



Plano de Testes DBC Captação/Provas

ÍNDICE

1. Introdução	3
2. Abordagem de Testes	4
3. Classificação de Bugs	11
4. Cenários de Testes	12
5. Critérios de Testes	15
6. Cronograma	16
7. Métrica Final da Execução dos Casos de Testes Planejados	16
8. Bug Report	19
9. Considerações finais	24

1 Introdução

1.1 Sobre esse Plano de Testes

Este documento apresenta o plano de testes desenvolvido pela equipe de Garantia de Qualidade (QA) para o sistema DBC - Captação/Provas. Nele, detalharemos as funcionalidades a serem testadas, os tipos de testes definidos e os recursos a serem empregados para garantir a qualidade do sistema.

1.2 Sobre o sistema DBC - Captação/Provas

O sistema DBC - Captação/Provas foi concebido para atender à necessidade de um sistema de avaliação de candidatos no processo seletivo Vem Ser DBC. Ele permite a construção de um banco de questões e a criação de provas, facilitando a gerência de usuários, a elaboração de questões objetivas e técnicas, a formação de provas personalizadas e a geração de relatórios de desempenho abrangentes, proporcionando uma avaliação completa do processo seletivo.

1.3 Sobre o Escopo dos testes

Os testes englobam os níveis de Integração, Unitários, Funcionais e Aceitação, tanto manualmente quanto por meio de automação. Os testes unitários serão realizados durante a fase de codificação do sistema pelos desenvolvedores backend e frontend.

1.4 Documentações do Projeto

Documento	Irá fazer parte do projeto? (sim ou não)	Observações
Especificações de Requisitos	Sim	Irá definir os critérios mínimos para a estória do usuário ser aceita.
Especificações das Funcionalidades	Sim	Descreverá as funcionalidades mínimas do Sistema.
Especificação das Regras de negócio	Sim	Definirá as regras e os comportamentos do sistema.
Trello	Sim	Incluirá todo o fluxo de trabalho da equipe.

2. Abordagem de Testes

2.1 Categorização dos Requisitos em Funcionais x Não Funcionais

Requisito Funcional	Requisito Não Funcional
RF001 - O sistema deve permitir o cadastro de questões objetivas.	NF001 - O sistema deverá ser desenvolvido nas linguagens de programação Java e Javascript.
RF002 - O sistema deve permitir o cadastro de questões práticas.	NF002 - O banco de dados deverá ser o PostgreSQL
RF003 - O sistema deve permitir a edição de questões cadastradas.	NF003 - O sistema deverá ter um Design para mobile responsivo.
RF004 - O sistema deve permitir a remoção lógica de questões cadastradas.	NF004 - O sistema deverá ser acessível via Browser, como Chrome e Edge.
RF005 - O sistema deve permitir a visualização de questões cadastradas.	NF005 - O sistema deverá possuir níveis de acesso para diferenciar usuários comuns de usuários administrativos que acessaram áreas restritas do sistema.
RF006 - O sistema deve permitir a criação de provas com questões selecionadas.	NF006 - O sistema deverá ter paginação obrigatória para os principais endpoints.
RF007 - O sistema deve permitir a edição de provas cadastradas.	NF007 - O sistema deverá garantir boas métricas de acessibilidade e performance.
RF008 - O sistema deve permitir a remoção de provas cadastradas.	
RF009 - O sistema deve permitir o filtro de questões por diferentes critérios.	
RF010 - O sistema deve permitir o filtro de provas por diferentes critérios.	
RF011 - O sistema deve permitir que candidatos realizem provas.	

