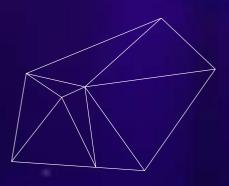


# Unidade 3

## Capítulo 3



Depuração e Versionamento





















### **Enunciado**

Para assegurar a assimilação dos conceitos iniciais de versionamento e do uso do Git e GitHub, propõe-se a criação de um repositório colaborativo. O objetivo principal é desenvolver habilidades práticas nessas ferramentas, com ênfase no trabalho em equipe e na colaboração. A atividade consistirá na configuração de um projeto simples, permitindo a simulação de um fluxo de trabalho real com versionamento.

Segue aqui uma *sugestão* de roteiro para solução do problema.

#### Título: Programa Conversor de Unidades em Linguagem C.

#### Formação das Equipe (máximo de 10 integrantes)

A equipe pode ser dividida em 1 Líder (gerencia repositório) e 9 desenvolvedores (cada um com tarefa específica).

#### Exemplo de Tarefas que o grupo poderá desenvolver.

Observação: Não é obrigatório implementar todas as conversões sugeridas.

- 1. Unidades de comprimento (metro, centímetro, milímetro)
- 2. Unidades de massa (quilograma, grama, tonelada)
- 3. Unidades de volume (litro, mililitro, metros cúbicos)
- 4. Unidades de temperatura (Celsius, Fahrenheit, Kelvin)
- 5. Unidades de velocidade (km/h, m/s, mph)
- 6. Watts (W), quilowatts (kW), cavalos-vapor (cv ou hp)
- 7. Unidades de área (metro quadrado, centímetro quadrado)
- 8. Unidades de tempo (segundos, minutos, horas)
- 9. Bits, bytes, kilobytes (KB), megabytes (MB), gigabytes (GB), terabytes (TB)
- 10. Interface de usuário
- 11. Testes e depuração

#### Destacamos aqui alguns passos necessários para este projeto.

- Líder cria repositório no GitHub.
- Cada desenvolvedor cria conta no GitHub.
- No git do VSCode configurar (nome, e-mail, senha, etc).

#### Na etapa de desenvolvimento deve-se:

- Cada desenvolvedor criar branch (ex: feature/comprimento).
- Desenvolvedores trabalham em suas tarefas.
- Fazer *commits* regulares com mensagens descritivas.
- Utilizar issues e labels para acompanhar progresso.
- Desenvolvedores criam *pull* requests.
- Líder revisa código e aprova pull requests.
- Resolver conflitos.
- Integrar mudanças.
- Implementar testes básicos para garantir que cada recurso esteja funcionamento corretamente.
- Adicionar um arquivo **README.md** com instruções de uso do programa.

Enfim, integrar todas as partes em um programa funcional, revisando e corrigindo conflitos de merge, se necessário. Ao final, todos executam o programa para validar o resultado.

Dicas: Utilize comentários no código; Mantenha o código organizado; Teste regularmente e Comunique-se com a equipe.

Lista com alguns comandos Git utéis para este projeto:

- 1. git init Inicializar repositório.
- 2. git add . Adicionar arquivos.
- 3. git commit -m "mensagem" Fazer commit.
- 4. git branch nome Criar branch.
- 5. git checkout nome Trocar de branch.
- 6. git merge nome Integrar mudanças.
- 7. git push origin nome Enviar mudanças para GitHub.
- 8. git pull origin nome Atualizar repositório local.

