**git help -->** It provides frequently used several git commands.

**git help <cmd-name>** **-->** It opens documentation of that perticular command.

**git init** --> It is used to create empty repository or re-initialise existing repository.

**git status -->** This command will display status of current repository.

Staged Files :

Files which are added and ready to commit.These file name will be displayed in green color.

Un-Staged Files :

Modified files will be displayed here,we need to stage these files to commit.

These file name will be displayed in red color.

Un-Tracked Files :

Newly created files ,we need to stage them to commit.

These file name will be displayed in red color.

**git add -->** This command is used to add file to staging area.

**Syntax** : git add <file-name> // ('.'/'--a' ==> all files)

**git rm -->** This command is used to un-stage newly created/added files.

**Syntax** : git rm --cached <file-name> // '\*' ==>all files

**git commit -->** This command is used to commit our changes to git local repository.

**Syntax** : git commit -m 'commit-message'

***Note :***  When we execute commit command it will consider all files which are in staging area.

* *To commit our files in remote repository we should execute below two commands*

**1) git remote add<repo-url>** 🡪 This is only first time

**2) git push -u origin master**  🡪 This is used to move changes from local to central

***Note :*** Git local repository available in our machine only.

**git reset -->** It is used to unstage a file which is already existing and try to modify

**Syntax** : git reset HEAD <file-name>

**git checkout -->** It is used to discard changes done in the file (We can say it 'Un-do " operation).

**Syntax** : git checkout --<file-name>

**git push -->** To publish local commits to central repository

***Note :*** Whenver we commit ,git will generate commit-id(40-alpha-numeric char but we see only 7 char)

**git log -->** To check commits history we will use git log command.

**Syntax :** git log

In commit logs it will display below details

* commit-id
* author
* timestamp
* commit msg

|  |
| --- |
| *Our TL commited project folder structure to git repository.*  *Developers job to clone that project and start their development.* |

**git clone -->** To take existing project from repository to local system we will use git clone command.

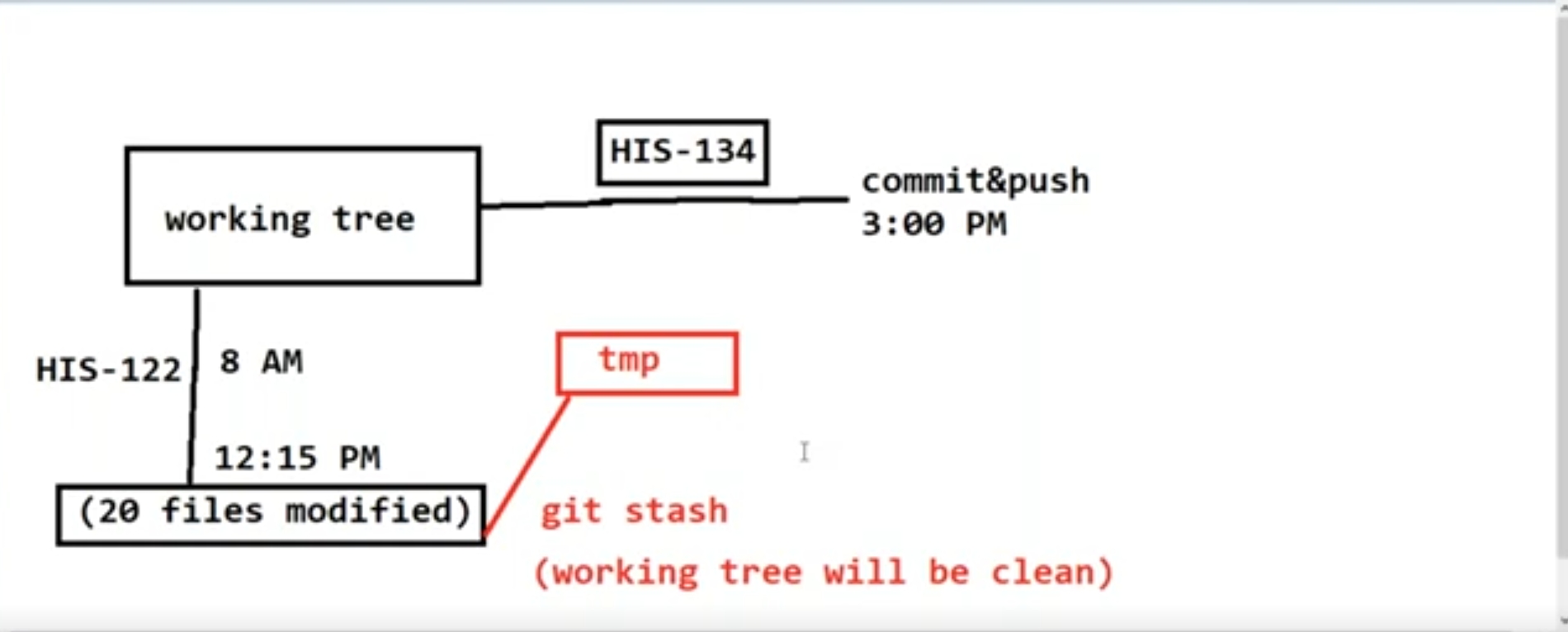
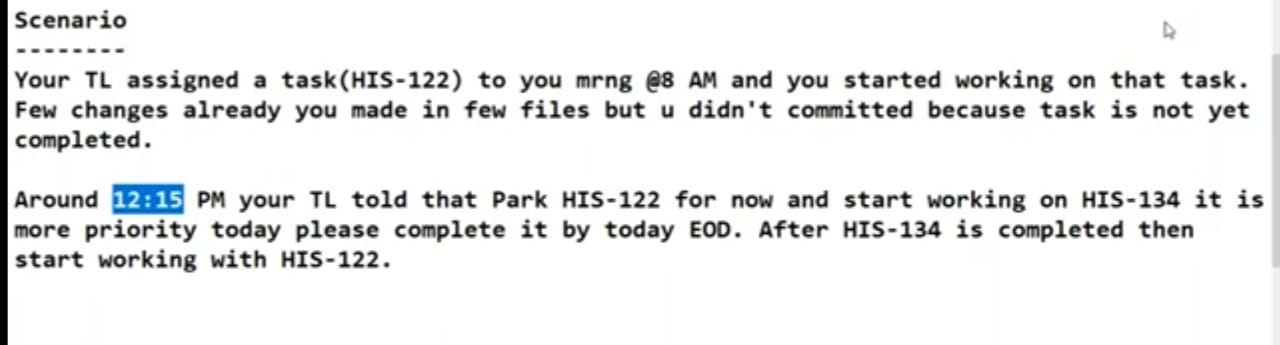
**Syntax** : git clone <repo-url>

