Growth Equestre (Hackathon) — Guia de Início (Windows + VS Code)

Manual passo a passo para instalar o VS Code, configurar Git/GitHub, clonar o repositório e começar a trabalhar com segurança (branches por equipe).

Repositório: https://github.com/brodyandre/growth_equestre_hackathon_2026
Data: 10/02/2026

Para quem é: integrantes do time sem familiaridade com Git/GitHub, usando Windows e VS Code.

Checklist rápido (5 minutos)

Se você já tem tudo instalado, siga só esta lista:

```
1) Instale VS Code e Git (se necessário)
2) Faça login no GitHub pelo VS Code
3) Clone o repositório
4) Troque para a branch da sua equipe (feature/ds, feature/be, feature/fe)
5) Crie um commit e faça push para sua branch
```

1) Instalar o que é necessário

Você precisa de VS Code (editor) e Git (controle de versão). Também é recomendado instalar a extensão do GitHub no VS Code para facilitar o login.

1.1 Baixar e instalar o VS Code

No Windows, baixe e instale o Visual Studio Code. Na instalação, marque as opções para adicionar o VS Code ao PATH e habilitar “Open with Code” (isso facilita abrir a pasta do projeto com botão direito).
Onde baixar: pesquise por “Visual Studio Code download” e escolha a versão para Windows.

1.2 Baixar e instalar o Git

Instale o Git for Windows. Ele fornece o comando git (obrigatório para clonar e enviar alterações). Mantenha as opções padrão, principalmente: “Git from the command line and also from 3rd-party software”.
Teste no PowerShell:

```
git --version
```

2) Conectar o VS Code ao GitHub

Você já foi adicionado como colaborador no repositório. Agora falta autenticar no seu computador para poder clonar e fazer push.

2.1 Login pelo VS Code (recomendado)

No VS Code, abra a paleta de comandos (Ctrl+Shift+P) e procure por “GitHub: Sign in”. O VS Code abrirá o navegador para você autorizar o acesso.

Depois do login, o VS Code memoriza a credencial e você não precisa digitar senha em cada push.

2.2 Configurar nome e e-mail no Git (uma vez)

No terminal:

```
git config --global user.name "Seu Nome"
git config --global user.email "seuemail@exemplo.com"
```

3) Clonar o repositório na sua máquina

Clonar significa baixar o projeto do GitHub para o seu computador (com histórico).

3.1 Clonar via VS Code (mais fácil)

No VS Code: Ctrl+Shift+P → Git: Clone → cole a URL:
`https://github.com/brodyandre/growth_equestre_hackathon_2026`
Escolha uma pasta (ex.: Documents\Repositorios) e depois clique em Open.

3.2 Clonar via terminal (alternativa)

No PowerShell/terminal do VS Code:

```
cd $env:USERPROFILE\Documents\Repositorios
git clone https://github.com/brodyandre/growth_equestre_hackathon_2026
cd growth_equestre_hackathon_2026
```

4) Trabalhar na branch correta (por equipe)

Equipe	Branch	O que vai aqui (exemplos)
Data Science	feature/ds	Modelos/heurísticas, pipelines, scripts de dados, notebooks.
Backend	feature/be	APIs, rotas, serviços, banco, integrações.
Frontend	feature/fe	UI Streamlit, componentes, CSS, UX, páginas do painel.

4.1 Atualize referências remotas (sempre antes de começar)

```
git fetch origin
```

4.2 Troque para a branch da sua equipe

Execute o bloco da sua equipe. Se a branch ainda não existir localmente, o comando cria e rastreia a remota.

```
# Data Science
git switch feature/ds || git switch -c feature/ds --track origin/feature/ds

# Backend
git switch feature/be || git switch -c feature/be --track origin/feature/be

# Frontend
git switch feature/fe || git switch -c feature/fe --track origin/feature/fe
```

Confirme a branch atual:

```
git branch --show-current
```

Regra de ouro: não subir nada na main. Use branch + Pull Request (PR).

5) Fluxo de trabalho (editar → commit → push → PR)

5.1 Commit pelo terminal (simples e direto)

```
git status
git add .
git commit -m "feat: descricao curta da entrega"
```

5.2 Push (enviar suas alterações)

```
# Data Science
git push origin feature/ds

# Backend
git push origin feature/be

# Frontend
git push origin feature/fe
```

5.3 Pull Request (PR) — o que é

PR (Pull Request) é um “pedido de revisão” para unir sua branch ao ramo principal do projeto. Ele permite que o time revise antes de integrar, evitando erros na main.

6) Rodar o projeto localmente (começo rápido)

A forma de rodar depende do repositório (Docker Compose ou comandos diretos). Se existir docker-compose.yml, a opção abaixo costuma ser a mais fácil.

6.1 Opção comum: Docker Compose

```
docker compose pull
docker compose up -d
docker compose ps
```

Checar saúde do backend (exemplo):

```
curl http://localhost:3000/health
```

Se o projeto não usa Docker, siga o README do repositório (ou peça ao time para criar/atualizar um README com os comandos).

7) Problemas comuns

Problema	Solução rápida
“git não é reconhecido”	Reinstale o Git for Windows e feche/reabra o terminal.
Push pede senha / falha autenticação	Faça login no VS Code com “GitHub: Sign in”.
Branch da equipe não existe	Confirme no GitHub. Se não existir, peça para criar no repositório.
Conflito de merge	Não force. Avise no PR e peça ajuda para resolver.
Docker não inicia	Verifique se Docker Desktop está aberto e se o WSL2 está habilitado.

Regras do time (resumo): não commitar/push na main; sempre conferir a branch antes de commitar; rodar git fetch origin no início do dia; abrir PR para integrar.