Commit Processing

1. [Prerequisite to test Commit Processing - Guide](https://servicenow.sharepoint.com/:w:/r/sites/QualityEngineering/Shared%20Documents/Releases/Rome/Partner%20Testing/Testing%20Guidance%20-%20Use%20Cases-Workflows/DevOps/Prerequisite%20Guide.docx?d=w8bd8085f5a2c473a82d912c63a5d57d8&csf=1&web=1&e=uqhglr)
2. [Commit Processing (Merge and Revert)](#_Commit_Processing_(Merge)
3. [Multiple work items association to commits](#_Multiple_work_items)
4. [Bulk Processing of Commits](#_Bulk_processing_of)
5. [Tags](#_Tags:)
6. [**Demo Link**](https://servicenow.zoom.us/rec/play/UT8vrg6NqLvk0d1evbcZTsNzHtL6TLrOfD5ATri6eIQ11GnaUOJ-4nKzF2nK200rYn0bMAietketKpQG.8FMP432UWX2N3rxQ?continueMode=true&_x_zm_rtaid=87let7W5TvSGc9tiKHtAFQ.1622028601642.eef24e1ce2d20533bb5ee82b88e9e26b&_x_zm_rhtaid=535)

## **Commit Processing (Merge and Revert):**

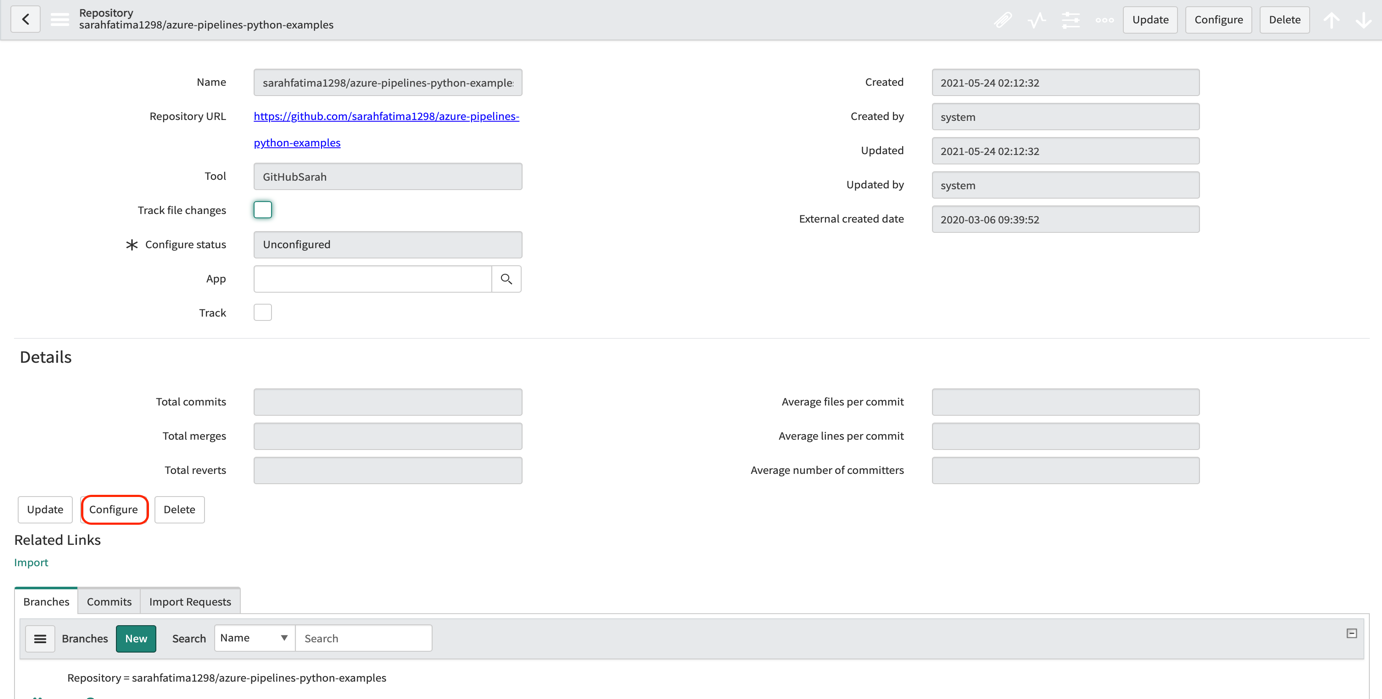
**Prerequisites:**

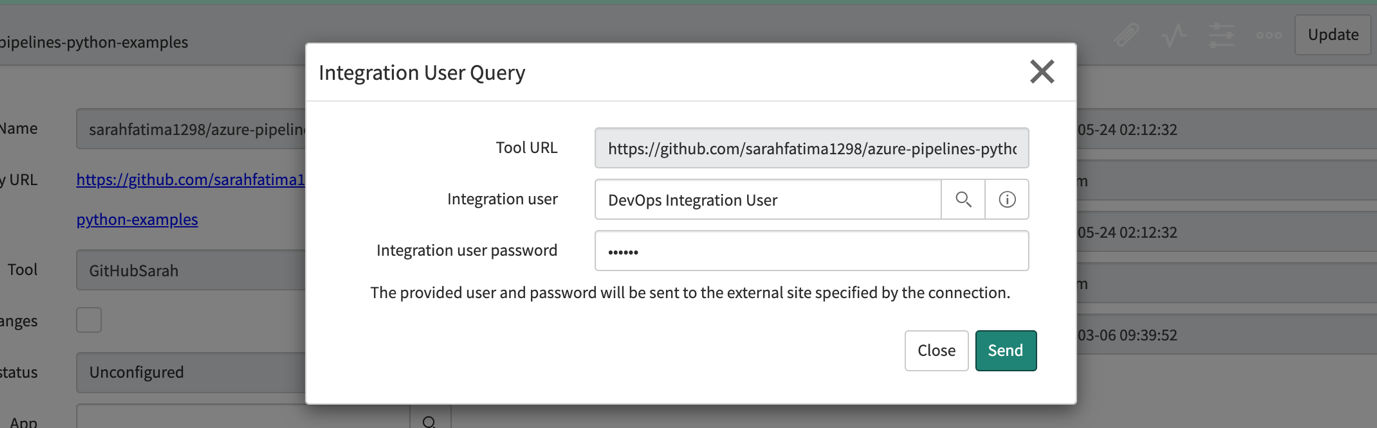
1. Coding tool in *connected* state and repository with *track* and *track file changes* enabled and *configured*