2.2 Detalhamento da abordagem de teste

Testes funcionais:

Objetivo	RF001 - O sistema deve permitir o cadastro de questões objetivas.			
Criticidade	Crítico			
Técnica	(X) manual		(X) automática	
Estágio do teste	Integração ()	Sistema (X)	Unidade ()	Aceitação ()
Abordagem do teste	Caixa branca ()		Caixa preta (X)	
Comportamento esperado	Cadastrar questão objetiva no banco de questões do sistema			

Objetivo	RF002 - O sistema deve permitir o cadastro de questões práticas.			
Criticidade	Crítico			
Técnica	(X) manual		(X) automática	
Estágio do teste	Integração ()	Sistema (X)	Unidade ()	Aceitação ()
Abordagem do teste	Caixa branca ()		Caixa preta (X)	
Comportamento esperado	Cadastrar questão prática no banco de questões do sistema			

Objetivo	RF003 - O sistema deve permitir a edição de questões cadastradas.			
Criticidade	Grave			
Técnica	(X) manual		(X) automática	
Estágio do teste	Integração ()	Sistema (X)	Unidade ()	Aceitação ()
Abordagem do teste	Caixa branca ()		Caixa preta (X)	
Comportamento esperado	A alteração deve ser salva com sucesso no banco de questões			

Objetivo	RF004 - O sistema deve permitir a remoção lógica de questões cadastradas.			
Criticidade	Moderado			
Técnica	(X) manual		(X) automática	
Estágio do teste	Integração ()	Sistema (X)	Unidade ()	Aceitação ()
Abordagem do teste	Caixa branca ()		Caixa preta (X)	
Comportamento esperado	Deve excluir a questão selecionada do banco de questões			

Objetivo	RF005 - O sistema deve permitir a visualização de questões cadastradas.			
Criticidade	Moderado			
Técnica	(X) manual		(X) automática	
Estágio do teste	Integração ()	Sistema (X)	Unidade ()	Aceitação ()
Abordagem do teste	Caixa branca ()		Caixa preta (X)	
Comportamento esperado	Deve ser possível visualizar a lista de questões cadastradas no banco de questões			

Objetivo	RF006 - O sistema deve permitir a criação de provas com questões selecionadas.			
Criticidade	Crítico			
Técnica	(X) manual		(X) automática	
Estágio do teste	Integração ()	Sistema (X)	Unidade ()	Aceitação ()
Abordagem do teste	Caixa branca ()		Caixa preta (X)	
Comportamento esperado	Cadastrar com sucesso uma prova no sistema			

Objetivo	RF007 - O sistema deve permitir a edição de provas cadastradas.			
Criticidade	Grave			
Técnica	() manual		(X) automática	
Estágio do teste	Integração ()	Sistema (X)	Unidade ()	Aceitação ()
Abordagem do teste	Caixa branca ()		Caixa preta (X)	
Comportamento esperado	Salvar com sucesso alterações feitas em uma prova cadastrada			

Objetivo	RF008 - O sistema deve permitir a remoção de provas cadastradas.			
Criticidade	Moderado			
Técnica	(X) manual		(X) automática	
Estágio do teste	Integração ()	Sistema (X)	Unidade ()	Aceitação ()
Abordagem do teste	Caixa branca ()		Caixa preta (X)	
Comportamento esperado	Deve excluir com sucesso uma prova cadastrada no sistema.			

Objetivo	RF009 - O sistema deve permitir o filtro de questões por diferentes critérios.			
Criticidade	Moderado			
Técnica	(X) manual		(X) automática	
Estágio do teste	Integração ()	Sistema (X)	Unidade ()	Aceitação ()
Abordagem do teste	Caixa branca ()		Caixa preta (X)	
Comportamento esperado	Filtrar com sucesso a exibição de questões de acordo com o filtro aplicado			

Objetivo	RF010 - O sistema deve permitir o filtro de provas por diferentes critérios.			
Criticidade	Leve			
Técnica	(X) manual		(X) automática	
Estágio do teste	Integração ()	Sistema (X)	Unidade ()	Aceitação ()
Abordagem do teste	Caixa branca ()		Caixa preta (X)	
Comportamento esperado	Filtrar com sucesso a exibição de provas de acordo com o filtro aplicado			

Objetivo	RF011 - O sistema deve permitir que candidatos realizem provas.			
Criticidade	Crítico			
Técnica	(X) manual		() automática	
Estágio do teste	Integração ()	Sistema (X)	Unidade ()	Aceitação ()
Abordagem do teste	Caixa branca ()		Caixa preta (X)	
Comportamento esperado	Responder todas as questões e concluir prova com sucesso			

Testes não funcionais:

