

Git Branching Model for Automation Framework

Jatin Shharma



Benefit the Branching Model for Framework



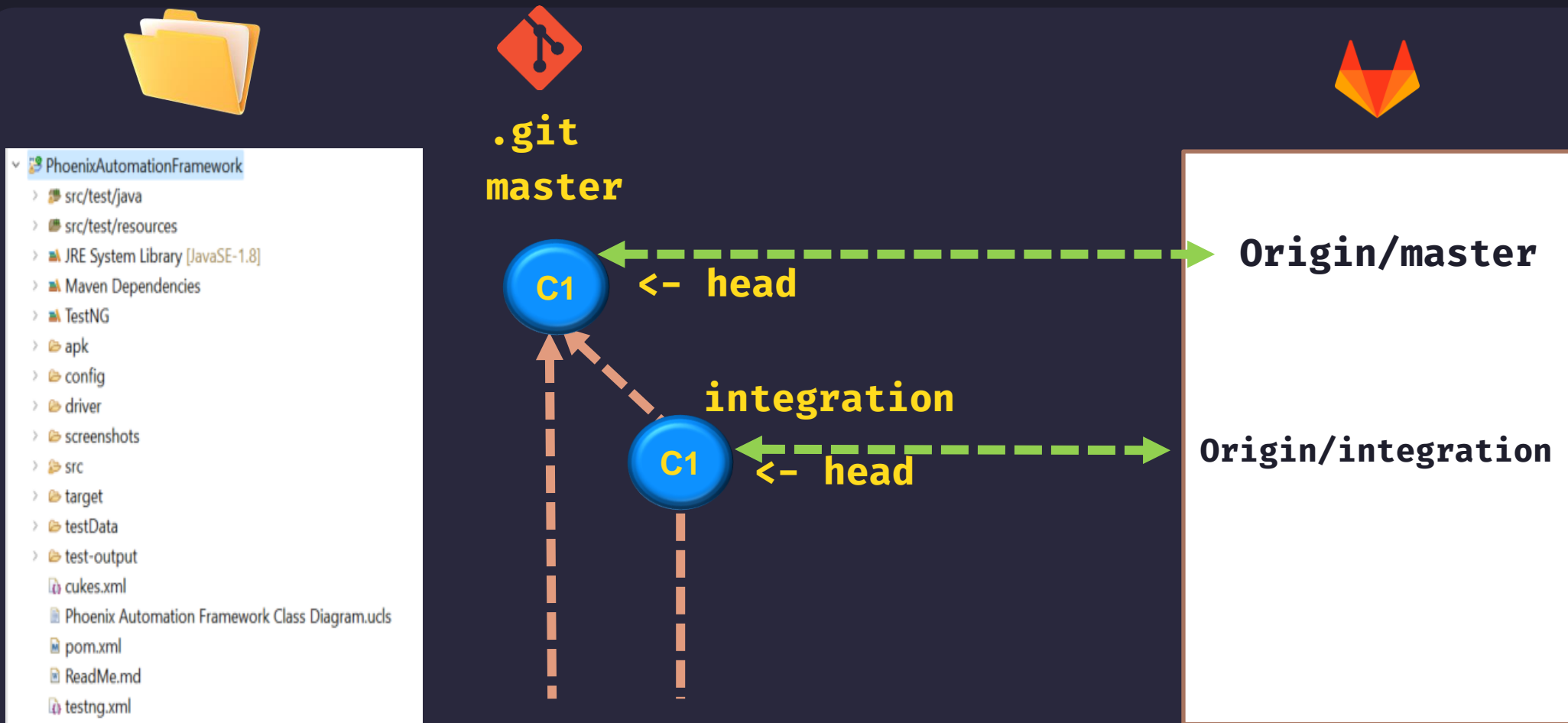
4 eyes Principles

Better **Knowledge Sharing**

Less Conflicts

Complete **Isolation of Work**

Initial Set up the Branching Model for Framework

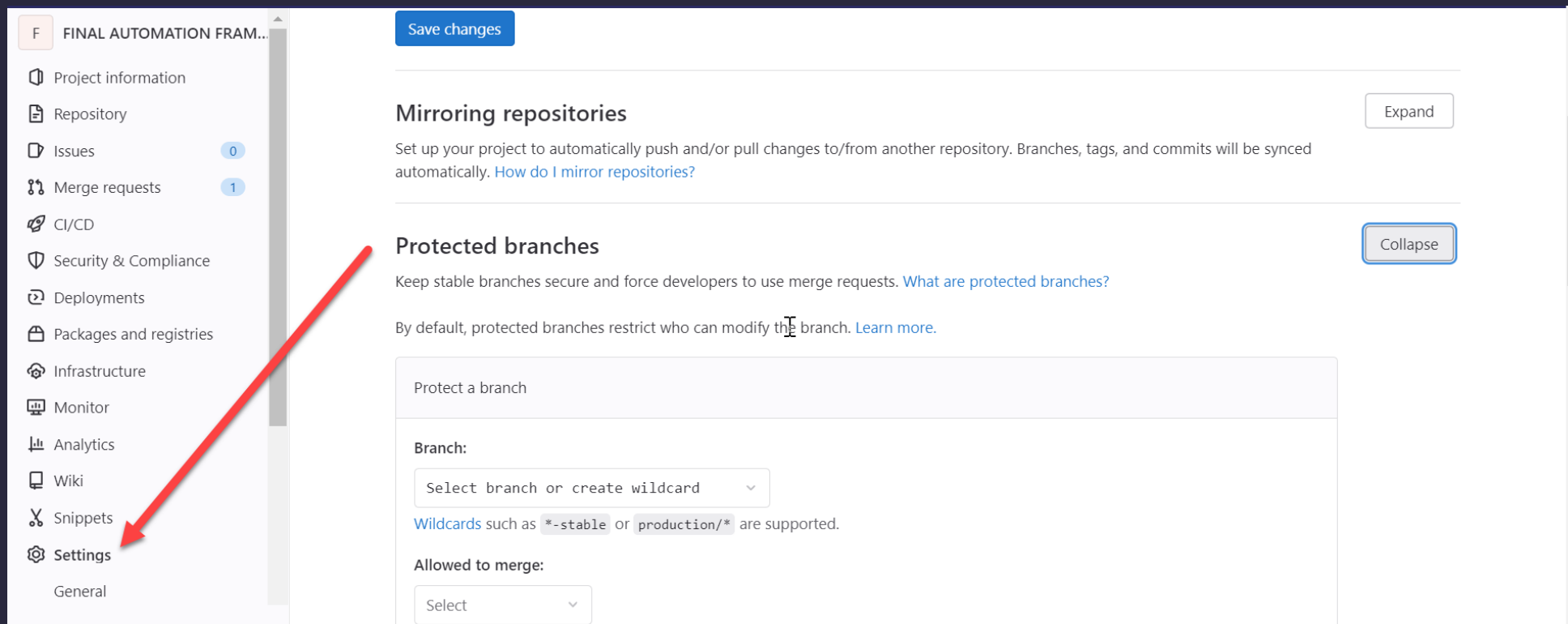


Setting up Rules on Branches in Gitlab

1. Make the master and integration branches as **protected**
2. A protected branch is a branch which is meant for pull and not push
3. No one is supposed to work on Protected Branches

Setting up Rules on Branches in Gitlab

Go to the Project Settings and select Repositories and scroll to Protected Branches



Setting up Rules on Branches in Gitlab

Go to the Project Settings and select Repositories and scroll to Protected Branches

F

FINAL AUTOMATION FRAM...

Project information

Repository

Issues 0

Merge requests 1

CI/CD

Security & Compliance

Deployments

Packages and registries

Infrastructure

Monitor

Analytics

Wiki

Snippets

Settings

General

Select

Allowed to push:

Select

Allowed to force push:

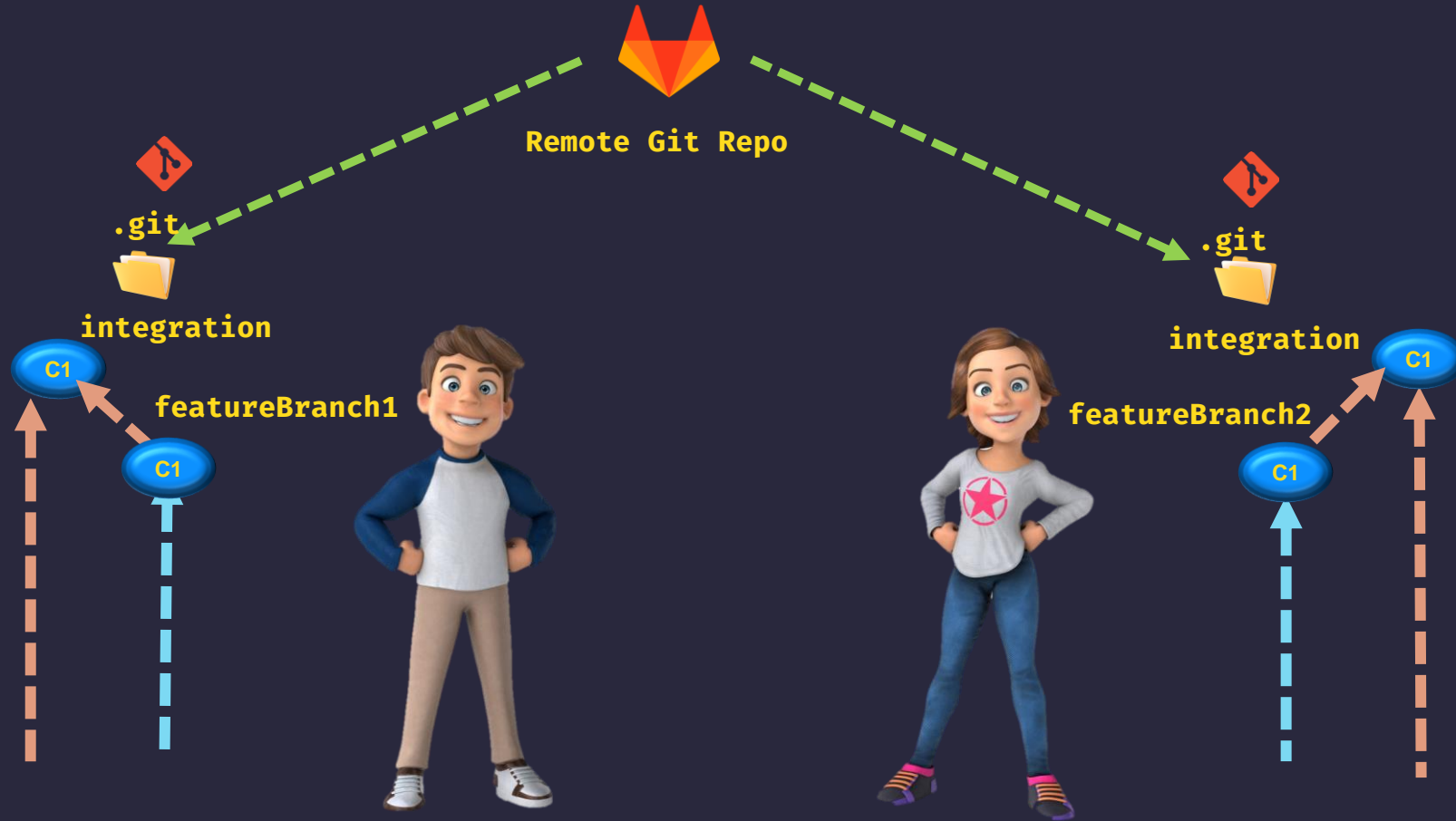
☒

Allow all users with push access to [force push](#).

Protect

Branch	Allowed to merge	Allowed to push	Allowed to force push ?
finalProject default	Maintainers	Maintainers	<input checked="" type="checkbox"/> Unprotect
master	Maintainers	Maintainers	<input checked="" type="checkbox"/> Unprotect