*Note: For the guide purpose we will use GitHub Coding tool*

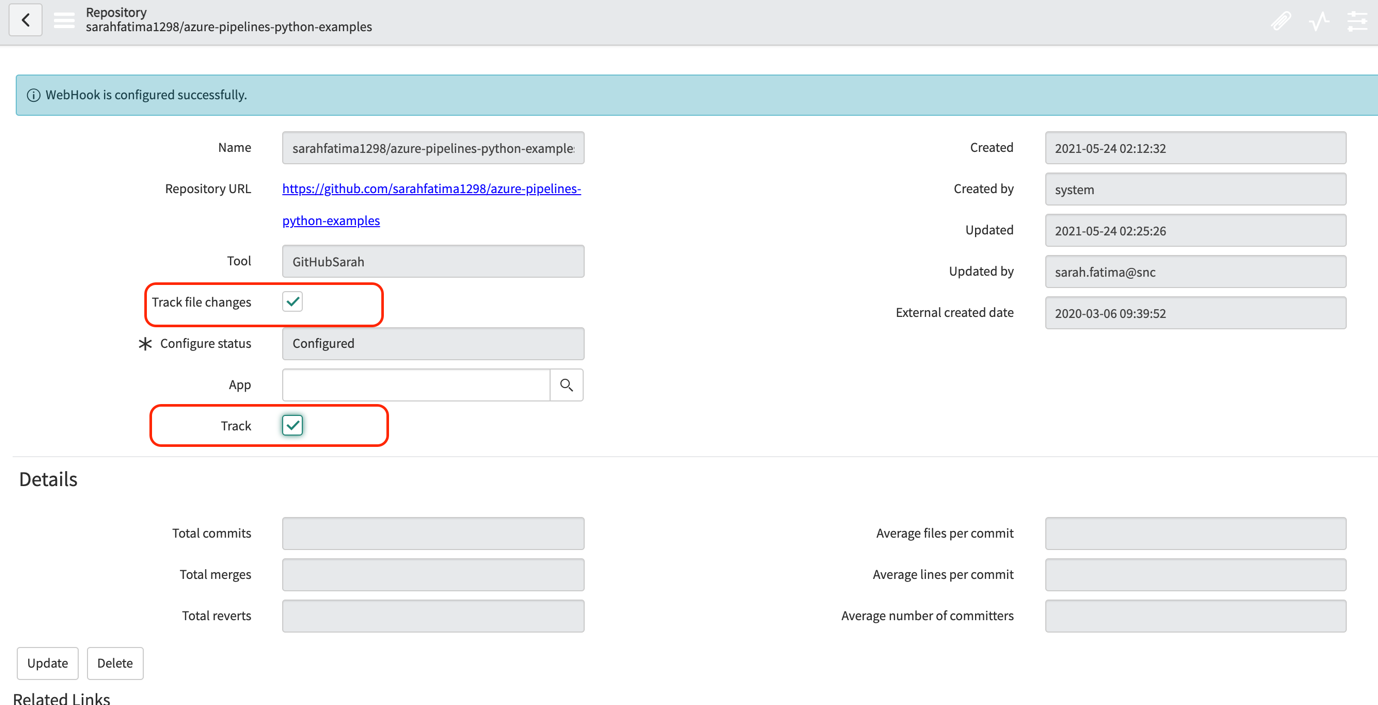
**Steps:**

1. To *configure* the repository and enable *track* and *track file changes* flags on the repository.
   1. Navigate to the repository you want to configure
   2. Click on *Configure* UI action, for configuring the repository (Username: *devops.integration.user* and it’s password)