Objetivo	NF003 - O sistema deverá ter um design responsivo			
Criticidade	Moderado			
Técnica	(X) manual		() automática	
Estágio do teste	Integração ()	Sistema (X)	Unidade ()	Aceitação ()
Abordagem do teste	Caixa branca ()		Caixa preta (X)	
Comportamento esperado	Os elementos se ajustam de acordo com as dimensões do dispositivo			

Objetivo	NF004 - O sistema deverá ser acessível via diferentes browsers.			
Criticidade	Grave			
Técnica	() manual		(X) automática	
Estágio do teste	Integração ()	Sistema (X)	Unidade ()	Aceitação ()
Abordagem do teste	Caixa branca ()		Caixa preta (X)	
Comportamento esperado	A página abre em diferentes navegadores			

Objetivo	NF005 - O sistema deverá possuir níveis de acesso para diferenciar usuários no sistema.			
Criticidade	Crítico			
Técnica	() manual		(X) automática	
Estágio do teste	Integração ()	Sistema (X)	Unidade ()	Aceitação ()
Abordagem do teste	Caixa branca ()		Caixa preta (X)	
Comportamento esperado	Reconhecer tipo de usuário e conceder os devidos acessos			

Objetivo	NF007 - O sistema deverá garantir boas métricas de acessibilidade e performance.			
Criticidade	Grave			
Técnica	(X) manual		() automática	
Estágio do teste	Integração ()	Sistema ()	Unidade ()	Aceitação (X)
Abordagem do teste	Caixa branca ()		Caixa preta (X)	
Comportamento esperado	Ser acessível para usuários com deficiência temporária ou permanente e possuir ?? pontos de performance			

2.3 Ferramentas

As seguintes ferramentas serão empregadas neste projeto de testes:

Ferramenta de automação	
Ferramenta	Versão
Rest Assured	5.4.0
Cypress	12.14.0
Jenkins	2.440.1 imagem docker: jenkins/jenkins:its-jdk17
Allure Report	2.20.1

2.4 Definições do ambiente de testes

- Testes unitários:
 - Serão executados pelos desenvolvedores durante a fase de codificação.
- Testes Automatizados e Manuais:
 - Responsabilidade dos QAs, utilizando dados reais e/ou dados fictícios (faker).
- Infraestrutura:
 - Máquinas locais com conexão à internet serão utilizadas para realização dos testes.

4. Ferramentas utilizadas:

- a. Testes Manuais de API: Postman.
- b. Testes Automatizados de API: Java + Rest Assured.
- c. Testes Automatizados de Frontend: JavaScript + Cypress.
- d. Ambientes de Desenvolvimento: IntelliJ e VScode.
- e. Geração de Relatórios: Jenkins + Allure Report.

5. Tecnologias Utilizadas no Desenvolvimento do Sistema:

- Linguagens: Java, HTML, CSS e TypeScript.

3. Classificação de Bugs

Os Bugs serão classificados com as seguintes severidades:

Ferramenta de automação		
ID	Nível de Severidade	Descrição
1	Crítico	Defeito que impede o pleno funcionamento das principais funcionalidades do sistema ou que permite o vazamento de dados sensíveis do usuário. Como por exemplo: Falha no login, no cadastro, nas permissões de acessos, etc.
2	Grave	Funcionalidade que não opera conforme o esperado, podendo causar efeitos irreversíveis em decorrência de inputs incomuns. Como por exemplo: Erro de interface, problema de acessibilidade, etc.
3	Moderado	Funcionalidade que não atende completamente aos critérios de aceitação, porém não afeta significativamente o funcionamento geral do sistema. Como por exemplo: Falhas na exibição de mensagens de erro ou sucesso, problemas na navegação, links quebrados, funcionalidades auxiliares comprometidas, etc.
4	Leve	Problemas de menor impacto na funcionalidade, porém que interferem na experiência do usuário. Como por exemplo: erros ortográficos e pequenos problemas de interface do usuário (UI).