Workflow to Write new Test Scripts



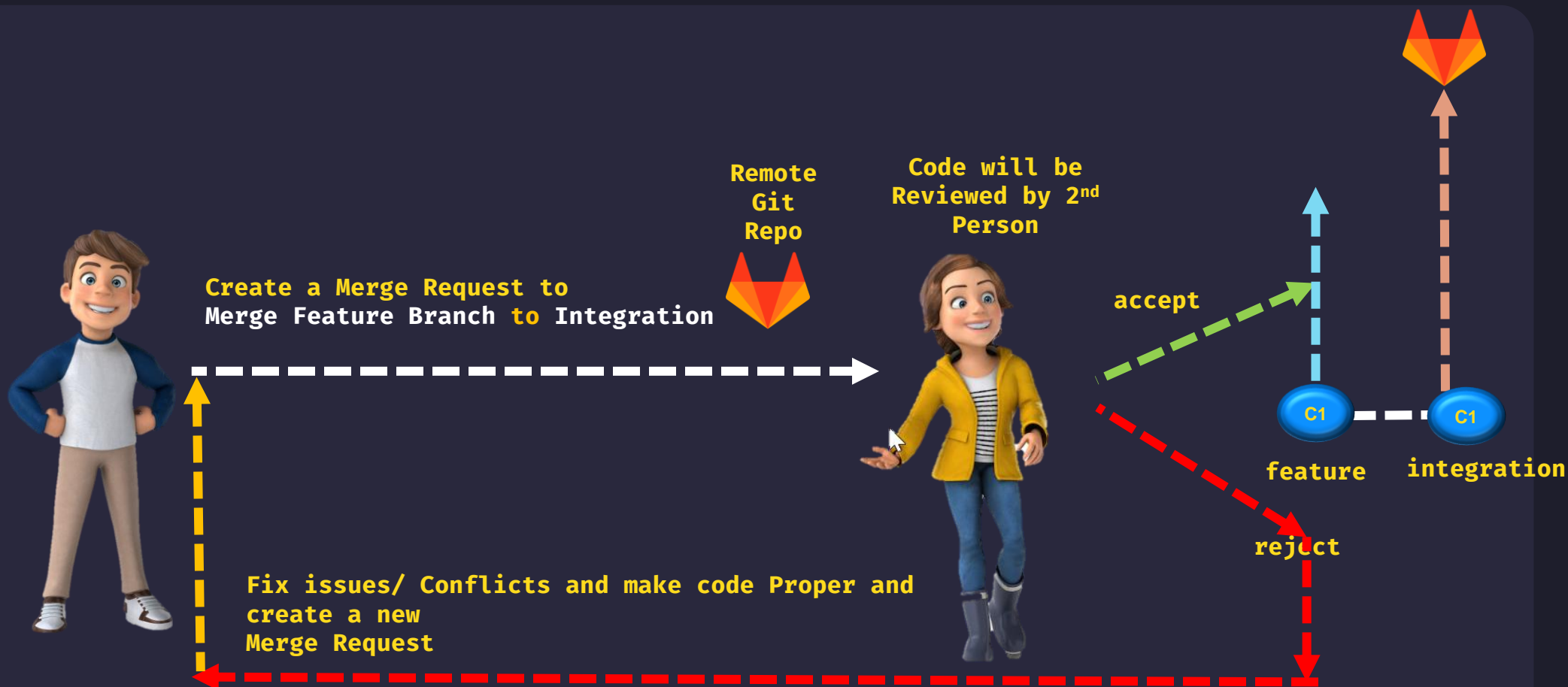
To create new features and test scripts every one will work on their **feature branches**

How new Test Script will be part of Remote Integration Branch



QAs will work on their respective feature branches and push the branch into the remote Repository and raise a Merge Request (MR) to merge Feature branches to Integration

Merge Request for Feature Branches to Integration



Merging Feature Branches into Integration Branch is a major deal breaker in your Project Cycle.

This is where our definition of done is achieved!

/02

Controlling Commit Message

Let's start ...



