentos protegidos por direitos autorais	**Propriedade intelectual vaga**
Empresa regulada com ferramentas não aprovadas	**Não conformidade com padrões e regulamentos, consequências legais**
Por onde começar a estratégia de GenAI	**Objetivos mensuráveis, como aumento de produtividade**
Auditores querem saber o que foi gerado com GenAI	**Obrigações de transparência**
Evitar que testware gerado siga sem verificação	**Portas de qualidade com revisão**
Tokens de entrada e saída, frequência e licenças	**Custo recorrente**

Modelo pouco usado, sem documentação nem ajuda	**Comunidade e suporte**
Terminologia específica do setor, adaptar com dados do domínio	**Potencial de ajuste fino**
Precisa analisar capturas de tela com histórias	**Recursos funcionais do modelo, como entrada multimodal**
Treinar, dar acesso e experimentar; priorizar casos de uso; integrado e medido continuamente	**Descoberta; iniciação e definição de uso; utilização e iteração**
Antes de enviar logs a um LLM externo	**Higienizar: remover ou mascarar dados sensíveis**
LLM grande de raciocínio para reformatar listas	**Modelo do tamanho certo para a tarefa**
Reuniões regulares para trocar práticas e lições	**Comunidades internas de prática**

🎯 Se você lembrar de só uma coisa deste capítulo...

- Shadow AI se evita com estratégia e adoção estruturada, não com proibição.
- Fases: descoberta; iniciação e definição de uso; utilização e iteração, podendo correr em paralelo por caso de uso.
- Critérios: desempenho nas tarefas de teste, potencial de ajuste fino, custo recorrente, comunidade e suporte.
- Estratégia começa por objetivos mensuráveis e tem diretrizes com transparência e portas de qualidade.
- Testador orienta e verifica o testware gerado; gerente equilibra pessoas e AI com governança.

Essenciais para a prova (checklist)

☐ Sei definir Shadow AI e listar seus três riscos.

☐ Sei os aspectos da estratégia de GenAI e por onde ela começa.

☐ Sei os três elementos das diretrizes de processo.

☐ Sei os quatro critérios de seleção de LLMs/SLMs e reconhecê-los por sintoma.

☐ Sei as três fases de adoção, na ordem, e que podem ocorrer em paralelo.

☐ Sei as habilidades e conhecimentos essenciais, incluindo higienização de dados e modelo do tamanho certo.

☐ Sei como criar capacidade: prática gradual, caminhos estruturados, padrões de prompt e comunidades de prática.

☐ Sei as novas tarefas do testador e as novas responsabilidades do gerente de teste.

Questões de revisão do capítulo

R3.1 · Implementação e Integração da AI Generativa em Organizações de Teste

Na definição da estratégia de GenAI para o teste, por que o syllabus destaca a qualidade dos dados de entrada?

A. Porque dados de baixa qualidade são corrigidos pelo LLM antes de gerar a resposta
B. Porque a eficácia do teste com LLM depende de entradas precisas, relevantes e protegidas
C. Porque volumes maiores de dados de entrada geram resultados melhores nas tarefas
D. Porque a qualidade dos dados só importa durante o ajuste fino, e não no uso diário

R3.2 · Implementação e Integração da AI Generativa em Organizações de Teste

Ao definir diretrizes de processo para o uso de GenAI no teste, uma organização quer evitar que testware gerado por AI siga para uso sem verificação. Qual diretriz descrita no syllabus atende a esse objetivo?

A. Uso de temperatura alta para variar o testware
B. Troca de revisores humanos por um segundo LLM
C. Portas de qualidade com revisão do testware gerado
D. Liberação direta do testware gerado para execução

R3.3 · Implementação e Integração da AI Generativa em Organizações de Teste

Em uma organização de teste habilitada para AI, quais atribuições de gestão o syllabus associa ao gerente de teste?

A. Equilibrar recursos humanos e de AI e estabelecer governança de AI
B. Delegar à AI a escolha dos casos de uso e dos limites de seu uso
C. Reduzir a equipe humana na mesma proporção dos testes automatizados
D. Desenvolver os modelos de linguagem usados nas ferramentas de teste

R3.4 · Implementação e Integração da AI Generativa em Organizações de Teste

Uma organização de teste está montando seu roteiro de adoção de GenAI e quer seguir as fases descritas no syllabus. Quais são essas fases, na ordem?

A. Capacitação; automação de processos; otimização contínua
B. Descoberta; iniciação e definição de uso; utilização e iteração
C. Avaliação; prova de conceito; implantação em larga escala
D. Exploração; padronização de prompts; expansão para a empresa

Respostas comentadas

R3.1. B) Porque a eficácia do teste com LLM depende de entradas precisas, relevantes e protegidas

Por quê: A estratégia considera objetivos mensuráveis, seleção de LLMs, qualidade dos dados, treinamento, métricas e diretrizes de processo (GenAI-5.1.2).

✗ A) Incorreta. O LLM não corrige a entrada; dados ruins tendem a produzir resultados ruins.
✗ C) Incorreta. O que importa é a qualidade e a relevância, não o volume.
✗ D) Incorreta. A qualidade da entrada afeta o resultado de cada uso do LLM.

> **Dica de prova:** Entrada boa e protegida, resultado confiável.

R3.2. C) Portas de qualidade com revisão do testware gerado

Por quê: Para garantir conformidade regulatória e diretrizes éticas, a estratégia de GenAI estabelece diretrizes de processo para o uso da GenAI (GenAI-5.1.2).

✗ A) Incorreta. A temperatura não garante qualidade.
✗ B) Incorreta. Comparar com outro LLM pode ajudar, mas não substitui a revisão prevista nas portas de qualidade.
✗ D) Incorreta. Isso contraria a revisão prevista nas diretrizes.

> **Dica de prova:** Testware gerado passa por porta de qualidade.

R3.3. A) Equilibrar recursos humanos e de AI e estabelecer governança de AI

Por quê: Os gerentes de teste passam a cuidar de estratégia de teste baseada em AI, gerenciamento de risco com AI e monitoramento e controle dos processos com AI (GenAI-5.2.3).

✗ B) Incorreta. Definir casos de uso e governança é responsabilidade do gerente.
✗ C) Incorreta. O syllabus fala em equilibrar recursos, não em reduzir pessoas proporcionalmente.
✗ D) Incorreta. Desenvolver modelos não é a atribuição descrita para o gerente de teste.

> **Dica de prova:** Gerente: equilíbrio entre pessoas e AI, com governança.

R3.4. B) Descoberta; iniciação e definição de uso; utilização e iteração

Por quê: A descoberta foca conscientização e capacitação; a iniciação e definição de uso prioriza casos de uso; a utilização e iteração integra a GenAI aos processos de teste (GenAI-5.1.4).

✗ A) Incorreta. Essa sequência não corresponde às três fases descritas no syllabus.
✗ C) Incorreta. Prova de conceito e implantação em larga escala não são os nomes das fases do syllabus.
✗ D) Incorreta. Padronizar prompts é uma prática de capacitação, não uma fase do roteiro de adoção.

> **Dica de prova:** Descoberta, iniciação e definição de uso, utilização e iteração.

04

CAPÍTULO 4 · 12,5% DA PROVA

Infraestrutura de teste com tecnologia LLM para Teste de Software

É o menor capítulo: 5 questões e 5 pontos, todas de 1 ponto, mas cheias de pares fáceis de inverter. Preciso saber a arquitetura (front-end, back-end, fontes de dados e LLM) e o papel de cada parte, como o RAG prepara e usa os documentos, o que são agentes com LLM (autônomos, semiautônomos e multiagentes) e como reduzir seus riscos, o ajuste fino com seus quatro desafios e o LLMOps nas três abordagens de uso. A prova costuma trazer um cenário e perguntar qual componente, qual técnica ou qual desafio. Onde mais erro: inverter RAG e ajuste fino, achar que o front-end pós-processa a saída e confundir excesso de ajuste com opacidade.

Neste capítulo
16 tópicos-chave · 14 pegadinhas · 7 macetes · 14 gatilhos · 4 questões comentadas

Tópicos-chave

Arquitetura de infraestrutura de teste com LLM

Front-end, back-end, fontes de dados externas e um LLM integrado. Vai além do cliente-servidor tradicional porque o LLM é um componente que interpreta e raciocina sobre produtos de teste, gerando insights dinamicamente, diferente de chatbots baseados em regras.

Front-end

A interface onde o testador digita consultas ou comandos e vê os resultados. Não faz pós-processamento nem hospeda o LLM.