4. Cenários de Testes (Manuais e Automatizados)

Antes mesmo dos times de desenvolvimento nos entregarem alguma funcionalidade para teste, decidimos mapear alguns cenários de testes baseados nas Histórias de Usuário e nas regras de negócio definidas pelo time. Abaixo traremos apenas um panorama geral. Para uma visualização mais detalhada de todas as histórias de usuário, de todos os cenários, com entradas e saídas esperadas e com a descrição de cada um dos testes utilizando a linguagem Gherkin, acesse o documento no link a seguir: [■ Captação _ Provas - Histórias de Usuários + CTs \(1\).pdf](#)

Feature 01: Cadastrar Questões Objetivas

Cenários de testes (CT):

- CT001** - Criar Questão Objetiva com Dados Válidos
- CT002** - Criar Questão Objetiva com Título Excedendo Limite
- CT003** - Criar Questão Objetiva com Título em Branco
- CT004** - Criar Questão Objetiva com Enunciado Excedendo Limite
- CT005** - Criar Questão Objetiva com Enunciado em Branco
- CT006** - Criar Questão Objetiva com Mais de 5 Alternativas
- CT007** - Criar Questão Objetiva sem Alternativa Correta
- CT008** - Criar Questão Objetiva não selecionando nível de dificuldade

Feature 02: Cadastrar Questões Práticas

Cenários de testes (CT):

- CT009** - Criar Questão Prática com Dados Válidos
- CT010** - Criar Questão Prática sem Inputs e Outputs
- CT011** - Criar Questão Prática com Título Excedendo Limite
- CT012** - Criar Questão Prática com Título em Branco
- CT013** - Criar Questão Prática com Enunciado em Branco
- CT014** - Criar Questão Prática com Enunciado Excedendo Limite
- CT015** - Criar Questão Prática não selecionando nível de dificuldade

Feature 03: Editar Questões

Cenários de testes (CT):

- CT016** - Editar Questão Objetiva com Dados Válidos
- CT017** - Editar Questão Objetiva para Adicionar Alternativas
- CT018** - Editar Questão Prática com Dados Válidos
- CT019** - Editar Questão Prática para Adicionar Inputs e Outputs
- CT020** - Editar Questão Objetiva sem escolher uma alternativa como correta
- CT021** - Editar Questão Prática removendo os Inputs e Outputs

Feature 04: Remover Questões

Cenários de testes (CT):

- CT022** - Remover Questão com Confirmação
- CT023** - Remover Questão sem Confirmação
- CT024** - Usuário sem permissão Remover Questão

Feature 05: Visualizar Questões

Cenários de testes (CT):

- CT025** - Visualizar lista de questões
- CT026** - Filtrar questões por tipo e nível de dificuldade
- CT027** - Selecionar uma questão para visualização detalhada, edição ou exclusão
- CT028** - Tentativa de acessar a página de listagem de questões sem permissão de Tech Lead

Feature 06: Criação da prova com questões

Cenários de testes (CT):

CT029 - Criar uma prova com questões selecionadas

CT030 - Criar uma prova selecionando questões aleatórias

CT031 - Título com mais de 100 caracteres

CT032 - Tentativa de criar uma prova sem selecionar questões

CT033 - Tentativa de criar uma prova sem definir o título

Feature 07: Editar prova

Cenários de testes (CT):

CT034 - Alterar título de prova

CT035 - Alterar questão da prova

CT036 - Editar prova inválida

CT037 - Salvar alterações com título maior do que o permitido

CT038 - Salvar alterações com título em branco

CT039 - Salvar alterações com quantidade de questões inválida

CT040 - Tentar adicionar mais de 10 questões

CT041 - Remover todas as questões da prova

Feature 08: Excluir prova

Cenários de testes (CT):

CT042 - Excluir prova com sucesso

CT043 - Excluir prova de edições anteriores

Feature 09: Filtrar Prova e Questão

Cenários de testes (CT):

- CT044** - Filtrar Prova e Questão com todos dados válidos
- CT045** - Filtrar Prova somente com o filtro "Versão da prova"
- CT046** - Filtrar Prova ou Questão somente digitando "Palavra chave Válida"
- CT047** - Filtrar Questão somente com o filtro "Nível de dificuldade Válido"
- CT048** - Filtrar Questão somente com o filtro "Tipo de questão Válido"
- CT049** - Filtrar Prova ou Questão não aplicando nenhum filtro
- CT050** - Filtrar Prova ou Questão somente digitando "Palavra chave Inválida"

Feature 10: Candidato - Realizar Prova

Cenários de testes (CT):

- CT051** - Realizar prova com sucesso
- CT052** - Realizar prova com questões em branco
- CT053** - Realizar prova sem selecionar uma linguagem de programação

5. Critérios de Testes

5.1 Critérios de suspensão

- 50% ou mais dos testes falham.