Setting up Git Message Template

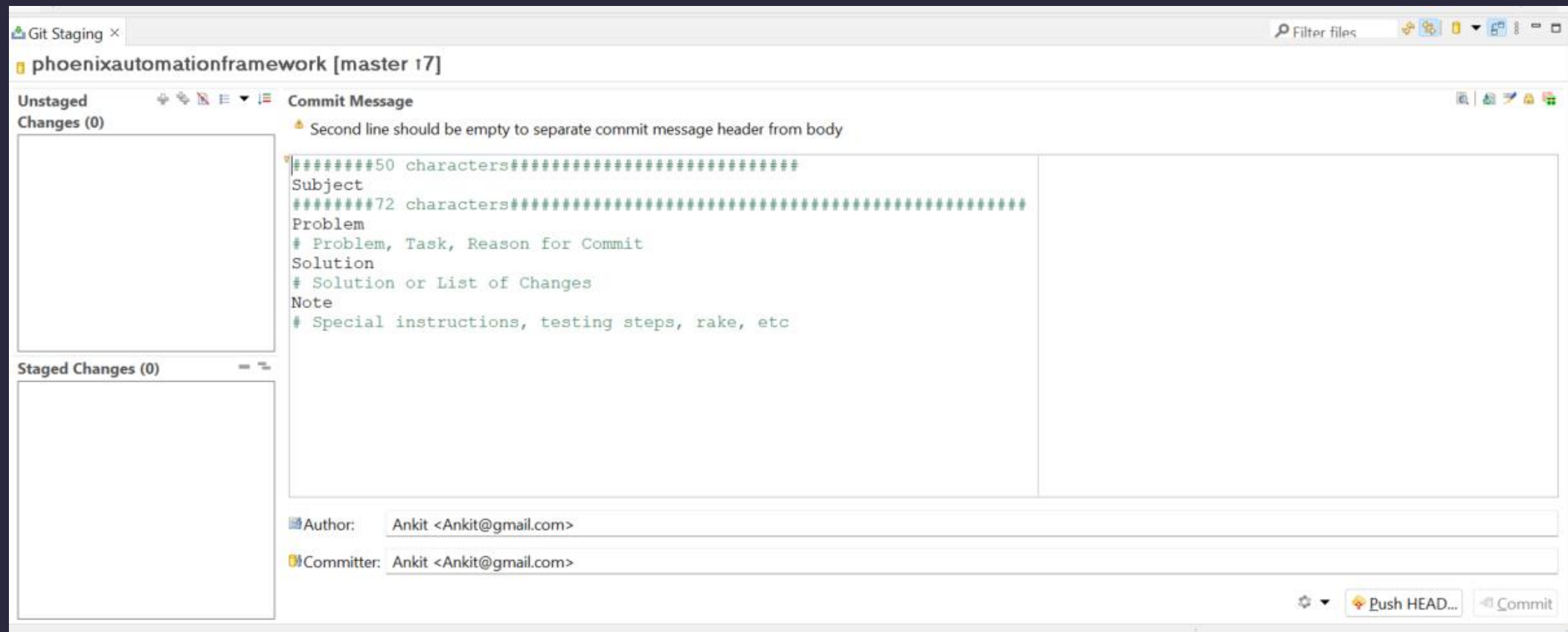
.gitmessage File helps in defining the **template** for all the commits from your system

```
≡ .gitmessage
1  #####50 characters#####
2  Subject
3  #####72 characters#####
4  Problem
5  # Problem, Task, Reason for Commit
6  Solution
7  # Solution or List of Changes
8  Note
9  # Special instructions, testing steps, rake, etc|
```

