# How to integrate GitHub Enterprise Cloud account with Jira software?

# **Problem Description**

The user wants to integrate GitHub Enterprise Cloud (GHEC) with Jira.

#### Resolution

#### Overview

The users can integrate their GitHub Enterprise Cloud account with Jira Cloud to get quick access to branches, commits and pull requests in the context of Jira Software issues being worked upon. This increases the efficiency during planning and stand-up meetings. The following article describes how to set up this linking.

### **Pre-requisites**

Make sure the connection between your GitHub Enterprise Cloud repositories and Jira Cloud account is working fine by checking the *Development section* of Jira issue on the left-hand side pane. If the Development section comprise of 3 subparts-*Branch*, *Commit and Pull requests* then the connection is enabled. If none of these subparts are present then user should raise a Incident Request through ServiceNow, refer to this article to raise request.

Note: The development section comes only for Jira Software projects.

#### Steps to be followed

#### Part A: Create a Branch on Git Repository

1. Open Jira and open an existing Jira issue or create a new issue.

**Note:** Pay attention to the **Key** of Jira issue.

2. After opening a specific issue, go to *Create branch* option in Development section on left hand side pane. Git command to create a branch will pop up, use this command in command line to create the required branch. Please remember to use the Key of that specific Jira issue as a part of the branch name for integration to work, this is very important step.

git checkout -b <br/>branch name including Jira issue Key as substring>

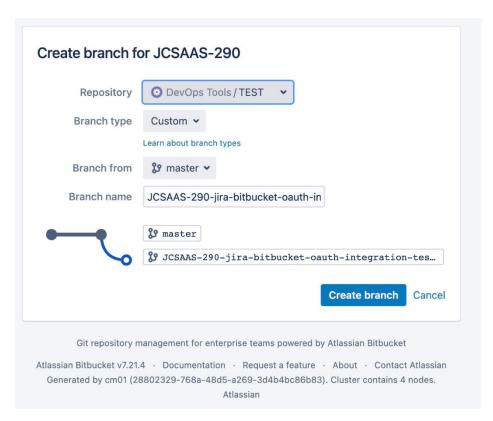
For example: If the Jira issue is having key- GHEC-4, then the commands can be

```
git checkout -b branch1-GHEc-4
git checkout -b branch xyz GHEC-4-feature-abc
```

**Note:** Make sure to clone the Git repository on local machine using **git clone** command before

running **git checkout** command. After git checkout command, use **git push** command.

- 3. The branch count will be updated under Development section of Jira issue, clicking on branch will give further details of the linked Git repository.
- 4. Create a branch from Jira ticket by clicking on the small square right hand side of **Create branch**. Add **Repository**, and **Branch name** as shown below.

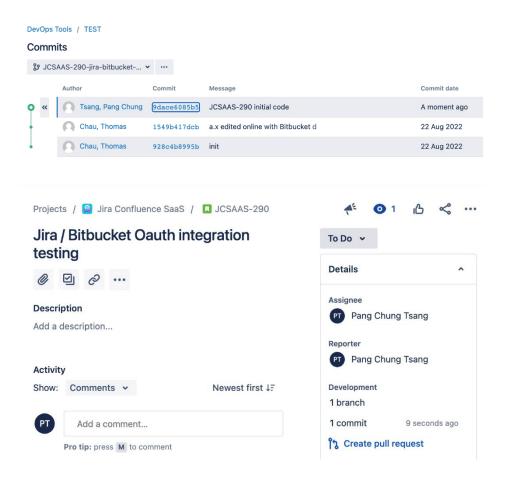


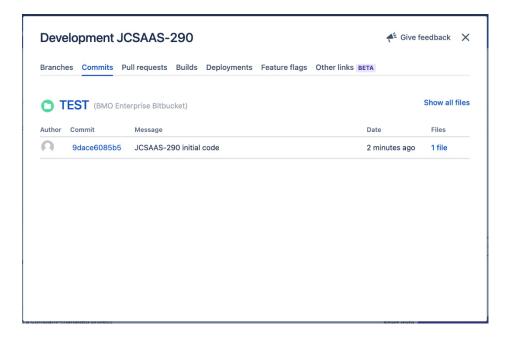
Then Bitbucket branch is linked to development tab as shown below.