5.2 Critérios de saída

- 90% de cobertura de teste, sendo destes, 30% manuais e 60% automatizados.

6. Cronograma





Tipo de teste	Data de início	Data de término
Compreender o projeto	23/02/2024	26/02/2024
Escrever Estória de Usuário	26/02/2024	26/02/2024
Planejar teste	27/02/2024	27/02/2024
Implementar, executar e avaliar testes	28/02/2024	04/03/2024

7. Métrica Final da Execução dos Casos de Testes Planejados

Deixamos claro que esse foi um plano de testes planejado e executado durante o Trabalho Final do Vem Ser DBC 13ª Edição. Portanto, tivemos um curto período entre a liberação das funcionalidades pelos times de desenvolvimento e a entrega do trabalho. Pensando nisso, calculamos os riscos e priorizamos a automatização dos endpoints mais importantes, varrendo o máximo possível das funcionalidades com o tempo disponível, para garantir maior qualidade na entrega para o cliente.

7.1. Testes Exploratórios

Realizamos testes exploratórios tanto no Swagger da API quanto na Interface desenvolvida pelo time de Front-End. Eles foram essenciais para o entendimento de todo o time sobre as funcionalidades do sistema, e criaram a base para que pudéssemos partir para a automatização dos endpoints mais importantes. Traremos aqui a documentação de alguns desses testes através de imagens que demonstram o passo a passo de como fazer as requisições para testes de determinadas funcionalidades.

-  Criar Questão Objetiva - Interface.pdf
-  Criar Questão Prática - Interface.pdf
-  Criar Prova - Swagger API.pdf
-  Listar Questões por Dificuldade - Swagger API.pdf


7.2. Testes Manuais com Postman

Quant. Casos de testes planejados	Quant. Casos de testes realizados	Quant. Casos de testes "rodando"	Quant. Casos de testes "falhando"
53	182	175	7

tf_captacao_postman - Run results

Run Again

Automate Run ▾

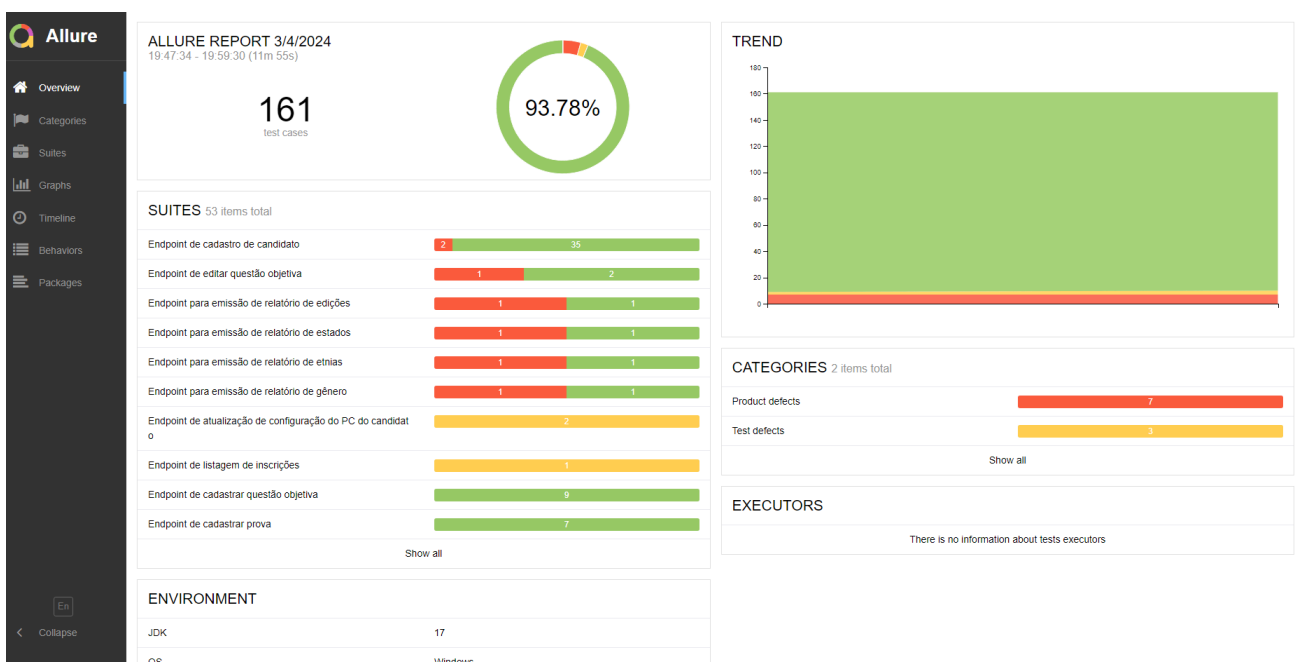
 Ran today at 21:12:47 · [View all runs](#)