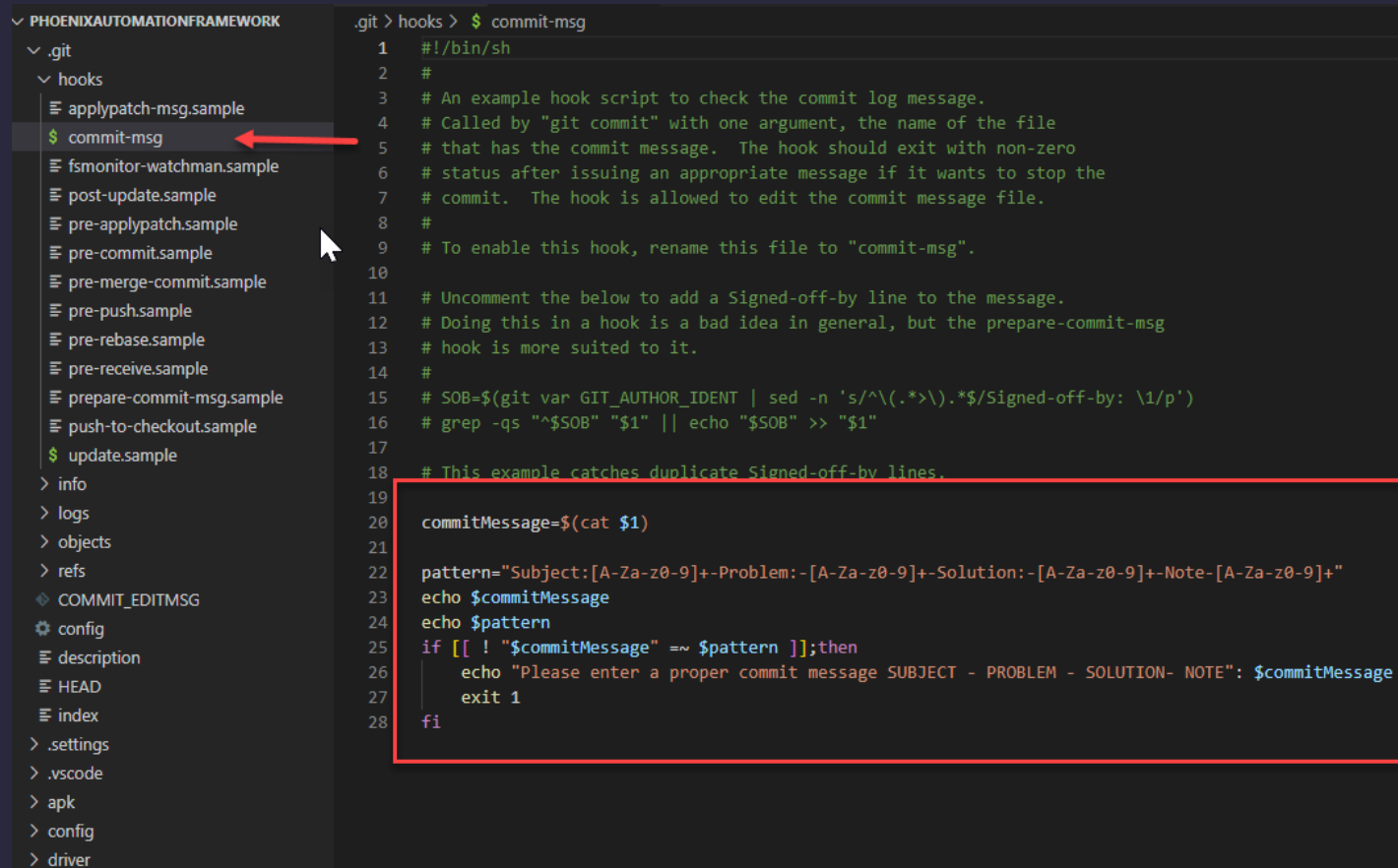
T

Command :`git config --global commit.template ~/.gitmessage`

How Eclipse/ VSCODE Commit Window will look Now



Setting up Git hook to control commit Message Template



```
.git > hooks > $ commit-msg
1  #!/bin/sh
2  #
3  # An example hook script to check the commit log message.
4  # Called by "git commit" with one argument, the name of the file
5  # that has the commit message. The hook should exit with non-zero
6  # status after issuing an appropriate message if it wants to stop the
7  # commit. The hook is allowed to edit the commit message file.
8  #
9  # To enable this hook, rename this file to "commit-msg".
10
11 # Uncomment the below to add a Signed-off-by line to the message.
12 # Doing this in a hook is a bad idea in general, but the prepare-commit-msg
13 # hook is more suited to it.
14 #
15 # SOB=$(git var GIT_AUTHOR_IDENT | sed -n 's/^(\.*)>\).*/Signed-off-by: \1/p')
16 # grep -qs "$SOB" "$1" || echo "$SOB" >> "$1"
17
18 # This example catches duplicate Signed-off-by lines.
19
20 commitMessage=$(cat $1)
21
22 pattern="Subject:[A-Za-z0-9]+-Problem:-[A-Za-z0-9]+-Solution:-[A-Za-z0-9]+-Note-[A-Za-z0-9]+"
23 echo $commitMessage
24 echo $pattern
25 if [[ ! "$commitMessage" =~ $pattern ]];then
26     echo "Please enter a proper commit message SUBJECT - PROBLEM - SOLUTION- NOTE": $commitMessage
27     exit 1
28 fi
```

Git hook controls commit Message

```
$ touch demo 123.txt

91983@Jatin MINGW64 ~/git/New folder/phoenixautomationframework (master)
$ git add .

91983@Jatin MINGW64 ~/git/New folder/phoenixautomationframework (master)
$

91983@Jatin MINGW64 ~/git/New folder/phoenixautomationframework (master)
$ git commit -m "ok"
ok
Subject:[A-Za-z0-9]+-Problem:-[A-Za-z0-9]+-Solution:-[A-Za-z0-9]+-Note-[A-Za-z0-9]+
Please enter a proper commit message SUBJECT - PROBLEM - SOLUTION- NOTE: ok
```

If our team doesn't enter the commit message consist of

Subject:

Problem:

Solution:

Note:

Then this commit will not happen!