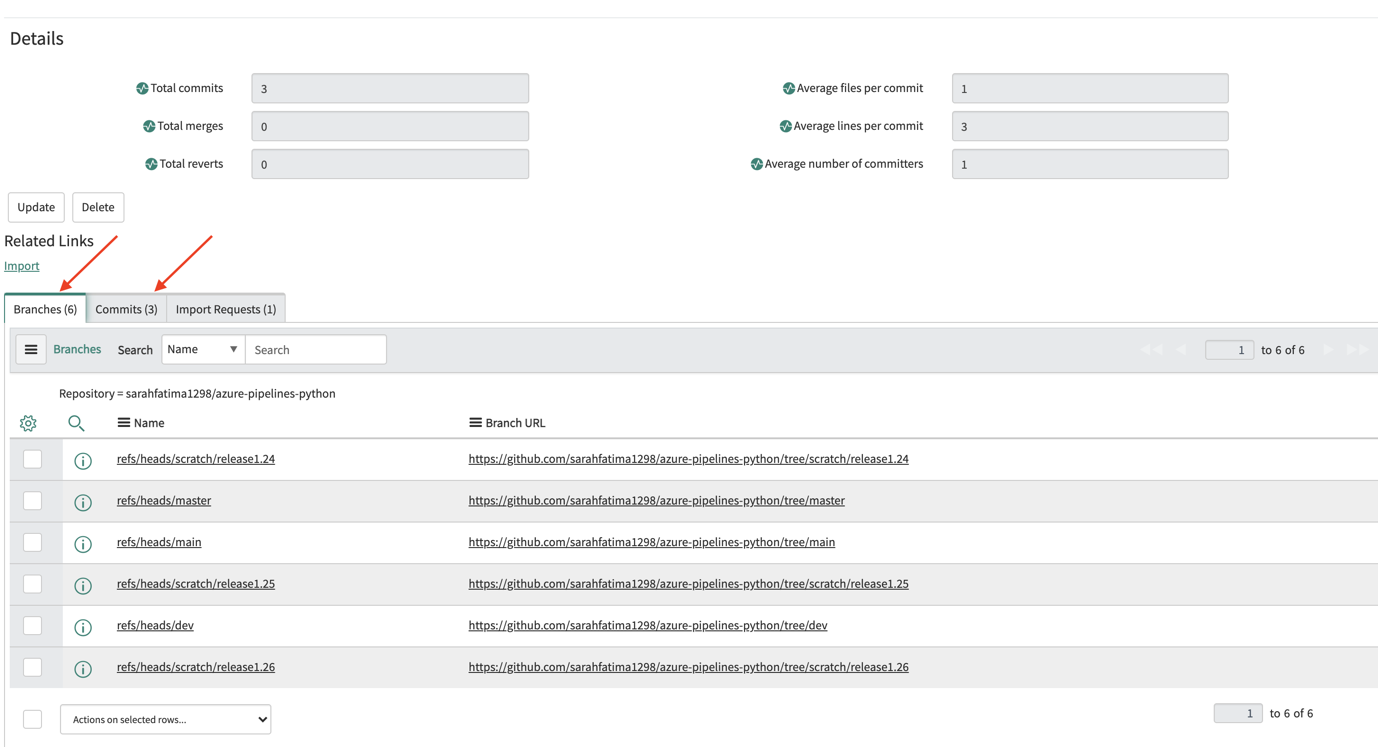




* 1. Enable tracking for the repository by checking the *Track* and *Track file changes* checkboxes for real-time tracking of the repository