# **Development JCSAAS-290**

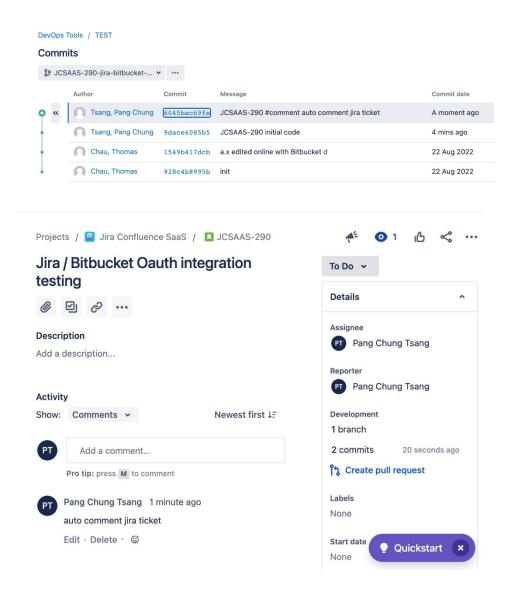


5. Commit code with issue key in the comment, links to the Jira ticket and Development Tab.

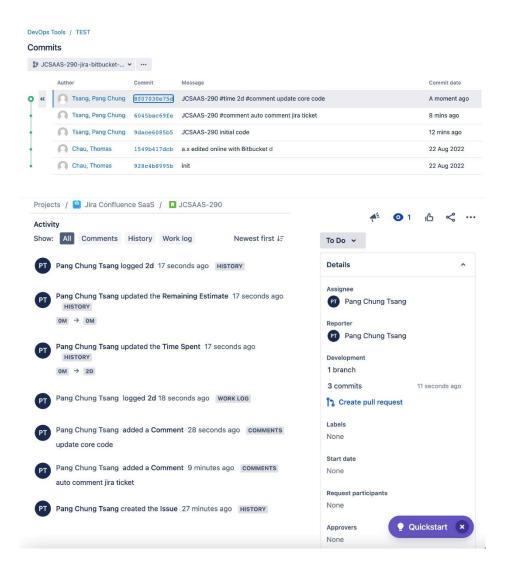




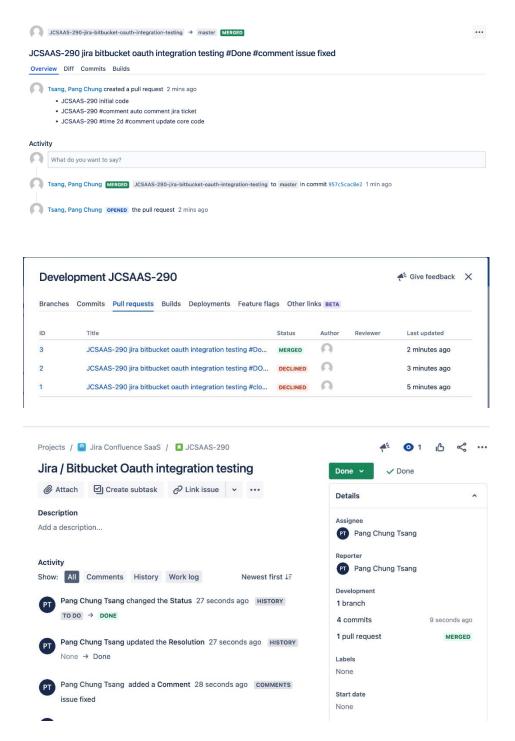
6. Commit code with message **<issue key> #comment <comment>**as descibed, then Comment will be automatically added to the Jira ticket as shown.



7. Commit code with message **<issue key> #time <time>.** Smart Commit is created, and work will be logged to the Jira ticket.



- 8. To perform smart commit with transition, create PR and comment the PR with message **<issue key> #transition.** As a result, PR will be linked to the Jira ticket and the ticket will be moved to expected state.
  - 1. Completion of the workflow transition changes the ticket status to **Done.**
  - 2. #comment in the message will be processed and added to the ticket comment.
  - 3. All PR with the ticket number will be listed including previous declined PR.



Part B: Commit code linking to a Jira Issue

- 1. Open the linked GitHub repository by clicking on *branch* from the Development section on the left hand side pane of the Jira issue.
- 2. Click on *Commits* beside Branches to check if there are any prior commits.
- 3. Go to branch of GitHub repository where you want to make a commit.

4. Do the code changes by editing the files in the selected branch and scroll down to Commit changes section. While filling in the *Update Commit-test field*, make sure to include the *Key* of Jira issue as substring and click Commit changes button.

**Note:** Below Update Committest field there is a field to add optional extended description. Even if the *Key* is added as substring in *optional extended description* only, then also the commit will get linked to the Jira issue.

- 5. Go to the Development section of Jira issue, the count of commit will be updated.
- 6. Click on *Commit* in Development section of the Jira issue and a pop-up reflecting the recent commit will open. The commit is now linked with the Jira issue and can be accessed by clicking on *commit hash*.

## Part C: Creating Pull Request on Git Repository using Jira

- 1. Click on *Create pull request* in Development section of the Jira issue and a drop-down reflecting branches will come. Select the branch that you want to create a pull request on. In case there is already a pull request in place then use "*plus*" button to create next new pull request.
- 2. You will be directed to *Open a pull request* page of the GitHub repository.
- 3. Click on *Create pull request* button on right, the panel will expand. In the *Title* field, make sure to enter the *Key* of the Jira issue and then again click on *Create pull request* button.
- 4. The pull request is created and linked with the Jira issue. Under the *Development section* of the Jira issue pull request will be updated with current Status as *Open*.
- 5. Now, open the pull request from Jira by clicking on pull request *Title* and merge or decline the request in GitHub. Depending on the GitHub operation, the pull request status gets updated to *Declined* or *Merged*.