



**CODE COMMIT IN BITBUCKET**

**CODE COMMIT IN BITBUCKET**

CONTENTS:

1. What is Bitbucket? ……………………………………………………………………………………………………………….…...3
2. Steps to commit code in Bitbucket ……………………………………………………….......................................4
   1. Cloning a repository from Bitbucket ........................................................................................4
   2. Connect to a remote repository ...............................................................................................4
   3. Branching a repository ............................................................................................................5
3. Update from the remote repository ...............................................................................................7
   1. Pull request process .................................................................................................................9
   2. Create pull.................................................................................................................................9
   3. Review process........................................................................................................................10
   4. Pull request authors................................................................................................................10
   5. Pull request reviewers.............................................................................................................10

**WHAT IS BITBUCKET?**

**Bitbucket** is a web-based [version control](https://en.wikipedia.org/wiki/Version_control) [repository](https://en.wikipedia.org/wiki/Repository_(version_control)) [hosting service](https://en.wikipedia.org/wiki/Shared_web_hosting_service) owned by [Atlassian](https://en.wikipedia.org/wiki/Atlassian), for [source code](https://en.wikipedia.org/wiki/Source_code) and development projects that use either [Mercurial](https://en.wikipedia.org/wiki/Mercurial) or [Git](https://en.wikipedia.org/wiki/Git_(software)) [revision control](https://en.wikipedia.org/wiki/Revision_control) systems. Bitbucket offers both commercial plans and free accounts. It offers free accounts with an unlimited number of private repositories.

[Bitbucket](https://www.atlassian.com/software/bitbucket/server?_ga=2.228536922.67184614.1544081135-324773371.1540208034&_gac=1.217328484.1540972295.CjwKCAjwpeXeBRA6EiwAyoJPKgPxmk1ANeYIkEvo_d076LK49XhHf74xJjT_OZ_rPunSCdTzP-3udxoCzdoQAvD_BwE) is a Git repository management solution designed for professional teams. It gives you a central place to manage git repositories, collaborate on your source code and guide you through the development flow. It provides awesome features that include:

* [Access control](https://confluence.atlassian.com/bitbucket/repository-privacy-permissions-and-more-221449716.html) to restrict access to your source code.
* [Workflow control](https://confluence.atlassian.com/bitbucket/branch-management-385912271.html) to enforce a project or team workflow.
* [Pull requests](https://confluence.atlassian.com/bitbucket/work-with-pull-requests-223220593.html) with in-line commenting for collaboration on code review.
* [Jira integration](https://confluence.atlassian.com/jiracloud/streamlining-your-development-with-jira-735642985.html) for full development traceability.

Bitbucket has 3 deployment models: Cloud, Bitbucket Server and Data Center.

**Image1:**

****

**STEPS TO COMMIT CODE IN BITBUCKET**

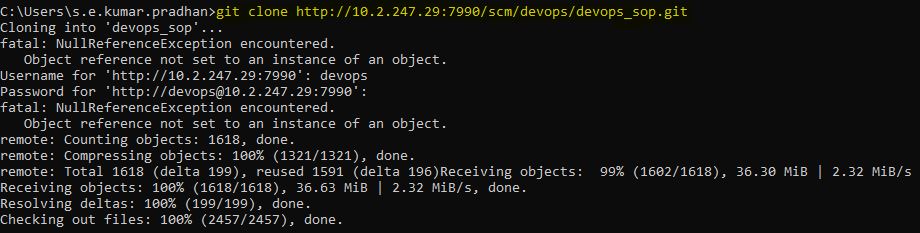
**Cloning a repository from Bitbucket:**

Cloning a git repository means that you create a local copy of the code provided by developer. You can simply do it with a command line.

Cloning a repository syncs it to your local machine. After you clone, you can add and edit files and then push and pull updates.

Command for cloning a repository is:  **Git <space>clone<space><URL>**

**Image2:**

****

**Connect to a remote repository:**

To connect your local repository to a remote server, add the server to be able to push to it.

Command for connection to remote repository is: **Git<space>remote<space>add<space>origin<server>**

git remote add origin <remote repository URL>

# Sets the new remote

Also, to list all currently configured remote repositories command is: **Git<space>remote<space>-v**

git remote -v

# Verifies the new remote URL.

>> origin http://10.2.247.29:7990/scm/devops/devops\_sop.git (fetch)

origin http://10.2.247.29:7990/scm/devops/devops\_sop.git (push)

**Branching a repository:**

Branching offers a way to work on a new feature without affecting the main codebase. You can create a branch from Bitbucket or from your terminal. After you make changes, you push your branch to Bitbucket so that you can get it reviewed with a pull request.