Back-end

Processa as entradas, autentica o usuário, recupera dados relevantes, prepara o prompt, interage com o LLM e pós-processa a saída para alinhá-la às condições de teste antes de mostrar ao testador.

Fontes de dados

O back-end integra banco de dados relacional para dados estruturados, como casos de teste, e banco de dados vetorial para busca semântica por embeddings.

LLM integrado

Pode ser serviço de terceiros acessado via API ou modelo interno personalizado. Gera respostas a partir dos prompts estruturados que o back-end prepara.

RAG (geração aumentada por recuperação)

Gera respostas fundamentadas em dados recuperados por similaridade semântica, e não só no treinamento prévio. No teste, mantém análise e projeto alinhados a especificações, requisitos e dados de teste mais recentes, sem retreinar o modelo.

Pré-processamento do RAG

Dividir os documentos em blocos (por exemplo, 256 a 512 tokens), limpá-los, gerar embeddings e guardá-los em banco vetorial. A divisão vem antes dos embeddings e garante recuperação focada e compatível com a janela de contexto.

Processamento do prompt no RAG

Duas etapas, nesta ordem: recuperação dos blocos pela semelhança semântica entre o embedding da pergunta e os embeddings dos blocos; depois geração da resposta pelo LLM com esses trechos como contexto. Não envia a documentação inteira e não ajusta o modelo.

Agente com tecnologia LLM

Processa tarefas de forma semiautônoma ou autônoma, usa o LLM para entender e gerar linguagem e pode agir invocando funções predefinidas, chamadas ferramentas. Integrados às ferramentas de teste, levam da execução baseada em scripts para a automação orientada por objetivos, cobrindo análise, modelagem, implementação, execução e relatório.

Agente autônomo x semiautônomo

O autônomo opera de forma independente, com o mínimo de intervenção humana, usando regras predefinidas, aprendizado por reforço e loops de feedback adaptativos. O semiautônomo executa sob supervisão humana periódica para garantir que as metas sejam atendidas.

Arquitetura multiagente

Agentes com funções especializadas (analisar requisitos, gerar casos, preparar dados) que se comunicam e se coordenam; esse esforço coordenado é a orquestração. Pode emular raciocínio e tomada de decisão humanos, mas não elimina erros.

Riscos e mitigação dos agentes

Agentes têm os mesmos problemas dos LLMs: alucinações, erros de raciocínio e vieses. Mitigação: procedimentos automatizados de verificação dos resultados ou agentes semiautônomos para tarefas críticas.

Ajuste fino

Aprendizado supervisionado com exemplos rotulados que atualiza os pesos e adapta um LLM ou SLM pré-treinado a tarefas ou domínios específicos, dando vocabulário e raciocínio especializados (ex.: pares de história e caso aprovado). Um SLM ajustado pode ter alto desempenho na tarefa sem a sobrecarga computacional de um LLM, mas ainda pode errar.

Desafios do ajuste fino

Usar dados de qualidade para evitar resultados tendenciosos ou imprecisos; reduzir o excesso de ajuste (ótimo no treino, ruim em dados novos); tratar a opacidade (difícil entender como o modelo decide, o que complica depurar e validar); gerenciar os recursos computacionais.

LLMOps

Práticas, ferramentas e processos para desenvolver, implantar, monitorar e manter LLMs em produção. As decisões variam conforme a abordagem de uso adotada.

Três abordagens de uso e o LLMOps

Chatbot de AI: gerenciar riscos de privacidade e segurança e otimizar o custo; usar LLM como serviço se as garantias necessárias forem dadas ou, para maior controle, infraestrutura interna com LLMs licenciados de código aberto. Ferramenta de teste com GenAI: considerações parecidas (privacidade, segurança, custos) e, além disso, avaliar as garantias de segurança de dados e de desempenho do fornecedor, com análise de custo-benefício e de riscos; ela normalmente complementa os processos existentes. Desenvolvimento interno: controle abrangente de privacidade e segurança, planejamento dos custos (recursos computacionais, armazenamento de dados, treinamento de pessoal), processos para validar e manter o que for desenvolvido e conhecimento especializado. Não são mutuamente exclusivas e podem incorporar RAG e ajuste fino.

Macetes para memorizar

💡 RAG x ajuste fino

RAG consulta a cola na hora; ajuste fino estuda antes da prova e muda a cabeça (os pesos).

💡 Pipeline do RAG

Preparo: Corta, Limpa, Vetoriza, Guarda. Uso: Recupera, depois Gera.

Arquitetura

Front fala, Back faz, Banco guarda, LLM pensa.

Bancos de dados

Relacional = tabela; Vetorial = sentido.

Desafios do ajuste fino

DEOR: Dados de qualidade, Excesso de ajuste, Opacidade, Recursos computacionais.

Agentes

Semi = supervisão periódica; Auto = intervenção mínima; Multi = orquestração. Todos podem errar.

Abordagens de LLMOps

Chatbot: privacidade e custo. Ferramenta: garantias do fornecedor. Interno: custos, especialistas e validação. E podem andar juntas.

Tabela de memorização: não confunda

Quando é...	...e quando é
RAG Dados que mudam com frequência; trechos recuperados entram como contexto da consulta, sem mexer nos pesos.	**Ajuste fino** Dar vocabulário, formato e raciocínio de domínio; treinamento adicional que atualiza os pesos.
Front-end Interface onde o testador digita e vê o resultado.	**Back-end** Autentica, recupera dados, monta o prompt, chama o LLM e pós-processa a saída.
Banco relacional Dados estruturados, como casos de teste.	**Banco vetorial** Embeddings para busca por similaridade semântica.
Agente autônomo Independente, com intervenção humana mínima.	**Agente semiautônomo** Supervisão humana periódica; indicado para tarefas críticas.
Excesso de ajuste Ótimo no projeto do treino, ruim em histórias de outros produtos: perdeu generalização.	**Opacidade** Ninguém consegue explicar como o modelo chegou às prioridades: difícil depurar e validar.
Agente com LLM Age invocando ferramentas e executa automação orientada por objetivos.	**Chatbot de AI** Foca pergunta e resposta, sem executar a automação.
Abordagem chatbot de AI Considerações centrais: privacidade, segurança e custo do serviço.	**Abordagem ferramenta comercial com GenAI** Além disso, avaliar as garantias de segurança e desempenho do fornecedor e o custo-benefício.

Números que caem

Item	Valor
Peso no exame	**5 questões e 5 pontos (12,5% das questões)**
Tempo de estudo do capítulo 4 no syllabus	**110 minutos**
Tamanho de bloco citado como exemplo no RAG	**256 a 512 tokens**
Etapas do processamento do prompt no RAG	**2 (recuperação e geração)**
Desafios do ajuste fino	**4**
Abordagens de uso de GenAI no LLMOps	**3 (chatbot de AI, ferramenta de teste com GenAI, desenvolvimento interno)**

Alertas de prova (pegadinhas)

⚠ **Alerta de prova:** O RAG atualiza os pesos com os documentos e o ajuste fino busca trechos a cada consulta.
Como não cair: Está invertido: RAG recupera trechos e os usa como contexto; ajuste fino atualiza os pesos.

⚠ **Alerta de prova:** Requisitos mudam toda semana: faça ajuste fino semanal ou aumente a temperatura para o modelo considerar as mudanças.
Como não cair: A resposta é RAG com as fontes de dados da empresa, sem retreinar. Temperatura só controla aleatoriedade.

⚠ **Alerta de prova:** O front-end faz o pós-processamento da saída, ou o back-end deixa de existir e o testador fala direto com o modelo.
Como não cair: Pós-processar, autenticar, recuperar dados e montar o prompt é trabalho do back-end.

⚠ **Alerta de prova:** Banco vetorial guarda os casos de teste estruturados e o relacional faz a busca semântica.
Como não cair: Papéis invertidos: relacional para estruturado, vetorial para semântico.

⚠ **Alerta de prova:** Gerar embeddings dos documentos inteiros, armazenar e só depois dividir em blocos.
Como não cair: A ordem é dividir, limpar, gerar embeddings e armazenar.

⚠ **Alerta de prova:** Dividir em blocos de 256 a 512 tokens serve para reduzir parâmetros, retreinar o modelo ou dispensar o banco vetorial.
Como não cair: Os blocos existem para recuperação focada e compatível com a janela de contexto; os embeddings vão para o banco vetorial.

⚠ **Alerta de prova:** O RAG escolhe os trechos pelo tamanho, pela data ou pela ordem alfabética.
Como não cair: O critério é a semelhança semântica entre o embedding da pergunta e os embeddings dos blocos.

