

## **Prerequisites:**

- **Git** installed.
- **Visual Studio Code (VS Code)** installed.
- A terminal/command prompt.

## **Part 1: The Setup (Creating the Paradox)**

**Scenario:** We are writing a story file. You will act as two different authors ("Alice" and "Bob") trying to write the ending of the story at the same time.

### **Step 1: Initialize the Project**

1. Create a folder named conflict-lab.
2. Open it in VS Code.
3. Open the Terminal in VS Code (Ctrl + ~ or Terminal > New Terminal).
4. Initialize the repository:  
Bash  
`git init`

### **Step 2: Create the Baseline**

1. Create a file named `story.txt`.
2. Add the following text:  
Plaintext  
Once upon a time, there was a brave knight.  
The knight traveled to the dark forest.

3. Commit this baseline:

Bash

```
git add story.txt
```

```
git commit -m "Chapter 1: The beginning"
```

## **Part 2: The Divergence (Parallel Development)**

Step 3: Branch A - The "Happy Ending"

Now you are Author A.

1. Create a new branch:

Bash

```
git checkout -b happy-ending
```

2. Modify story.txt by adding this line to the end:

Plaintext

The knight found a chest of gold and lived happily ever after.

3. Save and commit:

Bash

```
git add story.txt
```

```
git commit -m "Wrote the happy ending"
```

Step 4: Branch B - The "Tragic Ending" (The Conflict)

Now you switch back to the main timeline and become Author B.

1. Switch back to main:

Bash

```
git checkout main
```

*(Notice the "happy ending" text disappears. You are back at the start.)*

The screenshot shows a terminal window with a code editor integrated at the top. The code editor has tabs for 'app.py' and 'story.txt'. The 'story.txt' tab contains the following text:

```
C: > Users > PC > conflict-lab > story.txt
1 Once upon a time, there was a brave knight.
2 The knight traveled to the dark forest.
```

Below the code editor, the terminal window displays a PowerShell session (PS) with the following commands and output:

```
PS C:\Users\PC> cd C:\Users\PC\conflict-lab
PS C:\Users\PC\conflict-lab> git add story.txt
fatal: pathspec 'story.txt' did not match any files
PS C:\Users\PC\conflict-lab> git add story.txt
PS C:\Users\PC\conflict-lab> git commit -m "Chapter 1: The beginning"
[master (root-commit) f10afbc] Chapter 1: The beginning
 1 file changed, 2 insertions(+)
 create mode 100644 conflict-lab/story.txt
PS C:\Users\PC\conflict-lab> git checkout -b happy-ending
Switched to a new branch 'happy-ending'
PS C:\Users\PC\conflict-lab> git add story.txt
PS C:\Users\PC\conflict-lab> git commit -m "Wrote the happy ending"
[happy-ending 17dc2c6] Wrote the happy ending
 1 file changed, 2 insertions(+), 1 deletion(-)
PS C:\Users\PC\conflict-lab> git checkout main
error: pathspec 'main' did not match any file(s) known to git
PS C:\Users\PC\conflict-lab> git branch
* happy-ending
  master
PS C:\Users\PC\conflict-lab> git checkout master
Switched to branch 'master'
PS C:\Users\PC\conflict-lab> []
```

2. Modify story.txt by adding a **different** line to the same spot:

Plaintext

The knight was eaten by a dragon. The End.

3. Save and commit:

Bash

```
git add story.txt
```

```
git commit -m "Wrote the tragic ending"
```

### **Part 3: The Collision (Merge Hell)**

#### Step 5: Attempt the Merge

You are now on the main branch. You try to merge the happy-ending branch into your tragic story.

