

GUIA COMPLETO – ORGANIZAÇÃO DO PROJETO NO GITHUB (MEU CONDOMÍNIO)

1. Estrutura Básica Solicitada pelo Professor

O professor Arthur destacou que cada tarefa (issue) deve ser organizada de forma completa, garantindo rastreabilidade e clareza no desenvolvimento. Cada issue precisa conter os seguintes campos obrigatórios:

1. **Assignees:** responsável pela execução da tarefa.
2. **Type:** tipo da atividade (ex: correção, nova página, melhoria, documentação).
3. **Status:** estágio da tarefa (To Do, In Progress, Done).
4. **Milestone:** marco ou entrega à qual a tarefa pertence.
5. **Branch associada:** nome da branch criada no VSCode para essa tarefa.

Esses elementos garantem que todas as etapas do projeto estejam documentadas, vinculadas a entregas específicas e devidamente atribuídas aos integrantes da equipe.

2. Criando uma Nova Issue (Formato Detalhado)

Cada issue deve ser tratada como um registro técnico completo, descrevendo de forma clara o que será feito, por que e como. Use sempre o modelo abaixo:

Passo 1 – Criação da Issue:

1. Acesse a aba **Issues** do repositório.
2. Clique em **New Issue**.
3. No campo **Title**, escreva um título objetivo e informativo (ex: 'Correções de Segurança — Dependabot').

Passo 2 – Preenchendo a Descrição Detalhada:

Use o seguinte modelo de descrição dentro da issue.

Este exemplo está resumido apenas para referência. Na prática, nós devemos redigir cada campo com explicações completas e técnicas.

1. **Descrição da Tarefa:** Atualizar dependências do projeto conforme alertas do Dependabot. Garantir que todos os módulos (backendpy, backend e Marketing Screens Package) estejam seguros e compatíveis.
2. **Objetivo:** Eliminar vulnerabilidades detectadas, assegurando estabilidade e segurança em todas as partes do sistema.
3. **Escopo da Tarefa:** Verificar pacotes vulneráveis, atualizar dependências, testar rebuild dos containers, registrar prints de antes e depois e validar execução sem erros.
4. **Evidências:** Prints e logs demonstrando que os alertas foram resolvidos e o sistema executa corretamente após o rebuild.
5. **Critérios de Aceitação:** Todos os alertas eliminados, sistema executando sem falhas, Pull Request aprovado e vinculado à issue.
6. **Observação:** Este é apenas um exemplo simplificado. Na tarefa real, cada etapa deve conter explicações completas, prints e justificativas técnicas.

Passo 3 – Preenchendo os Campos Obrigatórios da Issue

Depois de preencher a descrição, complete os campos no lado direito da tela:

1. **Assignees** – quem vai executar a tarefa.
2. **Labels** – tipo de atividade (ex: Segurança, Backend, Frontend).
3. **Status** – escolha To Do, In Progress ou Done.
4. **Milestone** – selecione a entrega correspondente (ex: Entrega 2 – Segurança e Dependabot).

PASSO 4 – Criando e vinculando a branch

Após criar a issue e antes de iniciar qualquer desenvolvimento, **é fundamental garantir que sua branch main local esteja atualizada** com o repositório remoto.

Isso evita conflitos de código e garante que você comece sua tarefa com a versão mais recente do projeto.

4.1. Atualizar a main local (passo obrigatório antes de criar a branch):

<u>git status</u>	# Verifique em qual branch está
<u>git fetch --all</u>	# Atualiza as referências remotas
<u>git checkout main</u>	# Vai para a branch main local
<u>git pull origin main</u>	# Puxa as atualizações mais recentes do repositório remoto

4.2. Criar uma nova branch específica para a tarefa:

Crie sempre a branch a partir da main atualizada, com um nome curto e descritivo. Evite espaços, acentos ou caracteres especiais.

Exemplo:

git checkout -b feature/pagina-faq

ou

git checkout -b security/dependabot-fixes

O “-b” antes do nome da branch é utilizado para **criar** a nova branch

Cada issue deve ter sua branch própria.

Jamais desenvolva ou faça alterações diretamente na main.

4.3. Após desenvolver e finalizar o que foi proposto, commit e enviar sua branch para o repositório remoto:

<u>git add .</u>	#Adiciona tudo que vc alterou/adicionou	
<u>git commit -m "Mensagem clara descrevendo o que foi feito"</u>		#Commit com o que foi feito
<u>git push -u origin nome-da-branch</u>	#Subindo para sua branch	

4.4. Vincular a branch à issue:

No GitHub, vá até a issue e:

- Localize o campo “Development” no lado direito.
- Clique em **“Link a branch or pull request”** e selecione a sua branch. (INDICADO)
- Ou, ao criar o Pull Request, use palavras-chave como:
- Closes #123
- (trocando o número pelo ID da issue). Assim, a issue será fechada automaticamente após o merge.

4.5. Garantir que sua branch está sincronizada com a main antes de abrir o Pull Request:

Antes de abrir ou atualizar o PR, traga as últimas mudanças da main para sua branch:

```
git fetch --all           # Atualiza referências do remoto  
git checkout main       # Vai para a 'main' local  
git pull origin main    # Garante que a 'main' local está atualizada  
git checkout nome-da-branch # Troca para a sua branch de trabalho (já existente). Se não  
existir, veja o bloco abaixo  
git merge origin/main # Mescla na sua branch as mudanças recentes da 'main' remota (mantém  
histórico claro)  
git push               # Envia sua branch atualizada para o remoto
```

Se a sua branch AINDA NÃO EXISTE, crie-a a partir da main atualizada:

```
git fetch --all           # Atualiza referências do remoto  
git checkout main       # Vai para a 'main' local  
git pull origin main    # Garante que a 'main' local está atualizada  
git checkout -b nome-da-branch # Cria e já troca para a nova branch a partir da 'main'  
atualizada  
git push -u origin nome-da-branch # Sobe a primeira vez e cria o tracking (-u) entre local e  
remoto
```

Se houver conflitos, resolva-os antes de enviar.

4.6. Abrir o Pull Request (PR):

No GitHub, abra um Pull Request da sua branch para a main, descrevendo claramente:

O que foi feito:
Prints de evidência (se aplicável):
E o número da issue vinculada.
Peça revisão do professor ou de colegas antes do merge.

Regras Importantes:

Nunca execute **git push origin main** diretamente.
Todas as alterações devem ser feitas em **branches específicas**.
A integração com a main só deve **acontecer via Pull Request revisado e com orientação ou
autorização do professor.**
Antes de iniciar qualquer nova tarefa, **sempre sincronize sua main local** com o repositório remoto.

Passo 5 – Acompanhamento e Fechamento da Issue

Durante o desenvolvimento:

1. Comente na issue informando o andamento da tarefa.
2. Adicione evidências (prints, logs, links de PR).
3. Ao concluir, escreva um resumo final do que foi feito.
4. Feche a issue e confirme que está vinculada ao milestone correto.

Dica Final

Evite descrições curtas ou genéricas. Cada issue deve funcionar como um pequeno relatório técnico, mostrando claramente o problema, as ações executadas, as evidências e o resultado final. O exemplo apresentado acima é apenas uma versão reduzida; nas tarefas reais, é obrigatório detalhar cada item com clareza.

Guia elaborado por: Keven Costa Negreiros
Projeto Integrador II — Sistema “Meu Condomínio”