⚠ **Alerta de prova:** Agentes dispensam verificação porque são mais precisos, ou mais autonomia faz o agente corrigir os próprios erros.
Como não cair: Agentes também alucinam e erram; use verificação automatizada dos resultados ou agente semiautônomo em tarefas críticas.

⚠ **Alerta de prova:** Em sistema crítico, use arquitetura multiagente sem supervisão para que um agente corrija o outro, ou agente autônomo com temperatura alta.
Como não cair: Orquestração não elimina erros e temperatura alta aumenta o risco; tarefa crítica pede semiautônomo com supervisão periódica.

⚠ **Alerta de prova:** Supervisão humana periódica descreve o agente autônomo.
Como não cair: Supervisão periódica é o semiautônomo; o autônomo opera com intervenção humana mínima.

⚠ **Alerta de prova:** Ajuste fino só se aplica a LLMs grandes, ou um SLM ajustado não alucina.
Como não cair: O ajuste fino também vale para SLMs, e modelos ajustados continuam podendo gerar resultados imprecisos.

⚠ **Alerta de prova:** O modelo ajustado vai mal em produtos novos por causa da janela de contexto ou do custo computacional.
Como não cair: Ótimo no treino e ruim no novo é excesso de ajuste. Opacidade é não entender as decisões; recursos computacionais são outro desafio.

⚠ **Alerta de prova:** A organização deve escolher uma única abordagem para simplificar o LLMOps, ou a ferramenta comercial deve substituir os processos.
Como não cair: As abordagens podem coexistir e incorporar RAG e ajuste fino; ferramentas comerciais normalmente complementam os processos existentes.

⚠ **Alerta de prova:** Se os dados não podem sair da empresa, basta mascarar e enviar ao provedor, ou liberar contas pessoais de chatbot.
Como não cair: Mascarar ainda manda os dados para fora e contas pessoais são Shadow AI; a opção é infraestrutura interna com LLMs licenciados de código aberto.

Como cai na prova

📝 **Como cai:** Cenário de dados ou conhecimento: RAG ou ajuste fino?
Como resolver: Se o problema é informação atual e mutável, sem retreinar: RAG. Se é vocabulário, formato ou raciocínio de domínio aprendido pelo modelo: ajuste fino. Descarte temperatura e zero disparo.

📝 **Como cai:** Qual componente da arquitetura faz determinada função.
Como resolver: Interface = front-end; tudo que é processamento entre o usuário e o LLM (autenticar, buscar, montar prompt, pós-processar) = back-end; armazenar = bancos; gerar = LLM.

📝 **Como cai:** Sintoma depois do ajuste fino: qual desafio é esse?
Como resolver: Perdeu generalização = excesso de ajuste; decisão inexplicável = opacidade; base ruim ou concentrada = dados de qualidade; custo alto = recursos computacionais.

📝 **Como cai:** Tipo de agente ou medida para reduzir o risco dos agentes.
Como resolver: Leia o grau de supervisão: mínima = autônomo; periódica = semiautônomo; vários coordenados = multiagente. Para risco, a resposta é verificação automatizada ou semiautônomo, nunca mais autonomia ou menos ferramentas.

📝 **Como cai:** Abordagem de uso e a consideração de LLMOps correspondente.
Como resolver: Identifique a abordagem (chatbot, ferramenta comercial, desenvolvimento interno) e escolha a consideração dela; lembre que as abordagens podem coexistir.

Gatilhos do enunciado

Se o enunciado trouxer...	Pense em
Onde o testador digita consultas e comandos	**Front-end**
Autentica, recupera dados, prepara o prompt, pós-processa a saída	**Back-end**
Vai além do cliente-servidor tradicional	**O LLM interpreta e raciocina sobre produtos de teste**

LLM via API de terceiros ou modelo próprio	**LLM integrado à arquitetura: serviço de terceiros via API ou modelo interno personalizado**
Dados que mudam toda semana, sem retreinar	**RAG**
Dividir, limpar, vetorizar, armazenar; recuperar e depois gerar	**Pré-processamento e processamento do RAG**
Treino adicional com exemplos rotulados; vocabulário de domínio	**Ajuste fino**
Ótimo no projeto do treino, ruim nos outros	**Excesso de ajuste**
Não dá para entender como o modelo decidiu	**Opacidade**
Base de treino mal escrita ou concentrada em um tipo	**Dados de qualidade para evitar resultado tendencioso ou impreciso**
Invoca funções predefinidas; automação orientada por objetivos	**Agente com tecnologia LLM (ferramentas)**
Vários agentes especializados que se coordenam	**Orquestração em arquitetura multiagente**
Sistema crítico com agentes	**Agente semiautônomo ou verificação automatizada dos resultados**
Implantar, monitorar e manter LLM em produção	**LLMOps**

🎯 Se você lembrar de só uma coisa deste capítulo...

- RAG recupera trechos por similaridade e os usa como contexto; ajuste fino muda os pesos.
- Back-end autentica, recupera, monta o prompt, chama o LLM e pós-processa; front-end é só a interface.
- Agentes erram como os LLMs: verificação automatizada ou semiautônomo em tarefas críticas.
- Desafios do ajuste fino: dados de qualidade, excesso de ajuste, opacidade e recursos computacionais.
- As três abordagens de uso não são mutuamente exclusivas e podem incorporar RAG e ajuste fino.

Essenciais para a prova (checklist)

- ☐ Sei o papel de front-end, back-end, bancos relacional e vetorial e LLM.
- ☐ Sei as duas formas de disponibilizar o LLM na arquitetura.
- ☐ Sei a ordem do pré-processamento e das duas etapas do RAG.

- ☐ Sei diferenciar RAG de ajuste fino e escolher entre eles num cenário.
- ☐ Sei diferenciar agentes autônomos, semiautônomos e multiagentes.
- ☐ Sei como mitigar o risco de resultados incorretos de agentes.
- ☐ Sei reconhecer os quatro desafios do ajuste fino por sintoma.
- ☐ Sei a consideração de LLMOps central em cada uma das três abordagens.

Questões de revisão do capítulo

R4.1 · Infraestrutura de teste com tecnologia LLM para Teste de Software

Em uma infraestrutura de teste com tecnologia LLM, o back-end integra várias fontes de dados. Qual combinação está de acordo com o syllabus?

A. Banco relacional para dados estruturados, como casos de teste, e banco vetorial para busca semântica
B. Banco relacional para os pesos do LLM e banco vetorial para as credenciais de autenticação dos usuários
C. Banco vetorial para os logs do front-end e banco relacional para os prompts do sistema do chatbot
D. Banco vetorial para casos de teste estruturados e banco relacional para busca por similaridade semântica

R4.2 · Infraestrutura de teste com tecnologia LLM para Teste de Software

Por que o syllabus afirma que a arquitetura de uma infraestrutura de teste com tecnologia LLM vai além de um modelo cliente-servidor tradicional?

A. Porque o front-end passa a armazenar os casos de teste em um banco de dados relacional
B. Porque o LLM é um componente que interpreta e raciocina sobre produtos de teste
C. Porque o LLM segue respostas roteirizadas, como um chatbot baseado em regras
D. Porque o back-end deixa de existir e o testador fala direto com o modelo

R4.3 · Infraestrutura de teste com tecnologia LLM para Teste de Software

Os requisitos de um produto mudam toda semana, e a equipe quer que o LLM gere condições de teste alinhadas à versão mais recente, sem retreinar o modelo. Qual abordagem do syllabus atende a esse objetivo?

A. Aumento da temperatura para o modelo considerar mudanças recentes
B. Geração aumentada por recuperação com as fontes de dados da empresa
C. Ajuste fino semanal do modelo com a nova versão dos requisitos
D. Prompting de zero disparo confiando no treinamento prévio do modelo

R4.4 · Infraestrutura de teste com tecnologia LLM para Teste de Software

Na arquitetura de uma infraestrutura de teste com tecnologia LLM, de que formas o LLM pode ser disponibilizado, segundo o syllabus?

A. Como modelo interno exclusivo, pois serviços externos não se integram ao back-end
B. Como serviço de terceiros acessado via API ou como modelo interno personalizado
C. Como parte do front-end, executado no navegador de cada testador da equipe
D. Como chatbot isolado, pois o LLM não pode ser chamado por outros componentes

Respostas comentadas

R4.1. A) Banco relacional para dados estruturados, como casos de teste, e banco vetorial para busca semântica

Por quê: O back-end integra bancos relacionais e vetoriais, recupera dados, prepara o prompt e pós-processa a saída do LLM (GenAI-4.1.1).

