Integração e Entrega Contínua – DSM – Professora Lucineide – 13/02/2026

Exercício 1 — Criar o repositório do projeto e adicionar colaboradores

Objetivo: criar um repositório privado no GitHub chamado inpe-alertas, com todos os membros do grupo como colaboradores, garantindo que o desenvolvimento seja colaborativo e seguro.

1- Acessar o GitHub

- **Passo:** Entre em https://github.com com sua conta. Se não tiver, crie uma gratuitamente.
- **Explicação:** O GitHub será o "local central" onde todo o código do projeto vai ficar. Usar uma conta própria garante que você possa gerenciar seus repositórios e manter histórico de contribuições.

2- Criar novo repositório

- **Passo:** Clique no botão **New** no canto superior esquerdo.
- **Explicação:** É aqui que vamos criar o espaço na nuvem para armazenar o código do projeto. Cada grupo terá seu repositório exclusivo para organizar e versionar o trabalho.

3- Nome do repositório

- **Passo:** No campo **Repository name**, digite:
- inpe-alertas
- **Explicação:** Escolher um nome claro e direto ajuda a identificar o projeto. Aqui usamos inpe-alertas para refletir a proposta do app: alertas climáticos e ambientais para o INPE.

4- Definir a visibilidade

- **Passo:** Em **Visibility**, marque **Private**.
- **Explicação:** Um repositório privado protege o código contra acesso público. Isso é essencial para evitar que pessoas fora da equipe vejam ou copiem o trabalho antes da entrega, e também é prática comum em projetos reais até que estejam prontos para publicação.

5- Criar README inicial

- **Passo:** Marque a opção **Add a README file**.
- **Explicação:** O README é a "capa" do repositório, onde explicamos do que se trata o projeto. Criar esse arquivo junto com o repositório evita a necessidade de um git push inicial vazio e já permite que todos no time tenham um ponto de partida para entender o objetivo do projeto.

6- Criar o repositório

- **Passo:** Clique em **Create repository**.
- **Explicação:** Este passo efetivamente cria o espaço online no GitHub com as configurações iniciais definidas. A partir daqui, já é possível clonar o repositório localmente e começar a desenvolver.

7- Adicionar colaboradores

- **Passo:** Vá até **Settings → Collaborators → Manage access** → **Add people** → digite o nome de usuário GitHub dos colegas → **Add**.
- **Explicação:** Adicionar colaboradores garante que todos no grupo possam enviar código, abrir PRs e contribuir de forma controlada. Se um colega não for adicionado, ele não conseguirá fazer alterações no repositório.

8- Aceitar convites

- **Passo:** Cada colaborador precisa aceitar o convite — notificação no e-mail e no próprio GitHub.
- **Explicação:** Enquanto o convite não for aceito, o GitHub não reconhece o colaborador como parte do projeto, bloqueando push e PRs.

Verificação final

- O link do repositório está acessível para todos os membros do grupo.
- O repositório está **Private**.
- O arquivo **README.md** existe e pode ser editado.
- Todos os colaboradores aparecem na lista de acesso.

Integração e Entrega Contínua – DSM – Professora Lucineide – 13/02/2026

Exercício 2 – Configuração inicial: README, branch dev e proteção da main

Objetivo: Criar a branch de desenvolvimento (dev), proteger a branch principal (main) e configurar um README inicial para guiar o time.

1- Editar o README

- **Passo:** Abra o arquivo README.md no GitHub e clique em **Edit**. Adicione, por exemplo:
- *# Aplicativo INPE – Monitoramento de Eventos Climáticos*
- *Objetivo: app móvel para alertas de queimadas, inundações, desmatamento e relatos da população em tempo real.*
- **Explicação:** O README é a apresentação oficial do repositório. Ele ajuda qualquer pessoa (e o próprio time) a entender rapidamente o objetivo e escopo do projeto. É boa prática atualizá-lo conforme o projeto evolui.
- Clique em **Commit change** para efetuar as alterações no arquivo**.**

2- Criar a branch dev

- **Passo (via terminal):**

git clone https://github.com/<seu-usuario>/inpe-alertas.git

cd inpe-alertas

git checkout -b dev

git push -u origin dev

(Ou via GitHub UI: ***Code → Branch: main → Switch branches/tags → Find or create a branch → dev → Create branch from main****.)*

- **Explicação:** A branch dev serve como ambiente de integração das funcionalidades. É nela que as mudanças são testadas antes de serem mescladas à main. Isso protege a branch principal de receber código instável.