Source	Environment	Iterations	Duration	All tests	Avg. Resp. Time
Runner	tf_postman	1	1m 34s	182	643 ms

All Tests Passed (175) Failed (7) Skipped (0)

7.3. Testes Automatizados de API com RestAssured

Quant. Casos de testes planejados	Quant. Casos de testes realizados	Quant. Casos de testes "rodando"	Quant. Casos de testes "falhando"
53	161	151	10



7.4. Testes Automatizados de Front-end com Cypress

Quant. Casos de testes planejados	Quant. Casos de testes realizados	Quant. Casos de testes "rodando"	Quant. Casos de testes "falhando"
53	21	20	1

(Run Finished)						
Spec	Tests	Passing	Failing	Pending	Skipped	
✓ form1steps.cy.js	02:32	18	18	-	-	-
✓ form2steps.cy.js	00:32	2	2	-	-	-
✗ loginCaptacao.cy.js	00:04	1	-	1	-	-
✗ 1 of 3 failed (33%)	03:10	21	20	1	-	-

8. Bug Report

Para boa organização e comunicação de toda a squad, especialmente do time de QA com os times de desenvolvimento, desenvolvemos um sistema simples e eficaz de Bug Report. Utilizamos uma coluna no Trello, nossa principal ferramenta de organização, para salvar cards detalhados que explicam onde está o bug, a sua criticidade, como ele está acontecendo e o que deveria acontecer idealmente, além de direcionarmos os cards para os desenvolvedores da stack específica responsável pela funcionalidade, para a pronta correção.



8.1 Bugs corrigidos durante a Sprint:

Abaixo seguem reports dos problemas encontrados e corrigidos durante a Sprint:

8.1.1 Título: Bug Report - Retorno 500 ao tentar listar provas por id no endpoint: /prova/pegar-prova?idProva=258.

Descrição:

Passos para Reproduzir:

1. Acessar o endpoint de listagem de provas por id.
2. Inserir o id válido da prova desejada.
3. Tentar listar a prova.
4. Observar o retorno da requisição.

Comportamento Esperado: Espera-se que ao inserir um id válido da prova, o sistema liste corretamente as informações relacionadas à prova correspondente.

Comportamento Atual: Ao tentar listar a prova por id, o sistema retorna um erro 500, indicando uma falha interna.

Passos para Possível Solução:

1. Colocar {idProva} no endpoint, por exemplo: /prova/pegar-prova/{idProva}.

8.1.2 Título: Bug Report - Erro 500 ao tentar listar todos os candidatos

Descrição: Quando tento listar todos os candidatos no endpoint “/candidato” com o perfil de Gestão de Pessoas, o sistema retorna um erro 500. Isso impede que eu visualize ou acesse as informações dos candidatos cadastrados.

Passos para Reproduzir:

1. Acesse a funcionalidade de listagem de candidatos com perfil GP.
2. Tente visualizar a lista completa de candidatos.

Comportamento Esperado: Deveria ser possível visualizar a lista completa de candidatos sem erros.