**git pull -->** Before making any changes to file(s) in local, it is highly recomonded to take latest changes from repository.For this we use pull command.

**Syntax** : git pull

**git stash -->** It is used to record current changes and make working tree clean.

*git help stash ,git stash list, git stash clear*



**git stash apply -->** It is used to get the file(s) which we are modified and save in *temporary memory(stash).*

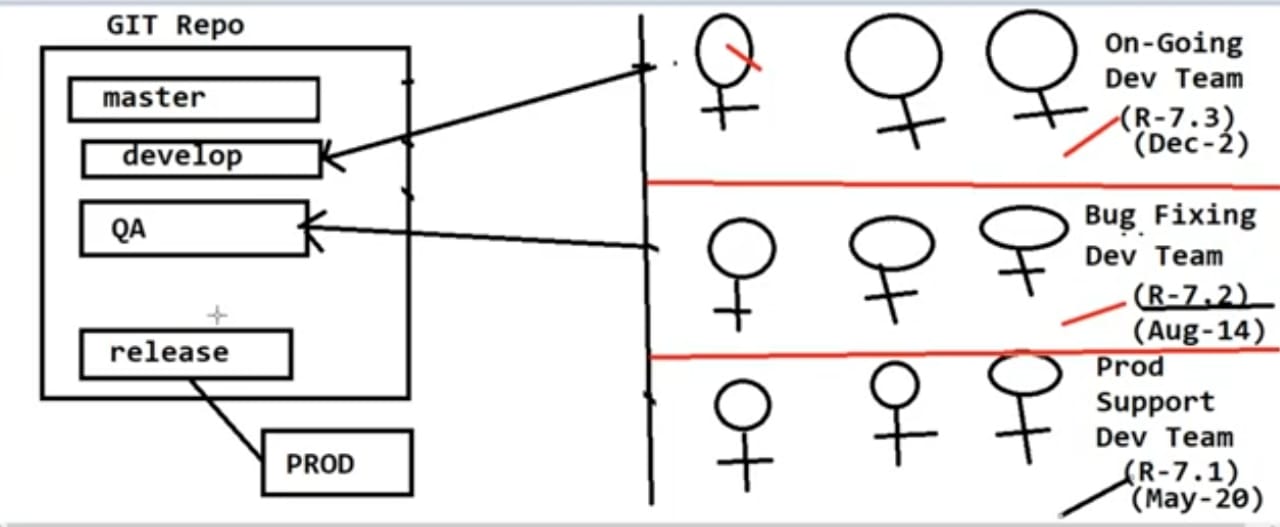
**Example :** If we are consider the above scenario ,after finishing most priority story(HIS-134) we want to start again previous story(HIS-122) for these we use *git stash apply.*