✗ B) Incorreta. Pesos do modelo e credenciais não são as fontes de dados descritas para o back-end.
✗ C) Incorreta. O syllabus não atribui esses papéis às fontes de dados.
✗ D) Incorreta. Os papéis estão invertidos.

> **Dica de prova:** Relacional: estruturado; vetorial: semântico.

R4.2. B) Porque o LLM é um componente que interpreta e raciocina sobre produtos de teste

Por quê: A arquitetura incorpora componentes de processamento inteligente, como LLMs, e back-ends que integram várias fontes de dados (GenAI-4.1.1).

✗ A) Incorreta. Os dados estruturados ficam nas fontes de dados integradas ao back-end.
✗ C) Incorreta. Diferentemente dos chatbots baseados em regras, a infraestrutura com LLM gera insights dinamicamente.
✗ D) Incorreta. O back-end continua responsável por autenticação, recuperação de dados, prompt e pós-processamento.

> **Dica de prova:** Além do cliente-servidor: o LLM interpreta e raciocina.

R4.3. B) Geração aumentada por recuperação com as fontes de dados da empresa

Por quê: No teste de software, o RAG mantém tarefas como análise e projeto de teste alinhadas às especificações, aos requisitos e aos dados de teste mais recentes (GenAI-4.1.2).

✗ A) Incorreta. A temperatura controla a aleatoriedade, não a atualização de conhecimento.
✗ C) Incorreta. O ajuste fino é treinamento adicional, justamente o que se quer evitar.
✗ D) Incorreta. O treinamento prévio não contém os requisitos recentes.

> **Dica de prova:** Dados que mudam toda hora: RAG, sem retreinar.

R4.4. B) Como serviço de terceiros acessado via API ou como modelo interno personalizado

Por quê: A arquitetura típica tem front-end, back-end, fontes de dados externas e um LLM integrado, hospedado como serviço de terceiros ou modelo interno (GenAI-4.1.1).

✗ A) Incorreta. Serviços de terceiros via API são uma das opções descritas.

✗ C) Incorreta. O front-end é a interface do usuário, não o local do LLM.

✗ D) Incorreta. O back-end interage com o LLM, que gera respostas a partir de prompts estruturados.

> **Dica de prova:** LLM: serviço via API ou modelo interno.

05

CAPÍTULO 5 · 17,5% DA PROVA

Introdução à AI Generativa para Teste de Software

Aqui eu preciso dominar a base: o espectro de AI (simbólica, machine learning clássico, deep learning e AI generativa), como um LLM funciona por dentro (tokenização, embeddings, transformador, janela de contexto, não determinismo), os tipos de LLM (básico, ajustado por instrução e de raciocínio), os multimodais e modelos de linguagem visual, os recursos do LLM em tarefas de teste e a diferença entre chatbot e aplicação integrada. São 7 questões, todas K1 ou K2 (1 ponto cada), e o LO de conceitos de LLM (GenAI-1.1.2) costuma render duas. A prova gosta de associação do tipo 1-4 com A-D e de cenários curtos em que eu reconheço o conceito pelo sintoma. Onde mais erro: trocar tokenização por embeddings, achar que janela maior dá mais conhecimento e confundir o ajustado por instrução com o de raciocínio.

Neste capítulo

18 tópicos-chave · 14 pegadinhas · 7 macetes · 14 gatilhos · 4 questões comentadas

Tópicos-chave

AI simbólica

Usa sistema baseado em regras e símbolos definidos explicitamente por pessoas para imitar a tomada de decisão humana. Exemplo típico: classificar defeitos com regras do tipo 'se o módulo for Pagamentos e o texto citar timeout, a gravidade é alta'. Não aprende com dados nem gera conteúdo novo.

Machine learning clássico

Abordagem orientada a dados que exige preparação de dados, seleção de recursos (atributos) e treinamento de modelos. Usado em tarefas como categorização de defeitos e previsão de módulos com problemas. A pista é: alguém escolheu os atributos relevantes.

Deep learning

Redes neurais que encontram padrões em dados grandes e complexos, como imagens, aprendendo os recursos automaticamente, sem que os usuários os definam. Ainda pode exigir envolvimento humano em anotação, ajuste ou validação. Um modelo que só classifica telas existentes é deep learning, não AI generativa.

AI generativa

Usa técnicas de deep learning para criar conteúdo novo (texto, imagens, código) imitando padrões dos dados de treinamento. A vantagem destacada sobre o machine learning clássico é aplicar modelos pré-treinados direto às tarefas de teste, sem fase extra de treinamento. Não garante respostas iguais nem elimina erros.

LLM (modelo de linguagem grande)

Modelo baseado no transformador pré-treinado, treinado em conjuntos de dados muito grandes, capaz de lidar com nuances do idioma e gerar conteúdo coerente. Não é banco vetorial, não é sistema de regras e não é o modelo compacto (esse é o SLM).

Tokenização

Divide o texto em unidades menores chamadas tokens, que vão de um caractere até uma subpalavra ou palavra. Uma palavra longa como 'parametrização' ser quebrada em várias partes é comportamento normal. Tokenização cria os blocos de construção; quem dá significado numérico são os embeddings.

Embeddings

Vetores numéricos que representam os tokens; tokens de significado parecido ficam próximos, por isso o LLM percebe que 'falha' e 'erro' são parecidos. Tokenização e embeddings juntos convertem a linguagem em forma numérica.

Transformador e inferência

A arquitetura transformador processa o contexto de sequências extensas e aprende como os tokens se relacionam. Na inferência, o LLM prevê o próximo token com base nas relações aprendidas e no prompt. O texto gerado é estatisticamente plausível, e plausível não é necessariamente correto.

Janela de contexto

Quantidade de texto, medida em tokens, que o modelo pode considerar. Se a entrada passa da janela, informações relevantes ficam de fora. Janela maior ajuda a manter coerência em textos longos, mas aumenta a complexidade computacional e o tempo de processamento; ela não amplia o conhecimento do modelo nem se ajusta sozinha.

Não determinismo

A inferência é probabilística e depende de hiperparâmetros, então o mesmo prompt pode gerar saídas diferentes. Não é retreinamento entre envios nem tokenização diferente: a tokenização de um mesmo texto é a mesma.

SLM (modelo de linguagem pequeno)

Modelo compacto, com menos parâmetros que um LLM, que oferece soluções de GenAI leves e focadas. É a resposta quando o cenário pede rodar com poucos recursos uma tarefa específica, como classificar mensagens de log.

LLM básico (de fundação)

De uso geral, pré-treinado em dados amplos; é flexível, mas costuma exigir adaptação para tarefas específicas. É o ponto de partida dos estágios de treinamento que originam os ajustados por instrução e os de raciocínio.

LLM ajustado por instrução

Obtido a partir do básico, refinado com pares de solicitação e resposta esperada para seguir melhor instruções humanas e manter respostas coerentes. Às vezes chamado de 'sem raciocínio'. É o indicado para seguir formato, como gerar cenários Gherkin ou casos em template definido.

LLM de raciocínio

Amplia o ajustado por instrução, treinado em tarefas com compreensão contextual, etapas intermediárias de raciocínio e síntese de informações complexas. Indicado para priorizar testes com riscos e dependências, planejar a ordem de execução e sintetizar relatórios de defeito para achar tendências.

LLM multimodal

Estende o transformador para texto, imagem, som e vídeo e aprende relações entre esses tipos; a tokenização é adaptada a cada tipo de dado e a imagem vira embeddings por meio de modelos de linguagem visual. Analisa capturas de tela e wireframes junto com histórias de usuário ou relatórios de defeito, gerando casos mais ricos e mais cobertura, mas a saída continua precisando de revisão.

Modelo de linguagem visual

Subconjunto dos LLMs multimodais focado em integrar dados visuais e textuais. Usado para checar a consistência entre imagem e texto e identificar discrepâncias entre o resultado esperado e os elementos visuais de uma tela.

Recursos do LLM para tarefas de teste

Análise e aprimoramento de requisitos (ambiguidades, inconsistências, informação ausente e perguntas aos stakeholders), apoio à criação de casos de teste, geração de oráculos (resultados esperados), geração de dados de teste (valores limite e combinações), suporte à automação (gerar scripts e melhorar os existentes), análise de resultados (resumir e classificar anomalias por gravidade e prioridade) e criação de testware (redigir e atualizar planos e relatórios). O LLM apoia; aprovar, decidir e executar ficam fora dele.

Chatbot de AI x aplicação de teste com LLM

Chatbot favorece interações ad hoc: feedback rápido, esclarecer conceitos, explorar requisitos, teste exploratório com refinamento a cada resposta e integração de novos testadores. A aplicação integra o LLM via API às estruturas de teste para tarefas bem definidas e automatizadas, como gerar casos em volume. Nas duas formas é preciso engenharia de prompts sólida.