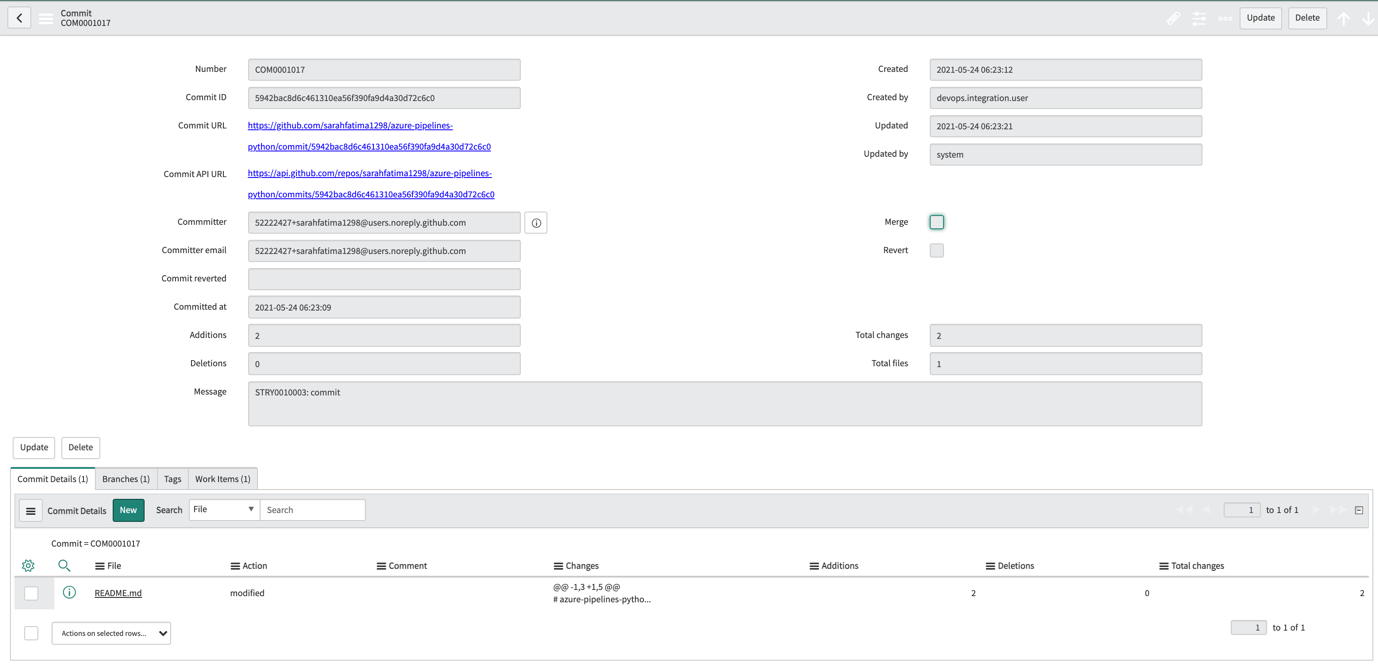


1. Once configured, make any commit or merge commits or revert commit to the repository on GitHub and the commit will be persisted in the *commits* related list on repository

