Aqui está uma explicação detalhada de cada comando Git que você mencionou:

**1. git init**

**Descrição:**

* O comando git init inicializa um novo repositório Git em um diretório existente. Ele cria um subdiretório .git, onde o Git armazenará os metadados do repositório e as informações sobre o histórico de versões.

**Uso:**

git init

* Após executar esse comando, o diretório atual se torna um repositório Git, pronto para o rastreamento de versões.

**2. git status**

**Descrição:**

* O comando git status exibe o estado atual do repositório. Ele mostra as alterações que foram feitas, quais arquivos estão em estado de staged (preparados para commit), quais arquivos não estão sendo rastreados e quais alterações ainda não foram adicionadas ao staged.

**Uso:**

git status

* Esse comando é útil para verificar o que foi modificado desde o último commit e o que ainda precisa ser adicionado ou comitado.

**3. git add .**

**Descrição:**

* O comando git add . adiciona todas as alterações (arquivos novos, modificados ou deletados) no diretório atual e seus subdiretórios ao staged. Isso prepara os arquivos para o próximo commit.

**Uso:**

git add .

* O ponto (.) indica que todas as alterações no diretório atual devem ser adicionadas ao staged. Você pode usar outros caminhos ou nomes de arquivos para adicionar especificamente arquivos ou diretórios.

**4. git commit -a -m "Primeiro commit"**

**Descrição:**

* O comando git commit captura uma "foto" do estado atual do projeto (ou seja, os arquivos que foram adicionados ao staged). Com as opções -a e -m, o comando combina várias etapas em uma só.

**Opções:**

* -a: Adiciona automaticamente todos os arquivos que foram modificados (e que já estão sendo rastreados pelo Git) ao staged antes de fazer o commit.
* -m "Primeiro commit": Permite que você adicione uma mensagem de commit diretamente na linha de comando.

**Uso:**

git commit -a -m "Primeiro commit"

* Esse comando faz o commit de todas as alterações rastreadas e adiciona a mensagem "Primeiro commit" ao histórico.

**5. git config --global user.email "fernando.rodrigues2024@outlook.com"**

**Descrição:**

* O comando git config --global define opções de configuração para o Git. Neste caso, está configurando o endereço de e-mail que será associado aos commits que você faz em qualquer repositório no sistema.

**Uso:**

git config --global user.email "fernando.rodrigues2024@outlook.com"

* Esse comando define o e-mail "fernando.rodrigues2024@outlook.com" como o e-mail global do usuário, que será usado para identificar quem fez os commits.

**6. git config --global user.name "Fernando"**

**Descrição:**

* Similar ao comando anterior, este comando configura o nome do usuário que será associado aos commits em qualquer repositório no sistema.

**Uso:**

git config --global user.name "Fernando"

* Esse comando define "Fernando" como o nome global do usuário, que será associado aos commits.

**7. git push origin master**

**Descrição:**

* O comando git push é usado para enviar os commits locais para um repositório remoto. Neste caso, está enviando os commits para o branch master no repositório remoto chamado origin.

**Opções:**

* origin: O nome padrão para o repositório remoto (geralmente o repositório original de onde o projeto foi clonado ou o repositório remoto ao qual você deseja enviar suas alterações).
* master: O nome do branch principal (pode variar dependendo da configuração do repositório, já que em muitos casos o branch principal é chamado de main).

**Uso:**

git push origin master

* Esse comando envia as alterações locais (commits) para o branch master no repositório remoto origin.

Esses comandos cobrem as etapas principais de configuração e operação básica com o Git: inicializar um repositório, rastrear mudanças, fazer commits, configurar informações do usuário e enviar mudanças para um repositório remoto.