Gestão e Qualidade de Software

Apresentação do projeto Strong Password, uma ferramenta de verificação de força de senhas adaptada ao contexto de uso.

Enrico Aguiar Vrunski – 82210618
Thiago Ferreira Lima Gonçalves – 824156179
Matheus Tognon Siqueira - 824141731
Felipe Soares de Oliveira – 824156311
Kayky Cerquiaro Prado - 822155538

Strong Password

INTRODUÇÃO AO PROJETO

Verificador de Força de de Senha

Implementado em Python com foco em boas práticas de segurança.

Adaptação ao Contexto Contexto

Requisitos mínimos baseados no ambiente de uso da senha.

Análise Completa

Avalia letras, números e caracteres especiais com feedback claro.

NIST e OWASP

Benchmark utilizado como base para os requisitos de senha

OBJETIVOS DO PROJETO

Segurança Digital

Proteção contra acessos não autorizados

Público Amplo

Usuários finais e profissionais de TI

Multiplataforma

Leve, portátil e facilmente integrável

A ferramenta oferece uma abordagem automatizada e personalizada para criação de senhas seguras, adaptando-se ao nível de risco do contexto.

MODELO DE PROCESSO: SCRUM

Sprints Semanais

Entregas incrementais com objetivos específicos

Melhoria Contínua

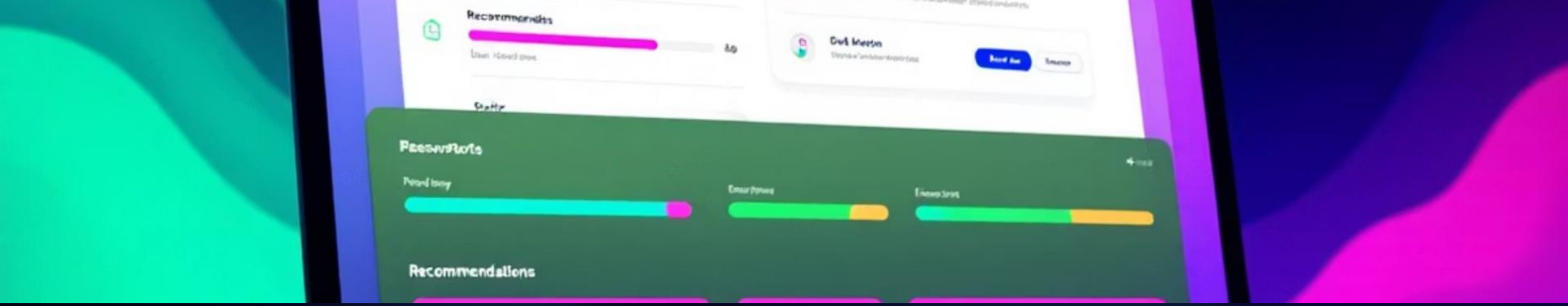
Colaboração entre desenvolvedores e usuários

Feedback Contínuo

Revisões frequentes com usuários

Validação

Testes ao final de cada sprint



REQUISITOS FUNCIONAIS

RF01 Entrada de Senha

Sistema permite ao usuário digitar uma senha para análise.

RF02 Avaliação de Força

Análise baseada em critérios de diversidade de caracteres.

RF03 Pontuação

Atribuição de nota de 0 a 5 conforme segurança da senha.

RF04 Recomendações

Exibição de dicas para melhorar senhas fracas.

REQUISITOS NÃO FUNCIONAIS

RNF01 Desempenho

Tempo de resposta inferior a dois segundos.

RNF02 Tecnologia

Desenvolvimento em linguagem Python.

RNF03 Boas Práticas

Código bem estruturado, comentado e modular.

RNF04 Compatibilidade

Funcionamento em Windows, Linux e MacOS.

SPRINTS

Sprint 1

- (28/04/2025 a 05/05/2025): Planejamento de testes e definição de critérios.

Sprint 2

- (06/05/2025 a 13/05/2025): Implementação dos primeiros testes funcionais.

Sprint 3

- (14/05/2025 a 21/05/2025): Execução de testes de integração e validação de requisitos.

Sprint 4

- (22/05/2025 a 29/05/2025): Ajustes e reexecução de testes conforme feedback.

Sprint 5

- (30/05/2025 a 06/06/2025): Finalização e testes de usabilidade e desempenho.

Sprint 6

- (07/06/2025 a 11/06/2025): Consolidação dos resultados e documentação final.

