

CENTRO UNIVERSITÁRIO – CATÓLICA DE SANTA CATARINA

**ENGENHARIA DE SOFTWARE
DOCUMENTO DE ESPECIFICAÇÃO FUNCIONAL
E PLANO DE TESTES
QUALIDADE DE SOFTWARE**

**MATHEUS JOHAN FORBICI
NATHALIA ALINE BERRI
WILLIAM PEREIRA**

Joinville, 20 de Abril de 2025

1. Objetivo

Este sistema tem como objetivo validar senhas digitadas por usuários, garantindo que elas atendam aos requisitos mínimos de segurança. A aplicação verifica se a senha inserida cumpre critérios de comprimento, uso de caracteres maiúsculos, números e caracteres especiais.

2. Escopo

O sistema validadorSenha feito em NodeJS faz uma validação de senhas fornecidas em tempo de execução, retornando mensagens claras sobre a validade ou invalidez da senha, bem como os motivos do não atendimento aos critérios.

3. Requisitos Funcionais

RF01 – Entrada da senha

- O sistema deve permitir que o usuário digite uma senha para ser validada.

RF02 – Instanciação da Classe Senha

- A senha digitada deve ser utilizada para criar uma instância da classe `Senha`.

RF03 – Verificação de requisitos



- O sistema deve verificar os seguintes critérios:
 - RF03.1 – A senha deve conter pelo menos **8 caracteres**.
 - RF03.2 – A senha deve conter pelo menos **uma letra maiúscula**.
 - RF03.3 – A senha deve conter pelo menos **um número**.
 - RF03.4 – A senha deve conter pelo menos **um caractere especial**
(ex: `@`, `#`, `!`, etc.).

RF04 – Instanciação da Classe ValidadorSenha

- A instância de `Senha` deve ser usada como parâmetro para criar uma instância de `ValidadorSenha`.

RF05 – Validação da senha

- O método `validarRequisitos()` da classe `ValidadorSenha` deve verificar se todos os requisitos foram cumpridos.

- Se **todos os critérios forem atendidos**, o sistema deve exibir:
 -  Senha válida
- Se **algum critério não for atendido**, o sistema deve exibir:
 -  Senha inválida
 - Uma mensagem com os erros encontrados

4. Requisitos Não Funcionais

RNF01 – Linguagem de Programação

- O sistema deve ser implementado em uma linguagem orientada a objetos, como JavaScript, Python, Java ou equivalente.

RNF02 – Clareza na interface

- As mensagens de retorno ao usuário devem ser claras e objetivas.
-

Plano de testes

Objetivo:

Garantir que o sistema de validação de senha esteja funcionando corretamente de acordo com os requisitos:

- Pelo menos 8 caracteres
- Pelo menos uma letra maiúscula
- Pelo menos um número
- Pelo menos um caractere especial

Caso de Teste 1 - Senha válida

Entrada: Senha123!

Resultado Esperado: Senha válida.

- | | |
|-----------------------------------|------|
| • Senha tem 8 caracteres | - OK |
| • Contém maiúscula (S) | - OK |
| • Contém número (1, 2, 3) | - OK |
| • Contém caracteres especiais (!) | - OK |

Resultado esperado final: Senha Válida

Mensagem: "✓ Senha válida."

Caso de Teste 2 — Senha sem letra maiuscula

Entrada: senha123!

Resultado Esperado: Senha inválida.

- Senha tem 8 caracteres - OK
- Não contém maiuscula - OK
- Contém número (1, 2, 3) - OK
- Contém caracteres especiais (!) - Erro

Resultado final esperado: Senha inválida.

Mensagem: "✗ Senha deve conter pelo menos uma letra maiuscula."

Caso de Teste 3 — Senha muito curta

Entrada: Sen1!

Resultado Esperado: Senha inválida.

- Senha tem 5 caracteres (menos de 8) - Erro
- Contém maiuscula (S) - OK
- Contém número (1) - OK
- Contém caracteres especiais (!) - OK

Resultado final esperado: Senha inválida.

Mensagem: "✗ Senha muito curta! Deve ter pelo menos 8 caracteres."