Resumo da aula 16-06-25

Resumo da Aula: Git e GitHub - Prática Colaborativa

A aula foca em conceitos práticos do Git e GitHub para preparar os alunos para o trabalho colaborativo, especialmente com o uso de pull, commit e push, e o entendimento dos estados do repositório.

Tópicos Chave e Falas Importantes:

- **Preparação para o Colaborativo:** A aula focou na sincronização individual, e a colaboração de várias pessoas no mesmo repositório será abordada no futuro.
- **Conceito de Sprint:** Relembra que sprint é o intervalo de tempo para um ciclo de desenvolvimento, que inclui entender o requisito (issue), implementar e verificar. Isso será usado no *hands-on*.
- **Commit - A Fotografia do Repositório:**
  - Um commit é uma alteração, edição ou incremento enviado para o repositório.
  - Pode ser adição, deleção ou inserção de arquivos ou linhas.
  - Cada commit é uma "fotografia" do estado do repositório, registrando um estado diferente.
- **Branches (Ramos) e Seus Estados:**
  - Branches (como feature ou bug) permitem diferentes linhas de desenvolvimento.
  - Cada branch tem um estado diferente de arquivos; nem todas as branches conhecem as mesmas alterações ou arquivos.
  - **Direção do Merge Importa:** Ao fazer merge, é crucial saber a direção (ex: git checkout main e depois git merge bug para trazer as correções do ramo 'bug' para o 'main').
  - Um estado do repositório (commit ou branch) só contém o que foi alterado até aquele ponto, não edições futuras em outros ramos.
- **Comparativo Git vs. Sistemas Antigos (CVS/Subversion):**
  - **Sistemas Antigos (CVS, Subversion - ex: 2008):** Centralizados no servidor.1
    - Requeriam rede para commit (enviar trabalho) e permissão.
    - Não permitiam trabalhar offline com o repositório completo.
  - **Git (Moderno - a partir de 2005):** Distribuído.
    - Sua máquina é um "servidor completo", permitindo trabalhar offline e fazer commits.
    - O envio para o repositório remoto (ex: GitHub) só é necessário com internet.
- **Fluxo de Comandos Essenciais do Git:**
  - **git init**: Para iniciar um repositório.
  - **git checkout [ramo]**: Para mudar de ramo (abrir um repositório em um estado específico).
  - **git add [arquivos]**: Adiciona arquivos à "área estagiada" (staging area), avisando que estão prontos para serem comitados.
  - **git commit -m "mensagem"**: Salva as alterações no seu repositório local (na sua máquina).
    - **Boas Práticas de Commit:**

        - Commits curtos (não grandes alterações).
        - Commits semânticos (com sentido na mensagem).
        - Incluir #issue na mensagem para rastreamento (excelente prática).
    - **git push**: Envia as alterações do seu repositório local para o repositório remoto (ex: GitHub).
        - Pede usuário/senha na primeira vez para sincronizar.
        - Exige que haja um commit anterior; não se faz push sem commit.
    - **git clone [URL]**: Clona um repositório remoto para sua máquina local pela primeira vez.
    - **git pull**: Puxa (baixa) as atualizações do repositório remoto para o seu local.
        - Boa prática: fazer pull toda vez que for começar a trabalhar para ter a versão mais atualizada.
- **Pull Request (PR):**
    - É uma **solicitação de incorporação**.
    - Usado quando não se tem acesso direto (push) ou quando a política do projeto exige revisão formal.
    - Você não faz o push para o projeto principal, mas solicita que alguém da equipe faça o pull do seu trabalho para o projeto.
    - Comum em comunidade de software livre e projetos com formalização de processo (ex: comunidade japonesa).
- **Configuração Inicial do Git Local:**
    - Verificar instalação: git --version no PowerShell ou Bash.
    - Configurar usuário e e-mail (do GitHub) para rastreamento dos commits: git config --global user.name "Seu Nome" e git config --global user.email "seu@email.com".
- **Ambiente de Trabalho:**
    - VSCode é recomendado para ajudar a encontrar erros, embora qualquer editor funcione.

Perguntas Específicas:

1. **Qual a diferença chave entre o Git e sistemas de controle de versão mais antigos como o Subversion, de acordo com a aula?**A diferença chave é que o Git é um sistema **distribuído**, enquanto sistemas mais antigos como o Subversion são **centralizados**. No Git, cada desenvolvedor tem uma cópia completa do repositório em sua própria máquina, o que permite trabalhar offline e fazer commits localmente. Nos sistemas antigos, era necessário estar conectado ao servidor para realizar commits, pois o trabalho era enviado diretamente para um repositório centralizado.
2. **Qual a sequência correta de comandos Git para enviar uma alteração feita localmente para um repositório remoto (como o GitHub), após ter clonado o projeto pela primeira vez?**Após ter clonado o projeto (uma única vez com git clone), a sequência para enviar uma alteração é:
    1. git add [arquivos]: Para adicionar os arquivos modificados à área estagiada.
    2. git commit -m "mensagem": Para registrar a alteração no repositório local.
    3. git push: Para enviar as alterações do repositório local para o repositório remoto.
3. **O que é um "Pull Request" e quando ele é usado?**Um "Pull Request" (PR) é uma **solicitação de incorporação**. Ele é usado quando um desenvolvedor não tem acesso

direto para fazer um push (envio) de suas alterações para o repositório principal do projeto, ou quando a política do projeto exige um processo de revisão formal antes da integração do código. Nesses casos, o desenvolvedor solicita que alguém da equipe (que tem as permissões) puxe (faça o pull) o trabalho que ele desenvolveu para o projeto principal. É uma prática muito comum em comunidades de software livre e projetos com processos formalizados.