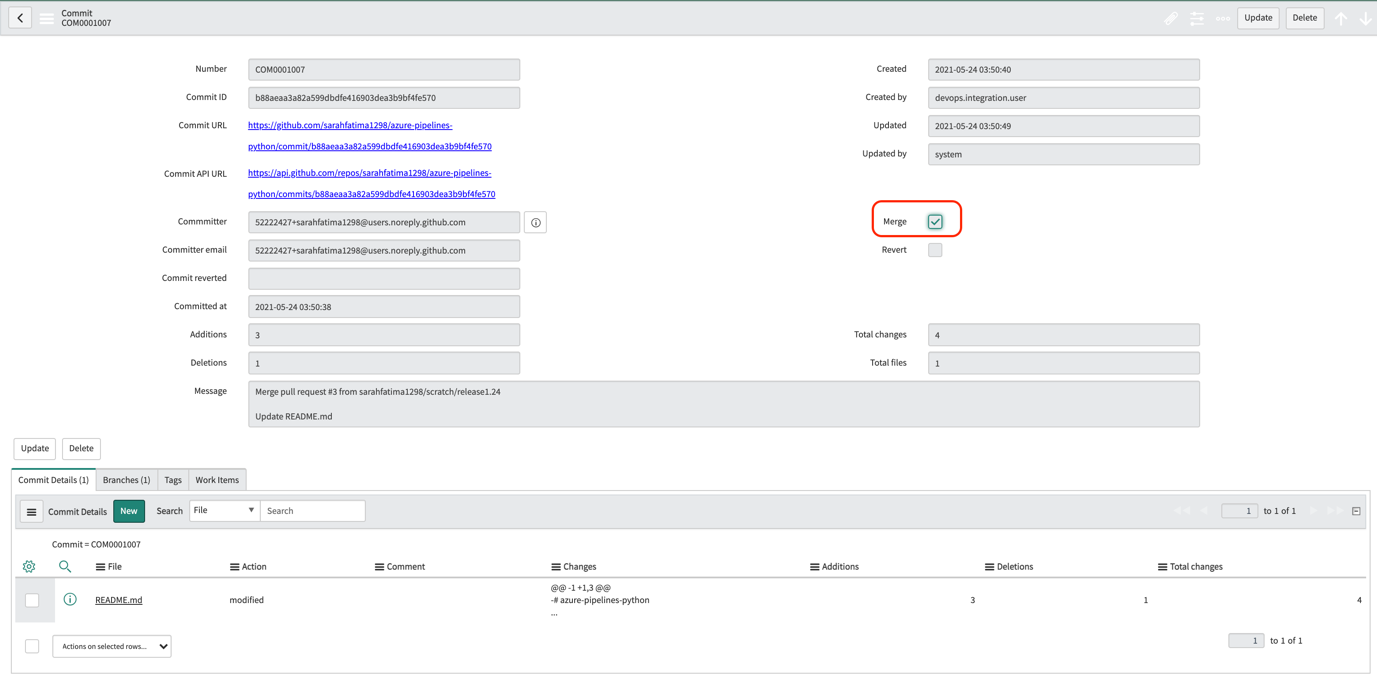


**Expected Results:**

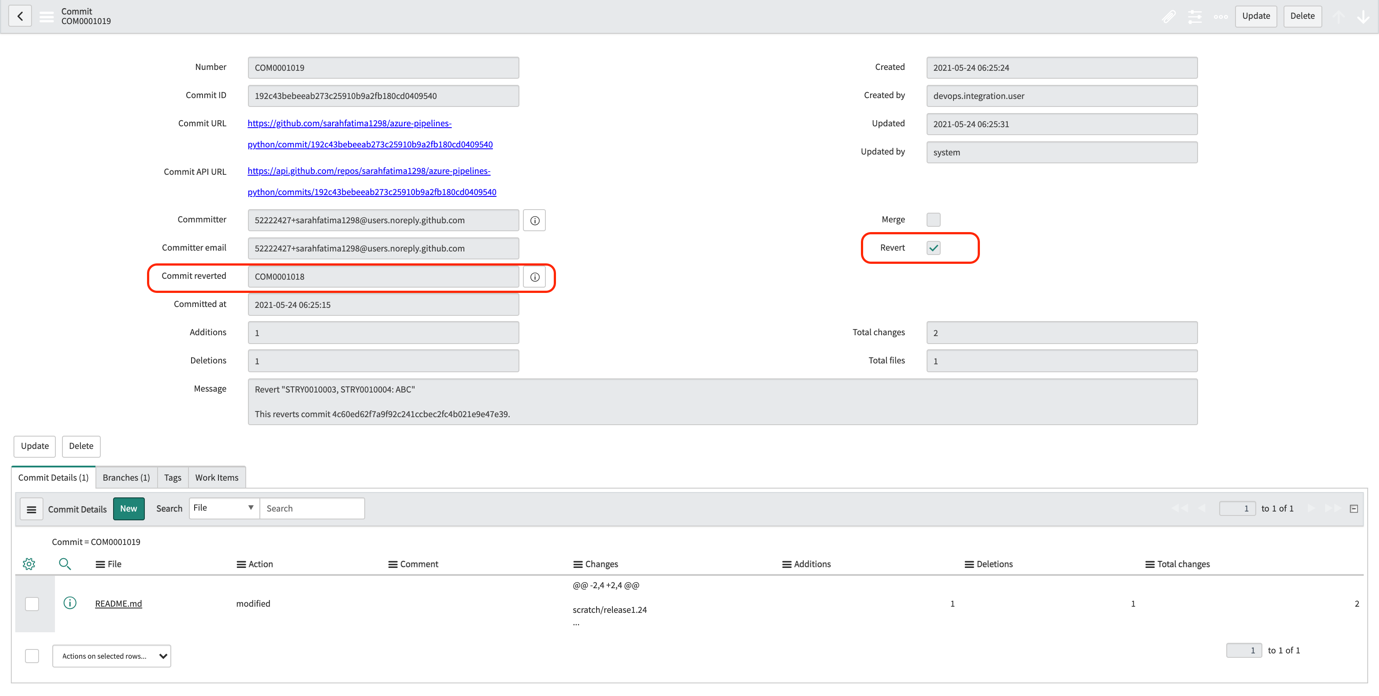
1. Normal commit will look something like this:
   1. Data related to files modified and additions and deletions can we viewed in *Commit Details*



1. Merge Commits: (The merge flag will be checked)
   1. Data related to files modified and additions and deletions can we viewed in *Commit Details*



1. Revert commit: (The revert flag will be checked)