*What is branch in GIT HUB ?*

* When we create Git Repository by default it will provite master branch.
* Branches are nothing but code bases
* We can create several branches : Git Repository
* Generally in git repository we will create branches like below
* master(default)
* develop(on-Going Dev team,Bug fixing Dev team,Prod Support Dev)
* feature
* QA
* UAT
* release

*Why we need branches in repository?*

* To support parallel development we need to have branches in git repository.



*How to create branches in GIT repository?*

* Login to git repository
* create branch <ex. develop> from master branch
* Clone <develop> branch code

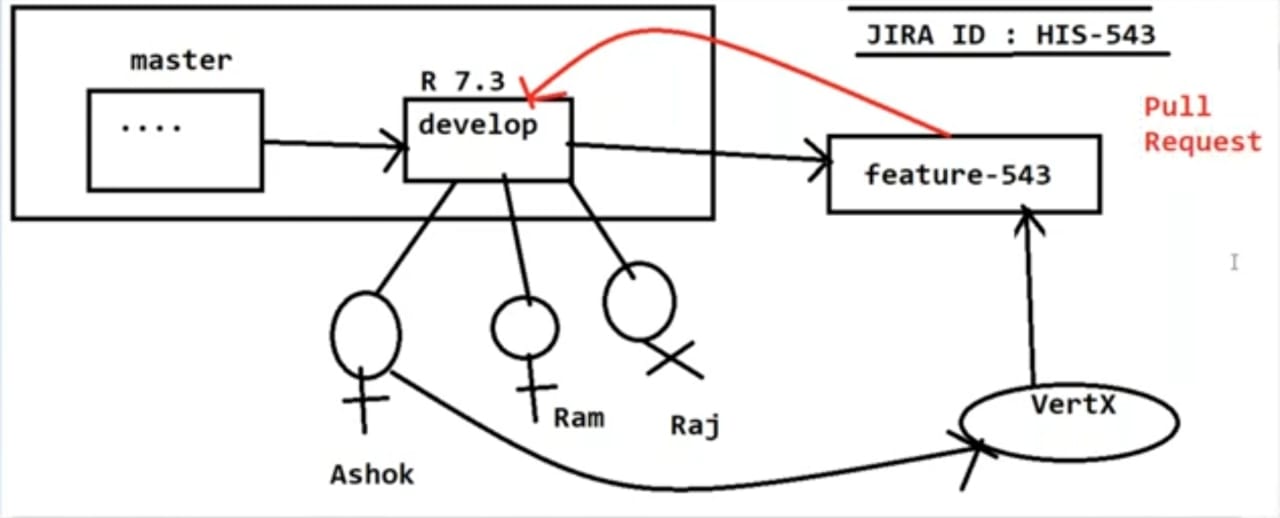
**git clone -b <branch-name> <repo-url> -->** It is used to clone branch.