1. Command for creating a new branch and switching to it: **Git<space>checkout<space>-b<space><branch name>**

git branch -b <branch name>

# creates new branch and switches to it.

**Image3:**

****

2. Switch from one branch to another: **Git<space>checkout<space><branch name>**

git checkout <branch name>

# switches to the specified branch

>>\*dev

**Image4:**

****

3. List all the branches in your repository, and tell you what branch you're currently in: **Git<space>branch**

git branch

#Tells you the list of branches + current branch

>>\*master

dev

4. Delete the feature branch: **Git<space>branch<space>-d<space><branch name>**

git branch -d feature1

#Deletes the specified branch

5. Push the branch to your remote repository, so others can use it: **Git<space>push<space>origin<space><branch name>**

git push origin dev

#Pushes the specified branch onto your repository.

6. Push all branches to your remote repository: **Git<space>push<space>--all<space>origin**

git push --all origin

# Pushes the all branch onto your repository.

7. Delete a branch on your remote repository: **Git<space>push<space>origin<space>:<branch name>**

git push origin : feature2

# Deletes the specified branch on your remote repository.

**Update from the remote repository:**

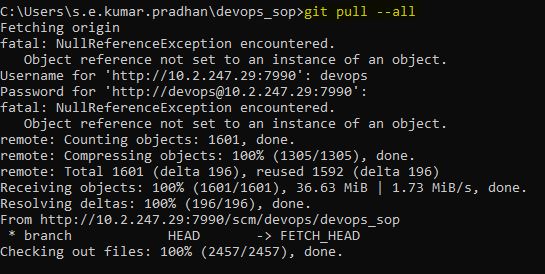
You can perform various actions on the repository like pulling the data from the repository to your local system, adding the file, committing the changes, pushing the files, merging the files to the dev branch, etc. Few of the commands are mentioned below:

1. Fetch and merge changes on the remote server to your working directory: **Git<space>pull<space>--all**

git pull --all

#Pulls the changes in remote repository up to the local repository.

**Image5:**

****

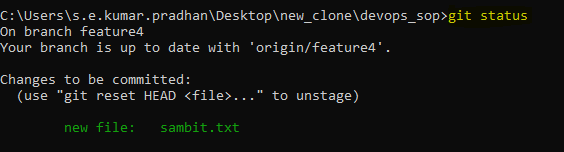
2. Add the files in your new local repository, this stages them for the first commit. After you have manually resolved any conflicts, you mark the changed file: **Git<space>add<space>.**

git add .

# Adds the files from the current directory of the local repository and stages them for commit.

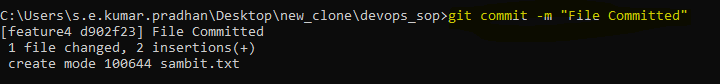
3. To check status of your remote repository: **Git<space>status**

**Image6:**



4. Commit the files that you've staged in your local repository: **Git<space>commit<space>-m<space><message>**

**Image6:**



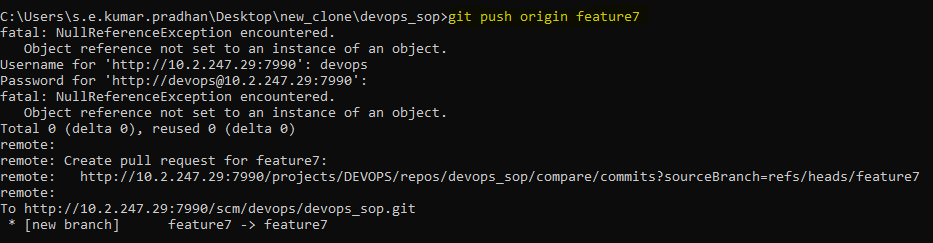
5. Git push to push commits made on your local branch to a remote repository:

**Git<space>push<space><remote name ><space><branch name>**

git push origin feature3

#Pushes the changes in your local repository up to the remote repository you specified as the origin.