   1. Data related to files modified and additions and deletions can we viewed in *Commit Details*

=

## **Multiple work items association to commits:**

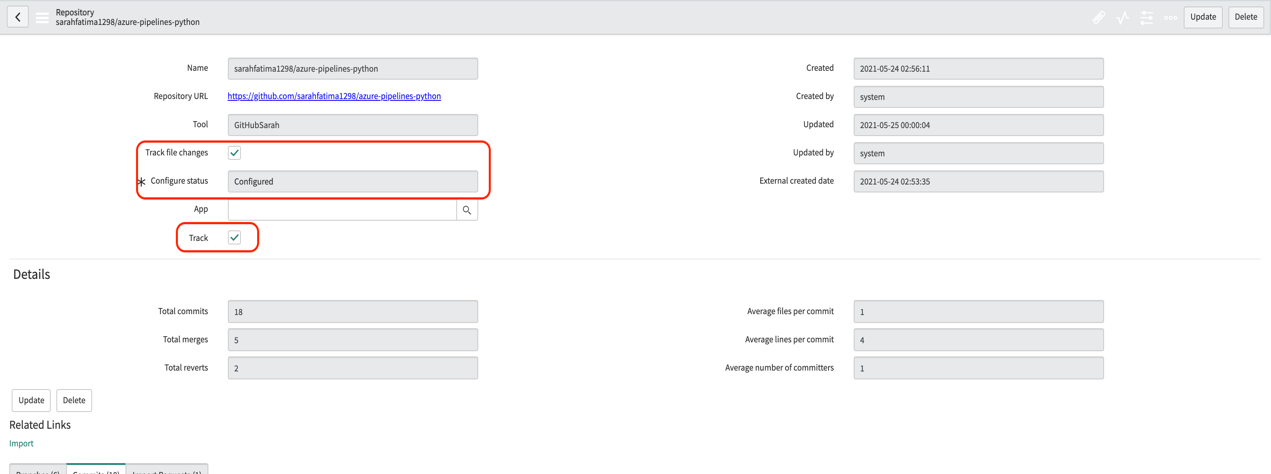
**Prerequisites:**

1. Coding tool in *connected* state and repository with *track* and *track file changes* enabled and *configured*
2. Planning tool with plans discovered and *track* enabled

*Note: For the guide purpose we will use GitHub Coding tool and Agile Development 2.0 (by ServiceNow) as our planning tool*