Comportamento Observado: Ao tentar acessar a lista de candidatos, o sistema retorna um erro 500, impedindo a visualização dos dados.

Capturas de Tela: em anexo

8.1.3 Título: Bug Report - Erro 500 ao tentar listar todos os candidatos com perfil GP

Descrição: Quando tento listar todos os candidatos no endpoint “/candidato”, o sistema retorna um erro 500. Isso impede que eu visualize ou acesse as informações dos candidatos cadastrados.

Passos para Reproduzir:

1. Acesse a funcionalidade de listagem de candidatos.
2. Tente visualizar a lista completa de candidatos.

Comportamento Esperado: Deveria ser possível visualizar a lista completa de candidatos sem erros.

Comportamento Observado: Ao tentar acessar a lista de candidatos, o sistema retorna um erro 500, impedindo a visualização dos dados.

Possíveis Causas:

1. Problemas no Servidor: Pode haver um problema no servidor que está impedindo a listagem dos candidatos. Isso pode incluir erros de código no backend, problemas de conexão com o banco de dados, entre outros.
2. Erros de Configuração: Configurações incorretas no servidor podem estar causando o erro 500 ao tentar acessar os dados dos candidatos.
3. Sobrecarga do Sistema: Uma carga excessiva no servidor pode estar causando problemas de desempenho e resultando em erros 500 ao tentar acessar os dados.

8.2 Bugs não corrigidos durante a Sprint:

Abaixo seguem reports dos problemas encontrados e não corrigidos durante a Sprint:

8.2.1 Título: Bug Report - Retorno 200 ao listar relatórios com usuário aluno.

Descrição: Ao tentar listar as edições através de qualquer endpoint relacionado aos Formulários no sistema, quando o usuário está autenticado como Aluno, o sistema retorna erroneamente um status HTTP 200 (OK). Isso está em desacordo com o comportamento esperado, pois tal usuário não tem permissão para tal visualização e deveria ter acesso negado.

Passos para Reproduzir:

1. Acesse qualquer endpoint relacionado a Formulário no sistema (por exemplo, /relatorios/quantidade-de-pessoas-inscritas-por-neurodiversidade).
2. Esteja autenticado como aluno no sistema.
3. Faça uma solicitação GET para listar as edições.

Comportamento Esperado:

- O sistema deve retornar um status HTTP 403 (Você não tem permissão para acessar este recurso), indicando que o acesso não é permitido sem autenticação.

Comportamento Observado:

- O sistema retorna um status HTTP 200 (OK), indicando que a solicitação foi bem-sucedida, o que não está correto.

8.2.2 Título: Bug Report - Retorno 201 ao cadastrar um candidato com e-mail já cadastrado.

Descrição: Ao tentar o cadastro de candidato através do endpoint /candidato, quando o usuário está autenticado como Gestão de Pessoas, o sistema permite o cadastro de candidato com um e-mail que já está cadastrado no banco. Isso está em desacordo com o comportamento esperado, pois não deveria ser permitida a criação de candidato com e-mail já existente no banco.

Passos para Reproduzir:

1. Acesse o endpoint relacionado a Candidato no sistema (por exemplo, /candidato).
2. Esteja autenticado como Gestão de Pessoas no sistema.
3. Faça uma solicitação POST para cadastrar um candidato com e-mail já cadastrado.

Comportamento Esperado:

- O sistema deve retornar um status HTTP 400 (Erro na inserção dos dados ou e-mail já cadastrado), indicando que já existe um e-mail cadastrado no sistema.

8.2.3 Título: Bug Report - Retorno 204 ao excluir formulário quando logado com aluno.

Descrição: Ao tentar a exclusão de formulário através do endpoint /formulario/delete-fisico/9 relacionado aos Formulários no sistema, quando o usuário está autenticado como Aluno, o sistema retorna HTTP 204. Isso está em desacordo com o comportamento esperado, pois tal usuário não tem permissão para excluir o formulário e deveria ter acesso negado.