**git branch-->** It is used to get the branch name.

***Note :*** If we execute *git clone<repo-url>* it clones master branch code by default.

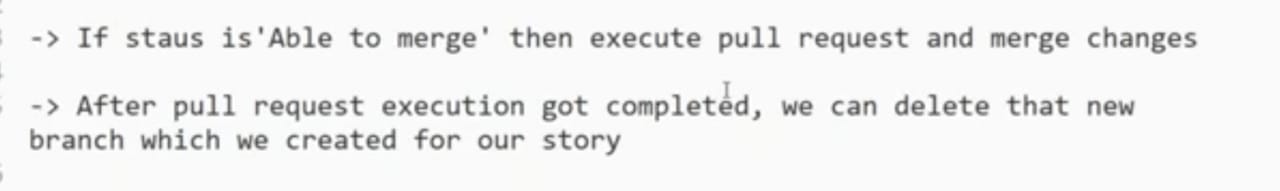
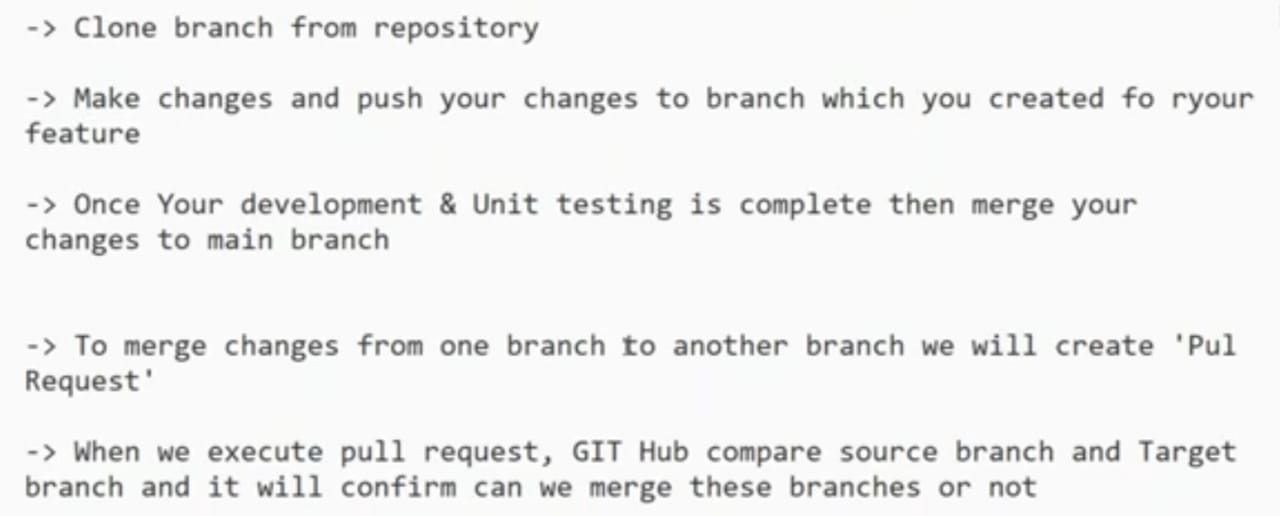
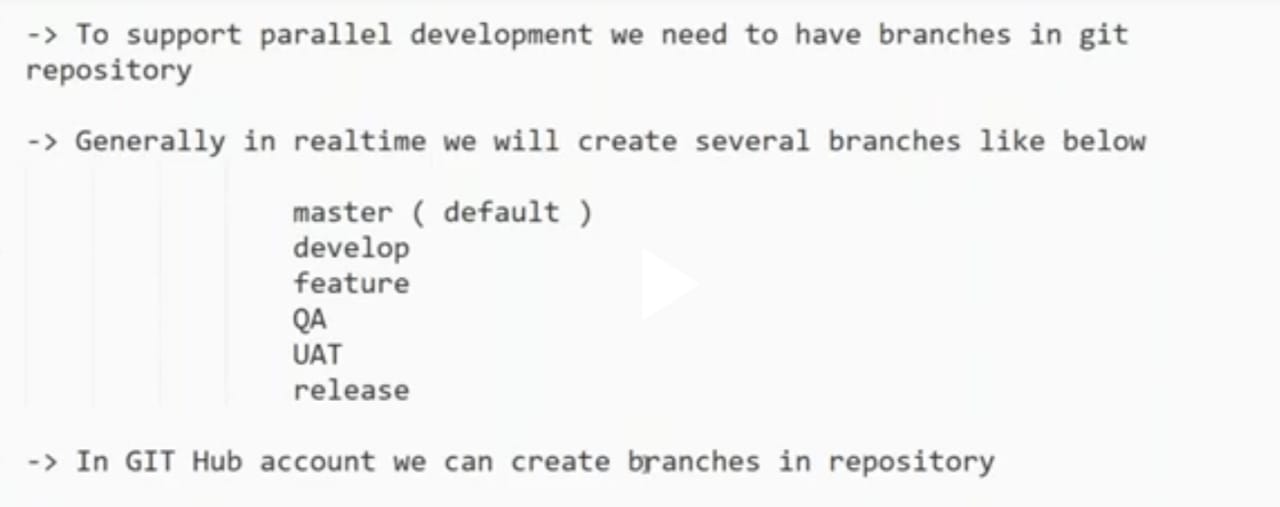
*How to merge branches in GIT Repository?*

* Using **pull request** we merge branches in Git Repository.