Macetes para memorizar

💡 Espectro de AI

Regra, Recurso, Rede, Gera: simbólica usa regra; ML clássico pede recurso escolhido por gente; deep learning é rede que acha o recurso sozinha; generativa gera conteúdo novo em cima do deep learning.

💡 Como o LLM responde

Corta, Vetoriza, Relaciona, Prevê: tokeniza, vira embedding, o transformador relaciona os tokens e prevê o próximo token.

💡 Tipos de LLM

Escada de três degraus: Básico (geral, precisa adaptar) → Instrução (segue formato) → Raciocínio (lógica em etapas). Cada degrau amplia o anterior.

💡 Multimodal x linguagem visual

Boneca russa: o modelo de linguagem visual mora dentro do multimodal.

💡 Janela de contexto

Janela maior: enxerga mais, paga mais (processamento e tempo), mas não sabe mais.

💡 Saída do LLM

Plausível não é correto: confira valores, oráculos e resultados esperados.

💡 Chatbot x aplicação

Chatbot conversa; aplicação integra e automatiza.

Tabela de memorização: não confunda

Quando é...	...e quando é

Tokenização Dividir o texto em unidades (caractere, subpalavra ou palavra); explica palavra quebrada em partes e contagem de tokens.	**Embeddings** Representar tokens como vetores numéricos; explica por que palavras de sentido parecido ficam próximas.
Machine learning clássico Pessoas preparam os dados e escolhem os atributos antes de treinar (previsão de módulos com defeito a partir de atributos escolhidos).	**Deep learning** Redes neurais aprendem sozinhas os recursos de dados grandes e complexos, como milhares de capturas de tela.
LLM ajustado por instrução Seguir instruções e formatos à risca: Gherkin, templates da organização, saída coerente.	**LLM de raciocínio** Inferência lógica em várias etapas: priorizar com riscos e dependências, planejar a ordem de execução, sintetizar tendências.
LLM multimodal Conceito amplo: texto, imagem, som e vídeo e a relação entre eles.	**Modelo de linguagem visual** Subconjunto do multimodal focado em imagem e texto; comparar a imagem com a descrição ou com o relatório de defeito.
Chatbot de AI Dúvida pontual, conceito, exploração dinâmica, teste exploratório, integração de novatos.	**Aplicação de teste com LLM via API** Volume, automação e integração à ferramenta de teste, como gerar casos para centenas de histórias por mês.
LLM Tarefas amplas, com mais parâmetros e mais recursos computacionais.	**SLM** Solução leve e focada, com poucos recursos e menos parâmetros.
Deep learning de classificação Distinguir ou classificar dados existentes (tela correta x tela quebrada).	**AI generativa** Criar conteúdo novo, como scripts e casos de teste em texto ou código.

Números que caem

Item	Valor
Peso no exame	**7 questões e 7 pontos (17,5% das questões)**
LO com 2 questões por prova	**GenAI-1.1.2 (conceitos de LLM)**
Tempo de estudo do capítulo 1 no syllabus	**100 minutos**
Tipos de AI no espectro	**4 (simbólica, machine learning clássico, deep learning, AI generativa)**

Alertas de prova (pegadinhas)

⚠ **Alerta de prova:** Trocar para um LLM com janela de contexto maior reduz alucinações porque o modelo passa a ter mais conhecimento.
Como não cair: A janela define quanto texto o modelo considera, não o que ele sabe. O efeito descrito é mais coerência em textos longos com mais custo computacional e tempo de processamento.

⚠ **Alerta de prova:** Se o log passa da janela, a janela se amplia automaticamente ou a tokenização compacta o excedente.
Como não cair: O tamanho da janela é característica do modelo. O que excede não é considerado e detalhes relevantes podem ficar de fora.

⚠ **Alerta de prova:** O mesmo prompt deu respostas diferentes porque o modelo foi retreinado com a primeira conversa ou porque a tokenização mudou.
Como não cair: O uso não altera os pesos e a tokenização de um mesmo texto é a mesma. A variação vem da inferência probabilística, que depende de hiperparâmetros.

⚠ **Alerta de prova:** A temperatura aproxima palavras de sentido parecido, ou a tokenização agrupa sinônimos em um único token.
Como não cair: Proximidade de sentido é coisa de embeddings. Temperatura controla aleatoriedade; tokenização só divide o texto.

⚠ **Alerta de prova:** Tokenização é converter tokens em vetores de alta dimensão para capturar o significado.
Como não cair: Isso descreve embeddings. Tokenização cria os blocos de construção (tokens) usados para entender e gerar texto.

⚠ **Alerta de prova:** Modelo que aprendeu sozinho a distinguir telas quebradas é machine learning clássico ou AI generativa.
Como não cair: Ninguém definiu os atributos e o modelo aprende de imagens: deep learning. Ele classifica telas existentes, não cria conteúdo, então não é generativa.

⚠ **Alerta de prova:** Modelo treinado com atributos escolhidos pela equipe para prever módulos com defeito é deep learning.
Como não cair: Preparação de dados com seleção de recursos por pessoas caracteriza o machine learning clássico.

⚠ **Alerta de prova:** LLM básico é o modelo compacto com poucos parâmetros para tarefas leves.
Como não cair: Compacto e leve é o SLM. O básico é de uso geral, pré-treinado em dados amplos, e costuma exigir adaptação.

⚠ **Alerta de prova:** Para formatar casos em Gherkin use o LLM de raciocínio e, para planejar a ordem de execução, o ajustado por instrução.
Como não cair: É o contrário: seguir formato pede o ajustado por instrução; planejar com dependências e riscos pede o de raciocínio.

⚠ **Alerta de prova:** Os modelos de raciocínio antecedem os básicos, ou os ajustados por instrução vêm de modelos de raciocínio simplificados.
Como não cair: A ordem é básico, depois ajustado por instrução, depois raciocínio, que amplia o ajustado por instrução.

⚠ **Alerta de prova:** Com imagem no prompt a saída fica mais determinística e dispensa revisão, ou consome menos energia que texto.
Como não cair: A saída do multimodal continua não determinística e precisa de verificação; processar imagem tende a exigir mais recursos que texto.

⚠ **Alerta de prova:** LLMs multimodais só processam texto e recebem a imagem já descrita por pessoas, ou dispensam tokenização.
Como não cair: O modelo processa a própria imagem, convertida em embeddings via modelo de linguagem visual, com tokenização adaptada a cada tipo de dado.

⚠ **Alerta de prova:** O LLM pode aprovar histórias pelo Product Owner, substituir a reunião de refinamento ou garantir produto sem defeitos.
Como não cair: O recurso é apontar ambiguidades e lacunas e sugerir perguntas aos stakeholders; quem decide são as pessoas e a saída precisa ser verificada.

⚠ **Alerta de prova:** Chatbots dispensam engenharia de prompts; só as aplicações integradas dependem dela.
Como não cair: Nas duas formas de interação, uma engenharia de prompts sólida, com instruções claras e específicas, é a base do sucesso.

Como cai na prova

📝 **Como cai:** Associação de tipos de AI (simbólica, ML clássico, deep learning, generativa) com descrições (A-D).
Como resolver: Ancore cada tipo na palavra-chave: regras, seleção de recursos, aprende recursos com redes neurais, cria dados novos usando deep learning. Resolva o par mais óbvio primeiro e elimine as opções incompatíveis; o distrator costuma inverter dois vizinhos.

📝 **Como cai:** Lista de afirmações (i a v) sobre LLMs básicos, ajustados por instrução e de raciocínio; escolha a combinação correta.
Como resolver: Marque V ou F em cada afirmação isoladamente: instrução combina com template e Gherkin; raciocínio combina com síntese, tendências e priorização; básico precisa de adaptação. Só depois procure a combinação.

📝 **Como cai:** Cenário com sintoma técnico (log maior que a janela, mesmo prompt com saídas diferentes, palavra quebrada em tokens).
Como resolver: Nomeie o mecanismo do syllabus (janela, inferência probabilística, tokenização). Descarte explicações inventadas: retreino a cada uso, janela que cresce, tokenização que compacta ou muda.

📝 **Como cai:** Qual recurso do LLM corresponde à tarefa descrita.
Como resolver: Mapeie o verbo: apontar ambiguidades = análise de requisitos; resultado esperado = oráculo; valores limite = dados de teste; gerar ou melhorar scripts = suporte à automação; resumir e classificar anomalias = análise de resultados; planos e relatórios = criação de testware. Elimine alternativas em que o LLM aprova, executa, corrige código ou garante.