Bash

```
git merge happy-ending
```

#### **Expected Output:**

Auto-merging story.txt

CONFLICT (content): Merge conflict in story.txt

Automatic merge failed; fix conflicts and then commit the result.

```
C: > Users > PC > conflict-lab > story.txt
1 Once upon a time, there was a brave knight.
2 The knight traveled to the dark forest.
Accept Current Change | Accept Incoming Change | Accept Both Changes | Compare Changes
3 <<<<< HEAD (Current Change)
4 The knight was eaten by a dragon. The End.
5 =====
6 The knight found a chest of gold and lived happily ever after.
7 >>>> happy-ending (Incoming Change)
8

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

create mode 100644 conflict-lab/story.txt
PS C:\Users\PC\conflict-lab> git checkout -b happy-ending
Switched to a new branch 'happy-ending'
PS C:\Users\PC\conflict-lab> git add story.txt
PS C:\Users\PC\conflict-lab> git commit -m "Wrote the happy ending"
[happy-ending 17dc2c6] Wrote the happy ending
1 file changed, 2 insertions(+), 1 deletion(-)
PS C:\Users\PC\conflict-lab> git checkout main
error: pathspec 'main' did not match any file(s) known to git
PS C:\Users\PC\conflict-lab> git branch
* happy-ending
  master
PS C:\Users\PC\conflict-lab> git checkout master
Switched to branch 'master'
PS C:\Users\PC\conflict-lab> git add story.txt
PS C:\Users\PC\conflict-lab> git commit -m "Wrote the tragic ending"
[master 9b7b17b] Wrote the tragic ending
1 file changed, 2 insertions(+), 1 deletion(-)
PS C:\Users\PC\conflict-lab> git merge happy-ending
Auto-merging conflict-lab/story.txt
CONFLICT (content): Merge conflict in conflict-lab/story.txt
Automatic merge failed; fix conflicts and then commit the result.
PS C:\Users\PC\conflict-lab> []
```

## Part 4: Visual Resolution (The Fix)

**Instructor Note:** This is the most important part. Students usually panic here. Show them how the tool makes it easy.

Step 6: Open the Editor

Look at story.txt in VS Code. You will see something that looks like this:

You will see colorful highlights and buttons provided by VS Code:

- <<<<< HEAD (Current Change): This is the "Tragic Ending" (what was already on Main).

- =====: The divider.
- >>>>> **happy-ending (Incoming Change)**: This is the "Happy Ending" (what you are trying to merge in).

#### Step 7: Choose Your Destiny

VS Code provides clickable buttons (small gray text) above the conflict:

- Accept Current Change (Keep the dragon).
- Accept Incoming Change (Keep the gold).
- Accept Both Changes (Keep both).

#### Action: Click **Accept Incoming Change**.

- Observation:* The strange symbols (<<<<, =====, >>>>) disappear, and only the "Happy Ending" text remains.

#### Step 8: Finalize the Merge

Now that the file looks correct, you must tell Git you are done.

- Save the file.

Run the commands:

Bash

```
git add story.txt
```

```
git commit -m "Resolved conflict: Chose the happy ending"
```

The screenshot shows a terminal window with two tabs open: 'app.py' and 'story.txt'. The 'story.txt' tab is active, displaying the following content:

```
C: > Users > PC > conflict-lab > story.txt
1 Once upon a time, there was a brave knight.
2 The knight traveled to the dark forest.
3 The knight was eaten by a dragon. The End.
```

Below the tabs, the terminal interface includes navigation buttons: PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL (which is underlined), and PORTS. The main area of the terminal displays a sequence of Git commands and their outputs:

```
PS C:\Users\PC\conflict-lab> git add story.txt
PS C:\Users\PC\conflict-lab> git commit -m "Wrote the happy ending"
[happy-ending 17dc2c6] Wrote the happy ending
 1 file changed, 2 insertions(+), 1 deletion(-)
PS C:\Users\PC\conflict-lab> git checkout main
error: pathspec 'main' did not match any file(s) known to git
PS C:\Users\PC\conflict-lab> git branch
* happy-ending
  master
PS C:\Users\PC\conflict-lab> git checkout master
Switched to branch 'master'
PS C:\Users\PC\conflict-lab> git add story.txt
PS C:\Users\PC\conflict-lab> git commit -m "Wrote the tragic ending"
[master 9b7b17b] Wrote the tragic ending
 1 file changed, 2 insertions(+), 1 deletion(-)
PS C:\Users\PC\conflict-lab> git merge happy-ending
Auto-merging conflict-lab/story.txt
CONFLICT (content): Merge conflict in conflict-lab/story.txt
Automatic merge failed; fix conflicts and then commit the result.
PS C:\Users\PC\conflict-lab> git add story.txt
PS C:\Users\PC\conflict-lab> git commit -m "Resolved conflict: Chose the happy ending"
[master f645cc1] Resolved conflict: Chose the happy ending
PS C:\Users\PC\conflict-lab> []
```