**Image7:**



**Create PULL Request**

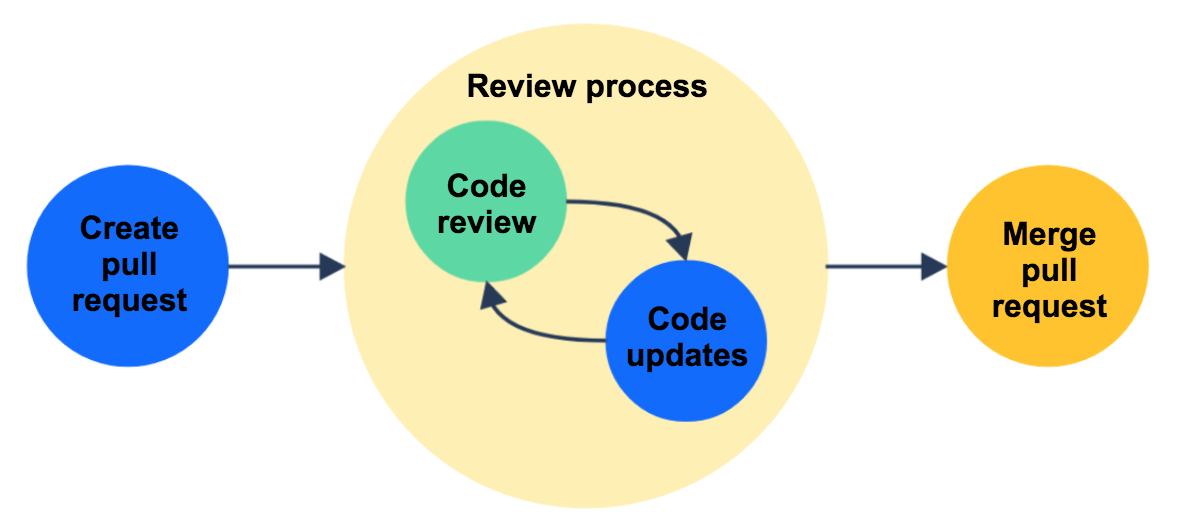
After you've added files and made updates to existing code, it's time to merge that code into your Bitbucket Cloud repository. Before you merge, you want to ensure that you maintain code quality and won't break already existing features. To get the feedback you need for code updates and improvements, you can create a pull request that includes all lines of code you've added.

**Pull requests** provide you with a method for requesting code reviews from your teammates and checking build status based on your most recent commit.

**Pull request process:**

Code review and collaboration are at the core of pull requests. Here's how the end to end process works, follow by more details about different steps in the process:

**Image8:**



1. **Create pull:** When you use branches or forked repositories to work on a separate line of code from the codebase, you can use pull requests to get your code reviewed and merge your changes from Bitbucket Cloud. When you're ready to start a discussion about your code changes, it's time to create a pull request. Before creating a pull request, you might want to compare your code changes to the destination repository.

2. **Review process**: After the author creates a pull request, several things can happen before a pull request is merged is mentioned below.

* **Reviewers** look through the diff to compare the changes with existing source code.
* **Reviewers** comment on changes and **authors** reply to comments, starting a discussion.
* **Anyone** can create a task based on a comment and **authors** resolve tasks as they complete them.
* **Authors** update code based on feedback and push new commits to the branch or fork, updating the pull request.
* **Authors** check that all tests are passing on the most recent commit.
* **Reviewers** approve the pull request when satisfied with changes.

Depending on your role, you may be an author, a reviewer, or both on two different pull requests.

3. **Pull request authors:**

To find pull requests you've created, check your **pull requests** list on your **work** dashboard.

As a pull request author, the code review process officially begins after you create the pull request with reviewers. If you don't add reviewers during creation, you can always edit the pull request to add them afterwards.

After you've created a pull request and added reviewers, you might be inclined to take a break while you wait for approvals. However, as reviewers look at your code and make comments, you'll receive email notifications of the ongoing discussion in the pull request, giving you an opportunity to respond and making you become an active participant in the code review process.

## 4. **Pull request reviewers:**

To find requests you've been asked to review, check the **Pull requests to review** list on your **work** dashboard. You can also go to the **Pull requests** page of your team's repositories to help your team with other pull requests where you're not listed as a reviewer.

When a team member adds you as a reviewer, we'll notify you by email. After the initial notification about pull request creation, you'll continue to get email notifications for these occurrences:

* The author makes updates
* Another user makes a comment
* Another reviewer approves
* Another user merges the pull request

To disable notifications, you can unwatched the specific pull request.

During your code review, you'll comment with feedback, suggestions, and ideas. You may take time to consider if there are any obvious logic errors, all cases are fully implemented, existing automated tests need to be rewritten, and the code conforms to existing style guidelines.

After your review, if the pull request is ready to be merged (or if you trust the author to resolve your tasks before merging), click the **Approve** button in the top right. A green checkmark appears next to your name in the **Reviewers** field when you've approve the pull request.