📝 **Como cai:** Qual forma de interação ou qual tipo de modelo escolher para um cenário.
Como resolver: Volume, automação e integração apontam para aplicação via API; pergunta pontual e exploração apontam para chatbot. Imagem no cenário pede multimodal; poucos recursos pedem SLM; lógica em etapas pede raciocínio.

Gatilhos do enunciado

Se o enunciado trouxer...	Pense em
Regras escritas por especialistas, 'se... então' para classificar	**AI simbólica**
Preparar dados e escolher atributos para prever problemas	**Machine learning clássico**
Aprende sozinho as características a partir de milhares de imagens	**Deep learning**
Cria scripts ou textos novos imitando padrões do treinamento	**AI generativa (construída sobre deep learning)**
Palavras de sentido parecido ficam próximas	**Embeddings (vetores numéricos)**
Palavra longa dividida em várias partes pelo tokenizador	**Normal: token vai de um caractere a uma subpalavra ou palavra**
Material maior que a janela de contexto	**O modelo não considera tudo; detalhes relevantes ficam de fora**

Mesmo prompt enviado duas vezes, respostas diferentes	**Não determinismo: inferência probabilística**
Redação convincente e formato perfeito, mas valor errado	**Texto estatisticamente plausível não é necessariamente correto**
Leve, focado, poucos recursos	**SLM**
'Sem raciocínio', seguir instruções detalhadas de formato	**LLM ajustado por instrução**
Várias etapas de inferência lógica, dependências, riscos, restrições	**LLM de raciocínio**
Captura de tela ou wireframe junto com história ou relatório de defeito	**LLM multimodal / modelo de linguagem visual para achar discrepâncias**
Dúvida rápida, novato, teste exploratório; ou volume automático dentro da ferramenta	**Chatbot de AI; ou aplicação com LLM integrada via API**

🎯 Se você lembrar de só uma coisa deste capítulo...

- AI generativa usa deep learning, e sua vantagem sobre o ML clássico é aplicar modelos pré-treinados direto às tarefas, sem fase extra de treinamento.
- Janela de contexto é quanto texto, em tokens, o modelo considera: maior dá mais coerência e custa mais processamento.
- Ajustado por instrução segue formato; de raciocínio resolve lógica em várias etapas; básico precisa de adaptação.
- O LLM prevê o próximo token: o resultado é plausível, não necessariamente correto, e varia entre execuções.
- O LLM aponta, sugere e rascunha; aprovar, decidir e executar no ambiente ficam com pessoas e ferramentas.

Essenciais para a prova (checklist)

☐ Sei diferenciar AI simbólica, machine learning clássico, deep learning e AI generativa por uma palavra-chave cada.

☐ Sei explicar a sequência tokenização, embeddings, transformador e previsão do próximo token.

☐ Sei o que acontece quando a entrada excede a janela de contexto e o custo de uma janela maior.

☐ Sei explicar por que o mesmo prompt gera saídas diferentes.

☐ Sei a ordem e o uso de LLM básico, ajustado por instrução e de raciocínio.

☐ Sei que o modelo de linguagem visual é subconjunto do multimodal e como a imagem vira embedding.

☐ Sei listar os recursos do LLM para teste e reconhecer cada um por um exemplo.

☐ Sei quando usar chatbot de AI e quando usar aplicação com LLM integrada via API.

☐ Sei quando um SLM é a melhor escolha.

Questões de revisão do capítulo

R5.1 · Introdução à AI Generativa para Teste de Software

Uma equipe quer apoio de AI generativa para priorizar casos de teste considerando riscos, dependências entre testes e restrições de ambiente, uma tarefa com várias etapas de inferência lógica. Que tipo de LLM é o mais indicado?

A. LLM de raciocínio
B. LLM básico
C. LLM multimodal
D. LLM ajustado por instrução

R5.2 · Introdução à AI Generativa para Teste de Software

Uma equipe treinou um modelo com o histórico de relatórios de defeito, depois de preparar os dados e escolher os atributos relevantes, para prever quais módulos tendem a apresentar problemas. Que tipo de AI é esse, segundo o espectro do syllabus?

A. AI simbólica baseada em regras
B. Deep learning com redes neurais
C. Machine learning clássico
D. AI generativa com LLM

R5.3 · Introdução à AI Generativa para Teste de Software

Um engenheiro de automação explica a um colega como um LLM percebe que 'falha' e 'erro' têm sentidos próximos. Qual conceito descreve essa capacidade?

A. Temperatura, que aproxima as palavras de sentido parecido durante a inferência
B. Embeddings, vetores numéricos em que tokens de significado parecido ficam próximos
C. Janela de contexto, que guarda uma lista de sinônimos consultada pelo modelo
D. Tokenização, que agrupa palavras de mesmo sentido em um único token compartilhado

R5.4 · Introdução à AI Generativa para Teste de Software

Uma equipe discute a diferença entre LLMs multimodais e modelos de linguagem visual. Qual afirmação está correta?

A. Os dois termos, multimodal e linguagem visual, designam exatamente o mesmo tipo de modelo
B. Modelos de linguagem visual são um subconjunto dos LLMs multimodais focado em imagem e texto
C. Modelos de linguagem visual processam só texto, e os multimodais processam só imagens
D. LLMs multimodais são um subconjunto dos modelos de linguagem visual focado em áudio

Respostas comentadas

R5.1. A) LLM de raciocínio

Por quê: A escolha entre LLMs ajustados por instrução e de raciocínio depende da complexidade e das demandas de raciocínio da tarefa; priorização com múltiplos critérios favorece o de raciocínio (GenAI-1.1.3).

✗ B) Incorreta. LLMs básicos são de uso geral e geralmente exigem adaptações adicionais para tarefas específicas.

✗ C) Incorreta. Ser multimodal diz respeito aos tipos de dados processados, não à capacidade de raciocínio em várias etapas.

✗ D) Incorreta. LLMs ajustados por instrução seguem bem instruções, mas a tarefa descrita exige raciocínio em várias etapas.

> **Dica de prova:** Muitas etapas lógicas: LLM de raciocínio.

R5.2. C) Machine learning clássico

Por quê: No espectro de AI, o machine learning clássico requer preparação de dados, seleção de recursos e treinamento de modelos e é usado em tarefas como categorização de defeitos e previsão de problemas (GenAI-1.1.1).

✗ A) Incorreta. A AI simbólica usa regras e símbolos definidos por pessoas, não treinamento com dados.

✗ B) Incorreta. O deep learning aprende os recursos automaticamente, sem que os usuários os definam.

✗ D) Incorreta. A AI generativa cria conteúdo novo; o cenário descreve uma previsão a partir de atributos escolhidos.

> **Dica de prova:** Atributos escolhidos e modelo treinado: machine learning clássico.

R5.3. B) Embeddings, vetores numéricos em que tokens de significado parecido ficam próximos

Por quê: Tokenização e embeddings convertem a linguagem em forma numérica; a proximidade entre embeddings permite ao LLM entender relações entre palavras (GenAI-1.1.2).

✗ A) Incorreta. A temperatura controla a aleatoriedade da saída, não a relação de significado entre palavras.

✗ C) Incorreta. A janela de contexto é a quantidade de texto considerada, não um dicionário de sinônimos.

✗ D) Incorreta. A tokenização divide o texto em unidades e não agrupa sinônimos em um mesmo token.

> **Dica de prova:** Sentido parecido, vetores próximos: embeddings.

R5.4. B) Modelos de linguagem visual são um subconjunto dos LLMs multimodais focado em imagem e texto

Por quê: LLMs multimodais processam texto, imagens, som e vídeo; modelos de linguagem visual integram dados visuais e textuais (GenAI-1.1.4).

✗ A) Incorreta. São conceitos relacionados, mas com escopos diferentes.

✗ C) Incorreta. Modelos de linguagem visual integram imagem e texto, e os multimodais processam várias modalidades.

✗ D) Incorreta. A relação é inversa.

> **Dica de prova:** Linguagem visual está dentro de multimodal.