**Steps:**

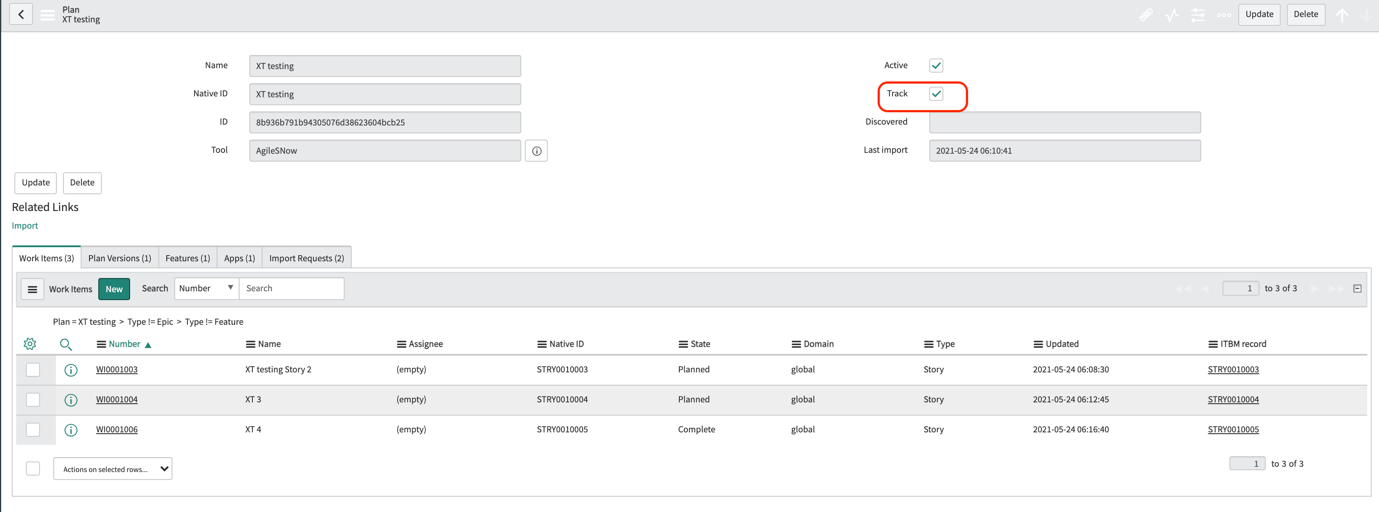
1. Once the prerequisites are met, the repository and plan record will look something like this:
   1. Repository:



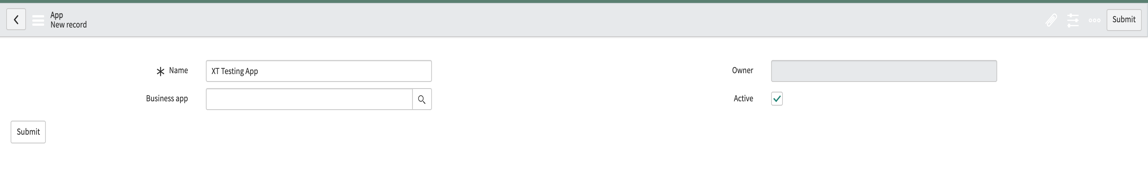
* 1. Plan:

Note: Agile Development 2.0 to ServiceNow mapping

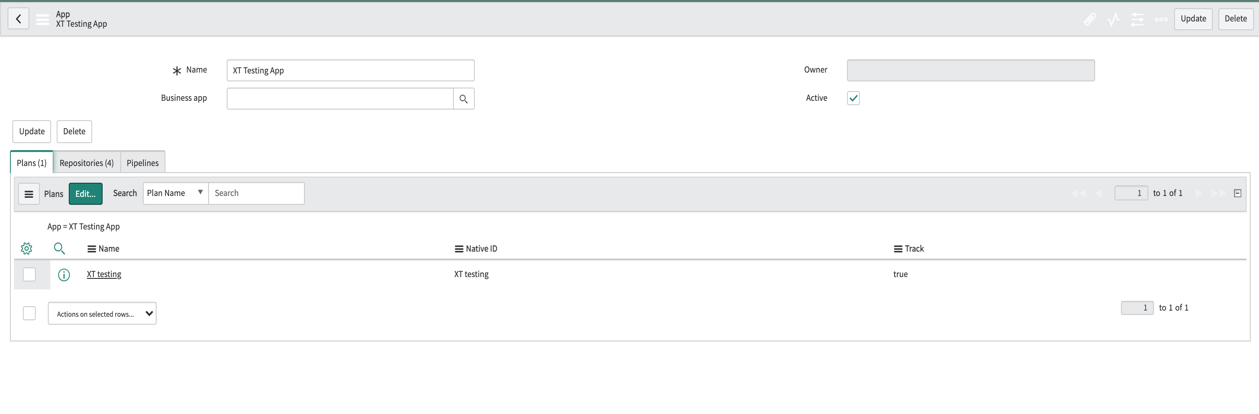
|  |  |
| --- | --- |
| Agile Development 2.0 | DevOps |
| Product | Plan |
| Release | Plan Versions |
| Epics | Features |
| Story | Work Items |
| Defects | Work Items |