3- Proteger a branch main

- **Passo:** No GitHub → **Settings → Branches → Add rule** →

- **Branch name pattern:** *main*
- Ativar:
  - *Require a pull request before merging*
  - *Require approvals*
  - (Opcional) *Include administrators* → **Save changes**.

- **Explicação:** Proteger a main evita que alguém faça alterações diretas sem revisão. É uma prática de segurança que garante qualidade e estabilidade no código final.

Verificação final

- README.md contém a descrição do projeto.
- Branch dev aparece na lista de branches no GitHub.
- Branch main está protegida.

Exercício 3 – Primeiro commit em dev e Pull Request para main

Objetivo: Criar o primeiro arquivo do app, trabalhar na dev e abrir um Pull Request (PR) para main.

Passo a passo comentado

1- **Estando no repositório, clique em Add file**

2- **Create new file**

3- **No campo de nome do arquivo, digite:** *index.js*

4- **No editor que aparece, escreva:**

console.log("Aplicativo de Monitoramento de Eventos Climáticos - INPE");

5- **Em Commit changes...**

a. Escreva a mensagem: *feat: adiciona entrada do app.*

b. Selecione a opção **Commit directly to the dev branch**.

6- Clique em **Commit changes**.

Explicação: Esse método é útil se você ainda não clonou o repositório localmente ou está começando agora com Git. O arquivo é criado diretamente na branch dev no GitHub.

2- Abrir o PR

- **Passo:** No GitHub → **Pull requests → New pull request** →
    - **Base:** main | **Compare:** dev
    - Adicionar título e descrição → **Create pull request**.
- **Explicação:** O PR permite que outros revisem seu código antes que ele seja integrado à branch principal. Isso é essencial para evitar erros e manter a qualidade.

Alerta: Antes de criar um PR, **sempre valide se a branch de origem tem algo diferente da branch de destino**:

- Use git log no terminal para ver os commits.
- Ou, no GitHub, compare as branches usando **Compare** antes de abrir o PR.

3- Aguardar revisão

- **Passo:** Solicitar que um colega revise e aprove o PR.
- **Explicação:** O *code review* é um momento de aprendizado coletivo e de prevenção de erros. Mesmo desenvolvedores experientes cometem deslizes — a revisão reduz riscos.

Verificação final

- PR aparece listado no GitHub.
- Branch main não foi alterada até o merge ser aprovado.

Exercício 4 – Ativar GitHub Actions e criar workflow inicial

Objetivo: Configurar um pipeline simples para rodar a cada alteração na dev.

1- Acessar Actions

- **Passo:** No repositório → **Actions** → Escolher **Node.js** ou **set up a workflow yourself**.

- **Explicação:** A aba Actions é onde criamos *workflows* — sequências de tarefas automatizadas que o GitHub executa sempre que um evento é disparado.

2- Criar o arquivo do workflow

- **Passo:** Criar .github/workflows/ci.yml com:

```
name: CI Simples

on:
  push:
    branches: [ dev ]
  pull_request:
    branches: [ dev ]

jobs:
  build:
    runs-on: ubuntu-latest
    steps:
      - name: Checkout do repositório
        uses: actions/checkout@v4
      - name: Configurar Node.js
        uses: actions/setup-node@v4
        with:
          node-version: 20
      - name: Teste de funcionamento
        run: echo "CI funcionando!"
```

- **Explicação:** Este arquivo diz ao GitHub Actions para rodar um job de teste simples sempre que houver commit ou PR na branch dev. É o primeiro passo para automatizar verificações.

3- Testar o workflow

- **Passo:** Fazer uma alteração qualquer no README.md e commitar na dev.
- **Explicação:** O objetivo é disparar o workflow para confirmar que está configurado corretamente.

Verificação final

- Workflow aparece listado em **Actions**.
- Log exibe a mensagem **"CI funcionando!"**.

Exercício 5 – Reflexão

Pergunta: *Como o CI/CD ajuda a reduzir erros no app do INPE?*

Resposta esperada:

- Detecta problemas cedo (antes de chegar à produção).
- Padroniza o processo de entrega.
- Automatiza testes e verificações.
- Garante rastreabilidade e histórico.