DIA DA PROVA

Estratégia de prova

O exame CT-GenAI v1.1 tem 40 questões de múltipla escolha em 60 minutos, de resposta única ou 'Selecione DUAS', com pontuação ponderada: K1 e K2 valem 1 ponto e K3 valem 2, somando 46 pontos, e a aprovação exige 30 pontos (65%). Os enunciados são cenários curtos de teste em que é preciso reconhecer o conceito do syllabus ou escolher a forma correta de aplicar a GenAI; os distratores usam termos verdadeiros de outro tópico (temperatura, janela de contexto, ajuste fino, popularidade) para parecerem técnicos. Associações do tipo 1-4 com A-D e combinações de afirmações i a v são frequentes, e a armadilha padrão é inverter dois pares vizinhos. A resposta certa quase sempre dá ao LLM a entrada adequada e mantém uma pessoa verificando a saída, enquanto as erradas fazem o LLM decidir, aprovar, executar o sistema ou garantir resultado. A certificação CTFL é pré-requisito.

Como responder

✓ Tempo: 60 minutos para 40 questões dá cerca de 1,5 minuto por questão; quem faz em idioma não nativo tem 25% a mais. Não trave em uma associação longa: marque e volte.

✓ Invista onde estão os pontos: o capítulo 2 vale 16 dos 46 pontos e o 3 vale 11. As K3 (GenAI-2.2.1 a 2.2.5 e GenAI-3.1.2) valem 2 pontos cada, então leia esses cenários com calma.

✓ Em cenário longo, leia primeiro a pergunta final para saber se pedem componente, técnica, métrica, atividade, risco ou vetor; depois procure no texto só a pista que decide.

✓ Nas associações, resolva primeiro o par mais óbvio e elimine as alternativas incompatíveis; normalmente sobram duas que diferem por uma inversão, e você só precisa conferir esse par.

✓ Em 'Selecione DUAS', avalie cada alternativa como verdadeira ou falsa isoladamente; as falsas costumam propor temperatura alta, respostas longas, janela maior, popularidade ou remoção de contexto.

✓ Elimine alternativas em que o LLM aprova, decide, executa o objeto de teste ou garante ausência de erros, e as que pertencem a outra atividade do processo de teste (gerar casos quando o pedido é monitoramento, por exemplo).

✓ Use a terminologia da v1.1: o vetor que extrai dados de treinamento se chama manipulação do contexto (antes exfiltração de dados).

✓ O exame não cobra nomes de produtos ou fornecedores de LLM nem leis que o syllabus não cita, como a LGPD; se uma alternativa depende disso, trate como distrator.

Erros que derrubam candidatos

✗ Confundir contexto com dados de entrada e função com contexto ao classificar trechos de prompt.

✗ Escolher 'aumentar a temperatura' ou 'janela de contexto maior' como solução para alucinação ou qualidade.

✗ Trocar manipulação do contexto por manipulação de solicitações ou por envenenamento de dados.

✗ Confundir precisão, recuperação e acurácia, e teste A/B com modificação iterativa.

✗ Escolher encadeamento quando o cenário pede repetir um padrão (poucos disparos) ou quando ninguém sabe escrever o prompt (meta prompting).

✗ Inverter RAG e ajuste fino, ou atribuir ao front-end o trabalho do back-end.

✗ Preferir a alternativa que tira o humano da verificação por parecer mais moderna.

Na semana da prova

- ☐ Refaça os simulados cronometrando 60 minutos e anote os erros por capítulo, priorizando os capítulos 2 e 3.
- ☐ Revise a tabela das sete métricas com os exemplos do syllabus e as cinco técnicas de refinamento.
- ☐ Revise os seis componentes do prompt, a tabela de escolha de técnicas e a divisão entre prompt do sistema e do usuário.
- ☐ Revise os quatro vetores de ataque, os três riscos de privacidade e as quatro referências normativas com seus tipos.
- ☐ Revise as listas do capítulo 5: fases de adoção, critérios de seleção, riscos da Shadow AI e papéis.
- ☐ Confira a logística: 40 questões, 60 minutos (mais 25% em idioma não nativo), 30 de 46 pontos para aprovar e CTFL como pré-requisito.

REVISÃO FINAL

Folha de cola

Leia na véspera e na manhã da prova: os gatilhos e as pegadinhas mais importantes de cada domínio.

1. Engenharia de Prompts para Testes de Software Eficientes

‣ 'Atue como...', 'Você é um...' → **Função**

‣ 'O objeto de teste é...', descrição do sistema e da funcionalidade → **Contexto**

‣ Verbo no imperativo dizendo a tarefa → **Instrução**

‣ 'Não inclua', 'use apenas', 'somente dados fictícios' → **Restrições**

‣ 'Responda em tabela', 'entregue em JSON com os campos...' → **Formato de saída**

⚠ Persona é função; descrição do objeto de teste é contexto. A inversão função/contexto é um distrator clássico nas associações.

⚠ Se é a informação que o LLM vai processar, são dados de entrada. Contexto só descreve o objeto e a funcionalidade.

⚠ Excluir tipos de teste ou limitar dados é restrição. A instrução é a tarefa (gerar casos) e o formato é a forma da resposta.

2. Gerenciando os Riscos da AI Generativa em Testes de Software

‣ Caso de teste de recurso que não está nos requisitos ou no briefing → **Alucinação**

‣ Ordem que quebra dependências, causa e efeito mal interpretados → **Erro de raciocínio (validação lógica, executar e conferir)**

‣ Dados que não representam o público, tipo de teste ausente, treino concentrado em um idioma → **Viés (revisar representatividade frente à estratégia)**

‣ Menor variação possível entre execuções do mesmo prompt → **Diminuir a temperatura**

‣ Reproduzir a geração da semana passada → **Definir semente aleatória**

⚠ Temperatura mais alta aumenta a aleatoriedade e a variação, e tende a piorar a invenção de regras. Nenhuma técnica de mitigação do syllabus sobe a temperatura.

⚠ A técnica do syllabus é avaliar o prompt em vários LLMs e comparar os resultados.

⚠ Sem contexto, o risco de alucinação aumenta. A mitigação é contexto completo em formato claro e estruturado.

3. Implementação e Integração da AI Generativa em Organizações de Teste

‣ Contas pessoais de GenAI sem aprovação nem supervisão formal → **Shadow AI**

‣ Licença confusa e documentos protegidos por direitos autorais → **Propriedade intelectual vaga**

‣ Empresa regulada com ferramentas não aprovadas → **Não conformidade com padrões e regulamentos, consequências legais**

‣ Por onde começar a estratégia de GenAI → **Objetivos mensuráveis, como aumento de produtividade**

‣ Auditores querem saber o que foi gerado com GenAI → **Obrigações de transparência**

⚠ O caminho é definir estratégia e etapas de integração; proibir não é o proposto e escolha individual sem aprovação é a própria Shadow AI.

⚠ Ferramentas gratuitas não aprovadas podem ter segurança frágil e licenças pouco claras.

⚠ Só existem três: descoberta; iniciação e definição de uso; utilização e iteração. Monitoramento acontece dentro da utilização e iteração.

4. Infraestrutura de teste com tecnologia LLM para Teste de Software

‣ Onde o testador digita consultas e comandos → **Front-end**

‣ Autentica, recupera dados, prepara o prompt, pós-processa a saída → **Back-end**

‣ Vai além do cliente-servidor tradicional → **O LLM interpreta e raciocina sobre produtos de teste**

‣ LLM via API de terceiros ou modelo próprio → **LLM integrado à arquitetura: serviço de terceiros via API ou modelo interno personalizado**

‣ Dados que mudam toda semana, sem retreinar → **RAG**

⚠ Está invertido: RAG recupera trechos e os usa como contexto; ajuste fino atualiza os pesos.

⚠ A resposta é RAG com as fontes de dados da empresa, sem retreinar. Temperatura só controla aleatoriedade.

⚠ Pós-processar, autenticar, recuperar dados e montar o prompt é trabalho do back-end.

5. Introdução à AI Generativa para Teste de Software

‣ Regras escritas por especialistas, 'se... então' para classificar → **AI simbólica**

‣ Preparar dados e escolher atributos para prever problemas → **Machine learning clássico**

‣ Aprende sozinho as características a partir de milhares de imagens → **Deep learning**

‣ Cria scripts ou textos novos imitando padrões do treinamento → **AI generativa (construída sobre deep learning)**

‣ Palavras de sentido parecido ficam próximas → **Embeddings (vetores numéricos)**

⚠ A janela define quanto texto o modelo considera, não o que ele sabe. O efeito descrito é mais coerência em textos longos com mais custo computacional e tempo de processamento.

⚠ O tamanho da janela é característica do modelo. O que excede não é considerado e detalhes relevantes podem ficar de fora.

⚠ O uso não altera os pesos e a tokenização de um mesmo texto é a mesma. A variação vem da inferência probabilística, que depende de hiperparâmetros.

