Trabalho Prático: Versionamento de Código com Git e GitHub

1. O que é um Repositório?

Imagine que você tem um fichário de receitas de família. Cada vez que testa uma receita, anota numa nova página as alterações feitas (mais sal, menos açúcar etc.) e guarda esse fichário cuidadosamente. Um repositório funciona de forma parecida:
- É uma pasta especial onde guardamos todo o código e os arquivos de um projeto.
- Cada vez que você faz alterações e registra um commit, é como se colocasse uma etiqueta na página do fichário, indicando o que mudou e quem fez.
- Assim, você pode voltar no tempo, comparando versões antigas e recuperando estados anteriores do projeto.
- No dia a dia de um desenvolvedor, o repositório é o ponto central de colaboração: todos baixam (clone) essa pasta com histórico completo, trabalham nela, salvam novas versões (push) e compartilham com a equipe.

Por exemplo, vamos imaginar que, no trabalho de semana passada, vocês já conhecessem o GitHub: cada exercício que vocês resolveram seria salvo como um arquivo em um repositório no GitHub — pensem nisso como um 'projeto' de desenvolvimento. Dessa forma, cada solução ficaria organizada em um único lugar, com histórico de todas as alterações.

2. Contexto e Objetivos

O objetivo deste trabalho é consolidar, na prática, os conceitos de versionamento de código com Git e GitHub. No modelo pedagógico do Senac, o foco é a aprendizagem ativa: mais do que ouvir, o aluno faz, explora, testa e reflete enquanto executa atividades reais. Este trabalho segue esse método, desafiando vocês a aplicar Git e GitHub diretamente no desenvolvimento de repositórios colaborativos e na documentação do processo.

3. Organização dos Grupos

- A turma foi dividida em dois grupos aleatórios.
- Cada grupo deve trabalhar colaborativamente em um ou mais repositórios no GitHub. A quantidade de repositórios é de livre escolha aos grupos.
- O uso de IA (ChatGPT, Copilot, etc.) está liberado para apoiar pesquisas, mas todo o conteúdo produzido deve ser entendido e validado pelo grupo. É importante também incluir tanto no trabalho escrito quanto na apresentação uma explicação objetiva sobre como a IA os apoiou.

4. Descrição da Atividade

1. Criar e configurar repositório(s)
- Inicializem um novo repositório no GitHub (público ou privado).
- Clonem-no localmente e façam os primeiros commits de arquivos de exemplo (por ex.: README.md, hello.txt).
- Instalem o GitHub Desktop (https://desktop.github.com) para facilitar commits e sincronização.
- Instalem o VS Code (https://code.visualstudio.com):
  VS Code é uma IDE (Integrated Development Environment), um ambiente que reúne editor de código, terminal e extensões para agilizar o desenvolvimento.

2. Versionar arquivos e histórico de commits
- Modifiquem, adicionem e excluam arquivos em pelo menos 5 commits distintos.
- Demonstrem a recuperação de uma versão antiga (checkout de commit anterior).
- Cada membro deve fazer pelo menos 2 commits individuais.
- Mensagens de commit claras e padronizadas, por exemplo:
  git commit -m "Cria função de soma de dois números"
  git commit -m "Ajusta README com instruções de uso"
  git commit -m "Corrige divisão por zero no cálculo"

Sugestão: Criem um repositório com exercícios anteriores
- No mesmo repositório, adicionem os códigos dos dois exercícios básicos e do desafio prático que fizeram nos últimos dias.
- Organizem cada exercício/desafio em sua própria pasta, contendo pseudocódigo (Portugol).

4. Documentar comandos e boas práticas
- Listem e expliquem os comandos Git mais usados (git init, git add, git commit, git push, git log, git checkout).

5. Material Escrito (2–5 páginas ou Markdown)

- Criem um arquivo TRABALHO.md no repositório para todo o texto.
- O GitHub renderiza automaticamente o Markdown, gerando um documento legível na própria plataforma.
- Cada alteração no .md fica registrada no histórico.

Sintaxe básica de Markdown:
- # Título nível 1
- ## Título nível 2
- - Listas, **negrito**, _itálico_, `código`

No TRABALHO.md, incluam:
- Definições de Git, GitHub e repositório.
- Exemplos de mensagens de commit.
- Capturas de tela do histórico de commits e da organização dos exercícios.

6. Apresentação em PowerPoint

Preparem um slide deck para cobrir:
- Introdução ao versionamento de código
- O que é repositório
- Git x GitHub
- Fluxo de commits e boas práticas de mensagens
- Uso de GitHub Desktop e VS Code
- Organização dos exercícios anteriores no repositório
- Conclusões e aprendizados

7. Entregáveis

- Link(s) do(s) repositório(s) no GitHub, com histórico de commits visível. Pode ser apenas um repositório de um aluno representando o grupo todo, mais de um repositório ou o repositório de cada aluno. No mínimo um.
- Arquivo AULA-VERSIONAMENTO.md no repositório, documentando todo o processo.
- Arquivo PowerPoint (.pptx) para apresentação.

8. Critérios de Avaliação

Critério	Peso
Funcionamento do fluxo Git (commits, push, histórico)	30%
Qualidade do material em Markdown (clareza, formatação)	20%
Organização e clareza do slide deck	20%
Mensagens de commit claras e consistentes	10%
Inclusão e organização dos exercícios anteriores no repositório	10%
Colaboração efetiva (contribuições de todos)	10%

9. Referências e Recursos

Estudo de Git (em Português):
- Livro Pro Git (completo): https://git-scm.com/book/pt-br
- Documentação Git em Português: https://git-scm.com/docs

Estudo de Markdown (em Português):
- Guia básico de Markdown: https://markdown.pro.br/sintaxe
- GitHub Flavored Markdown em Português: https://guides.github.com/features/mastering-markdown/?locale=pt-br

Ferramentas:
- GitHub Desktop: https://desktop.github.com
- VS Code: https://code.visualstudio.com — explore as extensões Git e Markdown Preview