

# Lab Exercise 3- Working with Git Rebase

**Name:** Vishal Pandey

**Sap ID:** 500125280

**Roll no.:** R2142231906

**Batch - 2**

## Lab Exercise: Git Rebase

This exercise demonstrates the use of git rebase in a scenario where no conflicts occur.

---

### Objective

1. Learn how to rebase branches when there are no conflicting changes.
  2. Understand the clean, linear history created by git rebase.
- 

### Prerequisites

1. Install Git on your system.
2. Initialize a Git repository:

```
PS C:\Users\ASUS\OneDrive\Desktop\DevSecOps> git init git-rebase-lab  
  
Initialized empty Git repository in C:/Users/ASUS/OneDrive/Desktop/DevSecOps/git-rebase-lab/.git/  
PS C:\Users\ASUS\OneDrive\Desktop\DevSecOps> cd git-rebase-lab
```

---

## Step-by-Step Instructions

### 1. Set Up the Repository

1. Create the main branch and make the initial commit:

```
echo "Line 1 from main branch" > file.txt
```

```
git add file.txt
```

```
git commit -m "Initial commit: Add Line 1 from main branch"
```

```
PS C:\Users\ASUS\OneDrive\Desktop\DevSecOps\git-rebase-lab> echo "Line 1 from master branch" > file.txt
PS C:\Users\ASUS\OneDrive\Desktop\DevSecOps\git-rebase-lab> git add file.txt
PS C:\Users\ASUS\OneDrive\Desktop\DevSecOps\git-rebase-lab> git commit -m "Initial commit: Add Line 1 from master branch"
[master (root-commit) 58ec874] Initial commit: Add Line 1 from master branch
1 file changed, 0 ins: ASUS\ns(+), 0 deletions(-)
create mode 100644 file.txt
```

2. Create a new branch feature-branch:

```
git checkout -b feature-branch
```

3. Add a new line to file.txt in feature-branch:

```
echo "Line 2 from feature branch" >> file.txt
```

```
git add file.txt
```

```
git commit -m "Add Line 2 from feature branch"
```

```
PS C:\Users\ASUS\OneDrive\Desktop\DevSecOps\git-rebase-lab> git checkout -b feature-branch
Switched to a new branch 'feature-branch'
PS C:\Users\ASUS\OneDrive\Desktop\DevSecOps\git-rebase-lab> echo "Line 2 from feature branch" >> file.txt
PS C:\Users\ASUS\OneDrive\Desktop\DevSecOps\git-rebase-lab>
```

4. Switch back to the main branch and add another line:

```
git checkout main

echo "Line 3 from main branch" >> file.txt

git add file.txt

git commit -m "Add Line 3 from main branch"
```

```
PS C:\Users\ASUS\OneDrive\Desktop\DevSecOps\git-rebase-lab> echo "Line 3 from master branch" >> file.txt
PS C:\Users\ASUS\OneDrive\Desktop\DevSecOps\git-rebase-lab> git add file.txt
PS C:\Users\ASUS\OneDrive\Desktop\DevSecOps\git-rebase-lab> git commit -m "Add Line 3 from master branch"
[master d74d596] Add Line 3 from master branch
1 file changed, 0 insertions(+), 0 deletions(-)
```

---

## 2. Rebase feature-branch onto main

1. Switch to feature-branch:

```
git checkout feature-branch
```

2. Rebase feature-branch onto main:

```
git rebase main
```

3. Git will replay the commit from feature-branch onto the main branch. Since there are no conflicts, the rebase completes automatically.
-

### 3. Verify the Rebase

1. View the commit history:

```
git log --oneline
```

```
git log --oneline --graph
```

2. Check the contents of file.txt:

```
cat file.txt
```

Output:

```
HEAD is now at ce5b873 Add Line 2 from feature branch
PS C:\Users\ASUS\OneDrive\Desktop\DevSecOps\git-rebase-lab> git log --oneline
ce5b873 (HEAD -> master, feature-branch) Add Line 2 from feature branch
d74d596 Add Line 3 from master branch
58ec874 Initial commit: Add Line 1 from master branch
PS C:\Users\ASUS\OneDrive\Desktop\DevSecOps\git-rebase-lab> git log --oneline --graph
* ce5b873 (HEAD -> master, feature-branch) Add Line 2 from feature branch
* d74d596 Add Line 3 from master branch
* 58ec874 Initial commit: Add Line 1 from master branch
PS C:\Users\ASUS\OneDrive\Desktop\DevSecOps\git-rebase-lab> cat file.txt
Line 1 from master branch
Line 2 from feature branch
Line 3 from master branch
PS C:\Users\ASUS\OneDrive\Desktop\DevSecOps\git-rebase-lab>
```