PLANEJAMENTO DE TESTE

CRONOGRAMA

EAP	Período	Atividades Principais	Entregas Esperadas
EAP 1	Apr 28, 2025 - May 05, 2025	- Planejamento da fase de testes- Definição dos critérios de aceitação e qualidade	Documento de planejamento e critérios de teste
EAP 2	May 06, 2025 - May 13, 2025	- Desenvolvimento dos primeiros casos de teste- Execução dos testes funcionais	Casos de teste desenvolvidos e executados
EAP 3	May 14, 2025 - May 21, 2025	- Execução de testes de integração - Validação dos requisitos	Relatório de testes de integração e validação
EAP 4	May 22, 2025 - May 29, 2025	- Análise de falhas encontradas- Ajustes no sistema e reexecução de testes	Versão ajustada e reavaliada do sistema
EAP 5	May 30, 2025 - Jun 06, 2025	- Testes finais de usabilidade e desempenho	Relatório de usabilidade e desempenho
EAP 6	Jun 07, 2025 - Jun 11, 2025	- Consolidação dos resultados de teste - Elaboração da documentação final	Documentação de testes e aprovação para entrega

REPOSITÓRIO DE GESTÃO DE CONFIGURAÇÃO DE SOFTWARE

GitHub

Criação de repositório no github, armazenando os códigos utilizados nos testes

Pipeline de Integração e Deploy
Deploy Contínuo (CI/CD)

O projeto utiliza o **GitHub Actions** para automatizar o processo de integração e deploy contínuo

PLANO DE TESTE

RECURSOS

Recursos Humanos

Analistas de Teste

Desenvolvedores

Engenheiros de Teste Automatizado

Gerente de Testes

Gerente de Projetos

Recursos Tecnológicos

Linguagem Python

Visual Studio Code (VSCode)

Git & GitHub

Recursos Organizacionais

Metodologias Ágeis

Modelos de qualidade

Documentação

Plano de Testes, Casos de Testes, Relatórios de Defeitos e Métricas, Políticas de Qualidade e Segurança, Tempo e Orçamento Adequados

TESTE DE CAIXA BRANCA

Caixa Branca 1 — Senha menor que o mínimo recomendado

Entrada: senha = "abc", contexto = 1 (mínimo 8)

Caixa Branca 2 — Senha com todos os tipos de caracteres (mínimo 8)

Entrada: senha = "Abc123! ", contexto = 1 (mínimo 8)

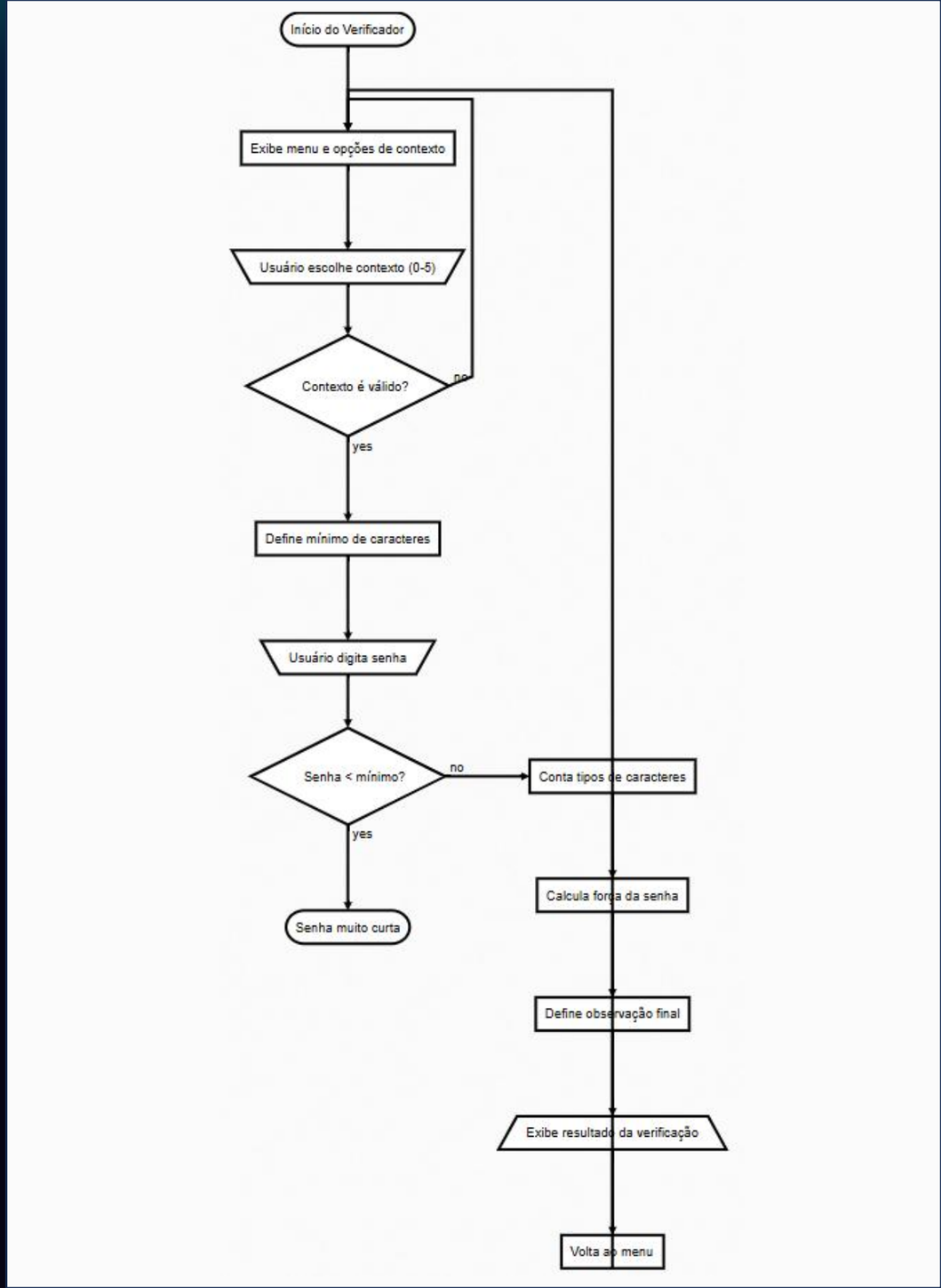
Caixa Branca 3 — Senha com somente números e minúsculas (mínimo 8)