GABARITO

Respostas do diagnóstico

D1. A) Encadeamento de prompts → revise o capítulo 1

Por quê: O encadeamento de prompts é útil quando a tarefa é complicada e exige decomposição em subtarefas e verificação sistemática dos resultados intermediários (GenAI-2.1.2).

✗ B) Incorreta. Não houve um exemplo para orientar a resposta.

✗ C) Incorreta. No meta prompting o LLM gera ou refina o prompt, o que não ocorreu.

✗ D) Incorreta. Não foram fornecidos exemplos no prompt.

> **Dica de prova:** Etapas sucessivas com verificação: encadeamento.

D2. D) 'Identifique ambiguidades nos requisitos abaixo e classifique cada uma.' → revise o capítulo 1

Por quê: Instruções claras, imperativas e concisas descrevem a tarefa e os requisitos relevantes para executá-la (GenAI-2.1.1).

✗ A) Incorreta. Essa linha especifica o formato de saída.

✗ B) Incorreta. Essa linha define a função, isto é, a persona do modelo.

✗ C) Incorreta. Essa linha indica os dados de entrada a processar.

> **Dica de prova:** Imperativo que diz a tarefa: instrução.

D3. B) Gerar o relatório de conclusão de teste com sucessos e lições aprendidas do ciclo → revise o capítulo 1

Por quê: No monitoramento e controle, a GenAI apoia a conclusão de testes e o aprendizado contínuo ao gerar relatórios de conclusão com sucessos e lições aprendidas (GenAI-2.2.4).

✗ A) Incorreta. O planejamento seguinte deveria aproveitar os dados e as lições do ciclo.

✗ C) Incorreta. Um ranking de pessoas não é um insight de conclusão de teste descrito no syllabus.

✗ D) Incorreta. Criar casos de teste é projeto de teste, não encerramento do ciclo.

> **Dica de prova:** Fim de ciclo: relatório de conclusão com lições.

D4. D) Entradas adulteradas podem desviar o resultado da análise feita pela AI → revise o capítulo 2

Por quê: Entre os vetores de ataque do syllabus, a manipulação de solicitações é introduzir dados que atrapalhem o resultado da AI (GenAI-3.2.2).

✗ A) Incorreta. Código com backdoor é geração de código malicioso.

✗ B) Incorreta. Alterar dados de treinamento é envenenamento de dados.

✗ C) Incorreta. Revelar dados de treinamento é o objetivo da manipulação do contexto.

> **Dica de prova:** Entrada adulterada na análise: manipulação de solicitações.

D5. A) ISO/IEC 42001:2023 é norma; Lei de AI da UE é regulamento; NIST AI RMF é estrutura → revise o capítulo 2

Por quê: O syllabus classifica as referências por tipo: normas ISO/IEC 42001:2023 e ISO/IEC 23053:2022, regulamento Lei de AI da UE e estrutura NIST AI RMF (GenAI-3.4.1).

✗ B) Incorreta. A Lei de AI da UE é regulamento, e o NIST AI RMF é estrutura.

✗ C) Incorreta. A ISO/IEC 42001 é norma, a Lei de AI da UE é regulamento e o NIST AI RMF é estrutura.

✗ D) Incorreta. A ISO/IEC 42001 é norma, e a Lei de AI da UE é regulamento.

> **Dica de prova:** ISO: norma. UE: regulamento. NIST: estrutura.

D6. C) A complexidade da tarefa e os recursos de computação que ela exige → revise o capítulo 2

Por quê: Gerar uma imagem pode consumir tanta energia quanto carregar um smartphone, enquanto gerar texto consome pequena porcentagem disso; limitar interações desnecessárias entre modelos ajuda a reduzir o impacto (GenAI-3.3.1).

✗ A) Incorreta. O armazenamento do resultado não é o fator descrito no syllabus.

✗ B) Incorreta. O syllabus não relaciona o consumo ao idioma do prompt.

✗ D) Incorreta. A revisão humana não é o fator de consumo de energia descrito.

> **Dica de prova:** Tarefa mais complexa, mais energia.

D7. D) Compartilhando bibliotecas de padrões de prompt e lições aprendidas → revise o capítulo 3

Por quê: Comunidades internas de prática fazem reuniões regulares para destacar aplicações bem-sucedidas, discutir desafios e refinar práticas (GenAI-5.2.2).

✗ A) Incorreta. O objetivo é o compartilhamento contínuo de conhecimento, não sua centralização.

✗ B) Incorreta. O syllabus recomenda praticar com vários LLMs/SLMs.

✗ C) Incorreta. O foco é o desenvolvimento de habilidades práticas.

> **Dica de prova:** Comunidade de prática: biblioteca de prompts e lições aprendidas.

D8. B) Avaliar recursos do LLM, refinar prompts e avaliar testware gerado por AI → revise o capítulo 3

Por quê: As habilidades essenciais combinam engenharia de prompts, compreensão de janelas de contexto e métodos de revisão com experiência em teste e no domínio (GenAI-5.2.1).

✗ A) Incorreta. Treinar e publicar modelos não está entre as competências essenciais descritas para testadores.

✗ C) Incorreta. Essas são atividades de desenvolvimento de modelos, não do teste com GenAI.

✗ D) Incorreta. Essas são atividades de gestão de contratos, não competências de teste com GenAI.

> **Dica de prova:** Testador com GenAI: avalia o LLM, refina prompts e revisa a saída.

D9. A) Fase de descoberta → revise o capítulo 3

Por quê: As fases da adoção são descoberta, iniciação e definição de uso, e utilização e iteração, e podem correr em paralelo para casos de uso diferentes (GenAI-5.1.4).

✗ B) Incorreta. Monitoramento não é uma fase de adoção; ele ocorre dentro da utilização e iteração.

✗ C) Incorreta. Essa é a fase avançada de integração total e monitoramento contínuo.

✗ D) Incorreta. Essa fase identifica e prioriza casos de uso práticos e avalia a infraestrutura.

> **Dica de prova:** Treinar, dar acesso e experimentar: descoberta.

D10. A) Ajuste fino do modelo → revise o capítulo 4

Por quê: O ajuste fino é um processo de aprendizado supervisionado que usa exemplos rotulados para atualizar os pesos do modelo e adaptá-lo a tarefas ou domínios específicos (GenAI-4.2.1).

✗ B) Incorreta. Meta prompting usa o LLM para gerar ou refinar prompts.

✗ C) Incorreta. O RAG recupera dados externos no momento da consulta, sem treinar o modelo.

✗ D) Incorreta. Poucos disparos coloca exemplos no prompt, sem alterar os pesos do modelo.

> **Dica de prova:** Treino adicional com dados da empresa: ajuste fino.

D11. C) Operações de LLM (LLMOps) → revise o capítulo 4

Por quê: LLMOps reúne práticas e ferramentas para implantação, monitoramento e manutenção de LLMs em produção, e as decisões variam conforme a abordagem de uso adotada (GenAI-4.2.2).

✗ A) Incorreta. O ajuste fino adapta o modelo a tarefas específicas; é uma técnica, não a operação do ciclo de vida.

✗ B) Incorreta. O RAG enriquece a geração com dados recuperados e não cobre implantação e manutenção.

✗ D) Incorreta. A engenharia de prompts trata da criação e do refinamento de prompts.

> **Dica de prova:** Operar LLM em produção: LLMOps.

D12. B) Passagem da execução baseada em scripts para uma automação orientada por objetivos → revise o capítulo 4

Por quê: Os assistentes com agentes podem executar análise, modelagem, implementação, execução e relatório de teste de forma semiautônoma, encurtando ciclos de feedback (GenAI-4.1.3).

✗ A) Incorreta. A automação se amplia, não volta a ser manual.

✗ C) Incorreta. Agentes também podem alucinar e errar, e seus resultados precisam de verificação.

✗ D) Incorreta. Os assistentes transformam histórias de usuário ou requisitos em artefatos de teste.

> **Dica de prova:** Agentes: de scripts para objetivos.

Agora, treine no formato real

Faça as 6 provas completas do curso ISTQB CT-GenAI: 240 questões inéditas, tempo do exame, explicação de cada alternativa e link da documentação para revisar.

Ver as 6 provas completas e o cupom do mês →

Até o dia da prova, uma revisão por dia no seu feed

Instagram @strecht.simulados

Um post de prova por dia: questão comentada, pegadinhas e o "não confunda" de cada certificação.

Seguir no Instagram →

YouTube Strecht Simulados

Shorts diários com pegadinhas de prova e, aos sábados, um simulado comentado.

Inscrever-se no canal →

Material independente de preparação, sem afiliação ou endosso de ISTQB. As questões são inéditas e não reproduzem a prova real.