Passos para Reproduzir:

1. Acesse o endpoint de exclusão de Formulário no sistema (por exemplo, /formulario/delete-fisico/9).
2. Esteja autenticado como aluno no sistema.
3. Faça uma solicitação DELETE para deletar um formulário.

Comportamento Esperado:

- O sistema deve retornar um status HTTP 403 (Você não tem permissão para acessar este recurso), indicando que o acesso não é permitido sem autenticação.

Comportamento Observado:

- O sistema retorna um status HTTP 204, indicando que a solicitação foi bem-sucedida, o que não está correto.

8.2.4 Título: Bug Report - Retorno incorreto de status HTTP ao listar edições sem autenticação

Descrição: Ao tentar listar as edições através de qualquer endpoint relacionado às Edições no sistema, quando o usuário não está autenticado, o sistema retorna erroneamente um status HTTP 200 (OK). Isso está em desacordo com o comportamento esperado, pois sem autenticação o acesso a essas informações deve ser negado.

Passos para Reproduzir:

1. Acesse qualquer endpoint relacionado às Edições no sistema (por exemplo, relatorios/quantidade-de-pessoas-inscritas-por-edicao).
2. Não esteja autenticado no sistema.
3. Faça uma solicitação GET para listar as edições.

Comportamento Esperado:

- O sistema deve retornar um status HTTP 403 (Você não tem permissão para acessar este recurso), indicando que o acesso não é permitido sem autenticação.

Comportamento Observado:

- O sistema retorna um status HTTP 200 (OK), indicando que a solicitação foi bem-sucedida, o que não está correto, pois o acesso sem autenticação não deve ser permitido.

8.2.5 Título: Bug Report - a API está deixando editar uma questão objetiva deixando ela com todas as alternativas falsas

Descrição: Ao tentar editar uma questão objetiva deixando todas as duas alternativas falsas, a API está permitindo essa edição.

Passos para Reproduzir:

1. Acesse o endpoint de editar questão objetiva (questao/editar-objetiva/{idQuestao}).
2. Esteja autenticado no sistema com login de instrutor.
3. Faça uma solicitação PUT para editar uma questão objetiva.
4. Envie no body uma questão com todas as alternativas falsas

Comportamento Esperado:

- O sistema deve retornar um status HTTP 400 (Bad request), indicando que ao menos uma alternativa deve ser verdadeira.

Comportamento Observado:

- O sistema retorna um status HTTP 200 (OK), indicando que a edição foi bem-sucedida, o que não está correto, pois uma questão objetiva não pode ter todas as alternativas falsas

9. Considerações Finais

Durante a elaboração do Trabalho Final, tivemos a oportunidade de aprimorar uma ampla gama de habilidades que adquirimos ao longo da Stack de QA. Este processo de aprendizado prático foi fundamental para consolidar nosso conhecimento teórico. No entanto, como é comum em projetos complexos, enfrentamos alguns desafios significativos. Um deles foi a inacessibilidade à API por quase três dias, o que teve um impacto substancial em todos os aspectos do desenvolvimento, tanto no frontend quanto no backend.

Além da inacessibilidade temporária, outro problema recorrente que enfrentamos foi a lentidão da API ao longo de todo o período de desenvolvimento. Essa lentidão não apenas afetou diretamente os testes que estávamos conduzindo, mas também dificultou o progresso no desenvolvimento de ambas as partes do sistema. Adicionalmente, alguns bugs reportados não puderam ser resolvidos a tempo pelo backend, o que levou à persistência de problemas que poderiam ter sido evitados.

No frontend, encontramos uma série de desafios adicionais. Problemas técnicos levaram membros da equipe a ficarem sem acesso aos seus computadores de trabalho, o que prejudicou significativamente nossa capacidade de avançar com o desenvolvimento e realizar testes eficazes. Ainda mais, as falhas na API complicaram ainda mais a situação, impedindo a integração adequada com o backend e, por consequência, a automação dos testes das telas que foram desenvolvidas.

Apesar desses contratemplos, permanecemos comprometidos em alcançar nossos objetivos e em superar esses obstáculos. Estamos confiantes de que essas experiências nos proporcionarão lições valiosas que poderemos aplicar em futuros projetos.