Entrada: senha = "abc12345", contexto = 1 (mínimo 8)

Cálculo da Complexidade Ciclomática V(G) = E - N + 2P

Total de decisões = 1 + 5 + 5 + 5 = 16

Assim, complexidade ciclomática = 16 + 1 = 17



TESTE DE CAIXA PRETA

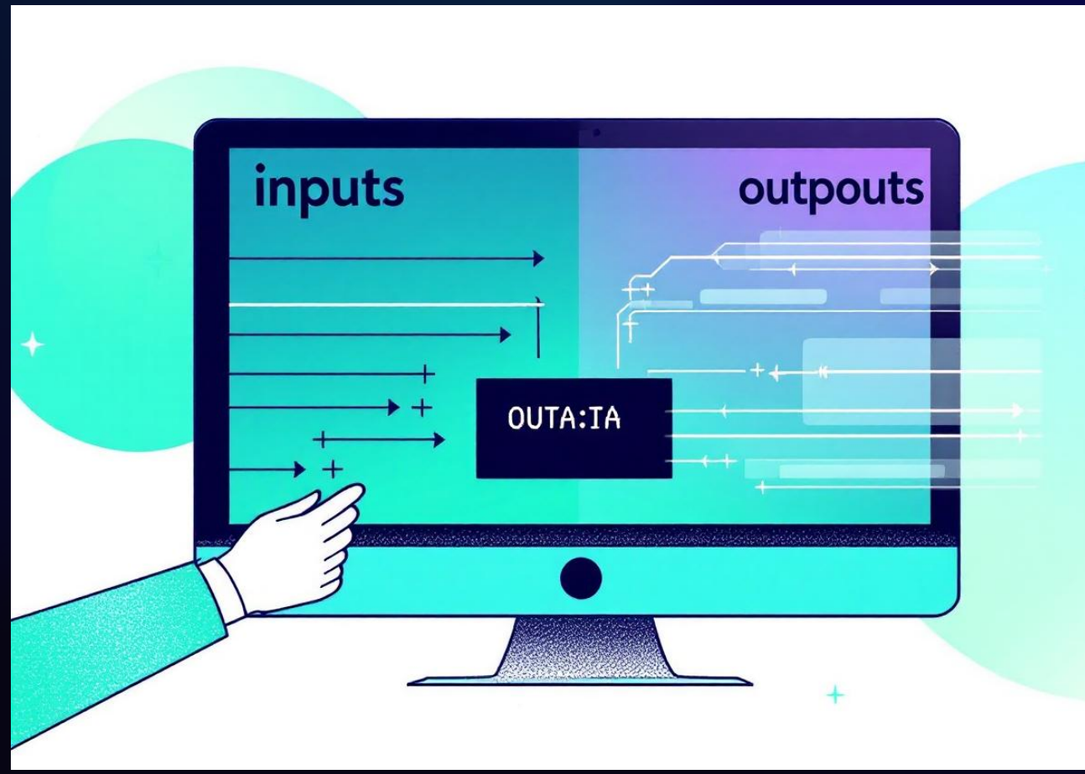
Caixa Preta 1 — Testar opção de contexto inválida

Entrada: Escolher opção 9 no menu

Esperado: Mensagem de opção inválida e novo pedido de escolha.

Roteiro:

- Iniciar programa
- Digitar 9 como escolha de contexto
- Verificar que mensagem “Opção inválida. Tente novamente.” aparece e que o menu é mostrado de novo.



Obrigado!