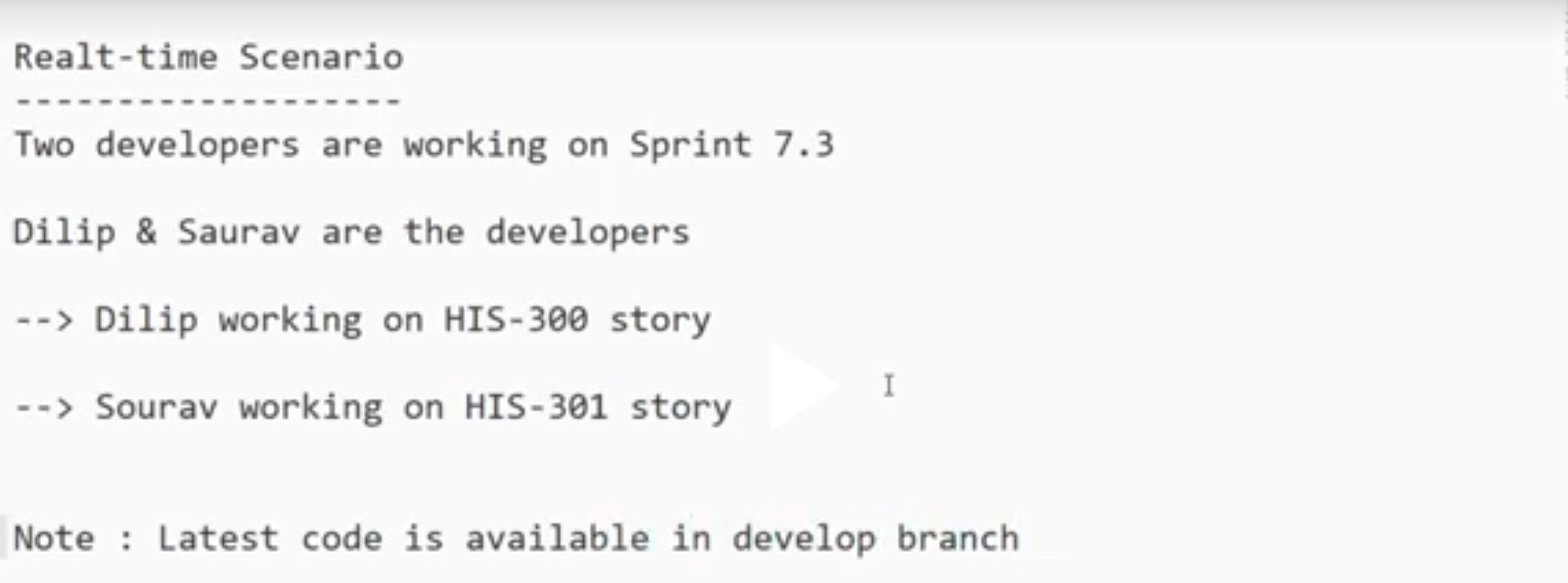


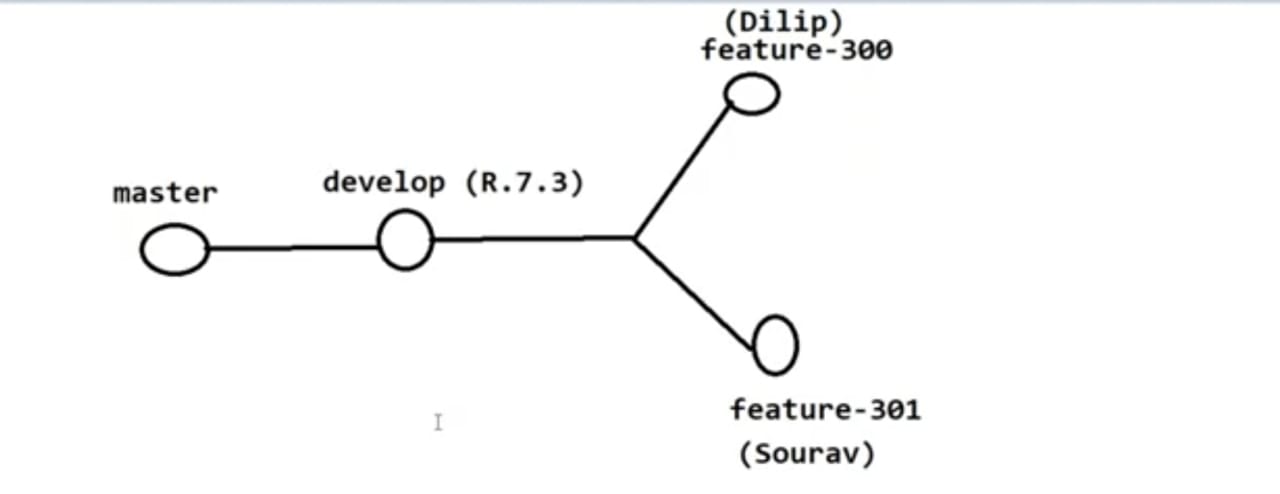
*What is pull request?*

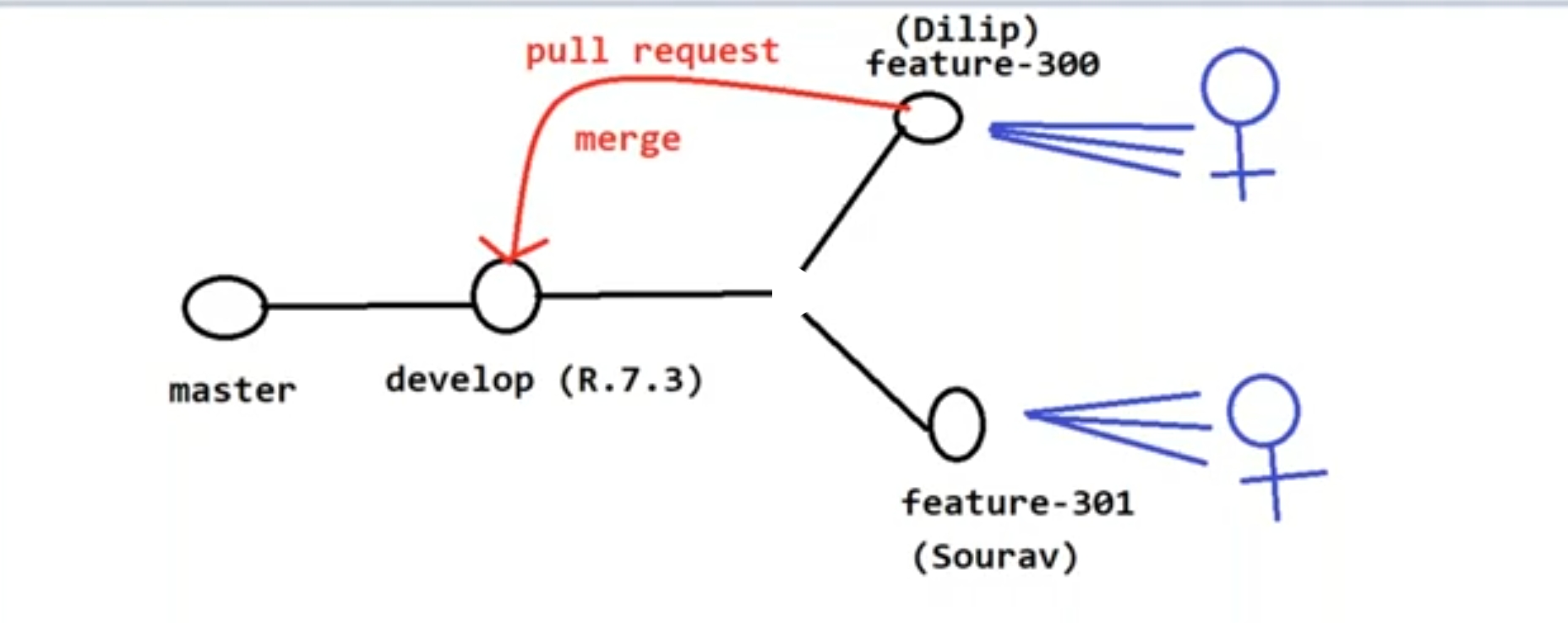
* Pull Request is used to merge one branch code to other branch .



*How to resolve conflicts in code?*







* When two or more developers change code at same line no. that causes conflicts at the time of branch merging.
* To solve conflicts to merge branches we have to use manual approach
* Go to GIT HUB repository
* Compare and pull request ( shows conflicts occur)
* Create pull request
* Resolve conflicts (This branch has conflicts that must be resolved)
* Remove the junk character(conflicts in the code) from the code
* Mark as resolve and Commit Merge
* Merge pull request
* Delete the unused branch