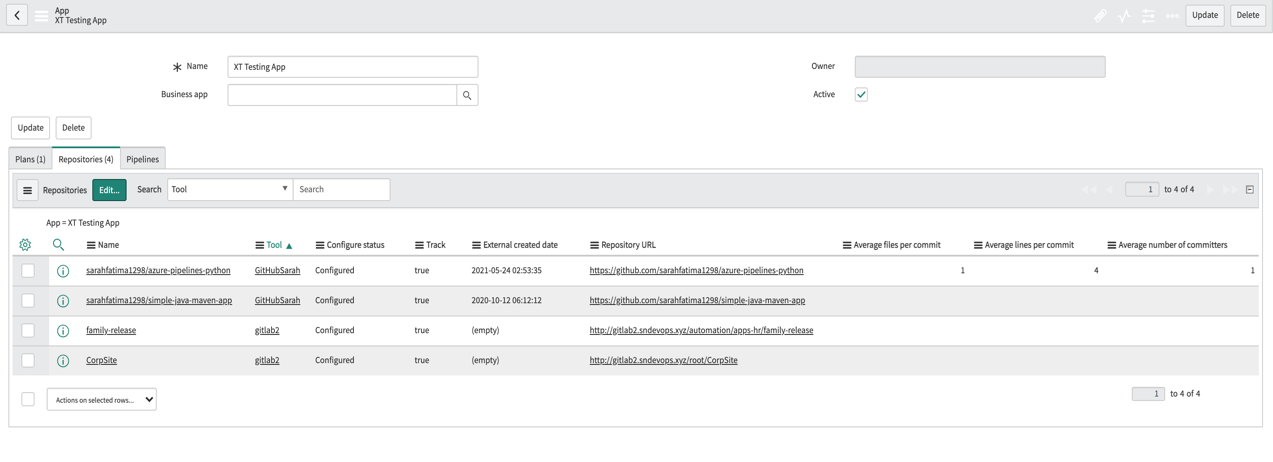


1. Navigate to DevOps > Apps & Pipeline > Apps
   1. Create a New App as below



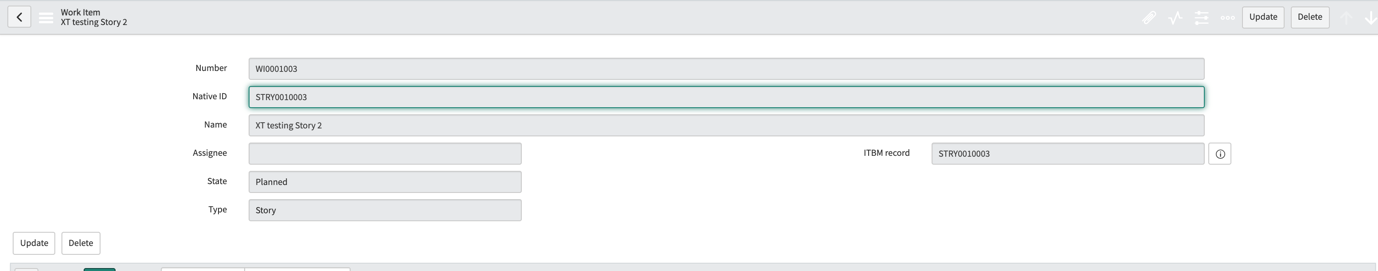
* 1. In Plans and Repositories related list click on *Edit* and add Plans and Repositories to associate with one another



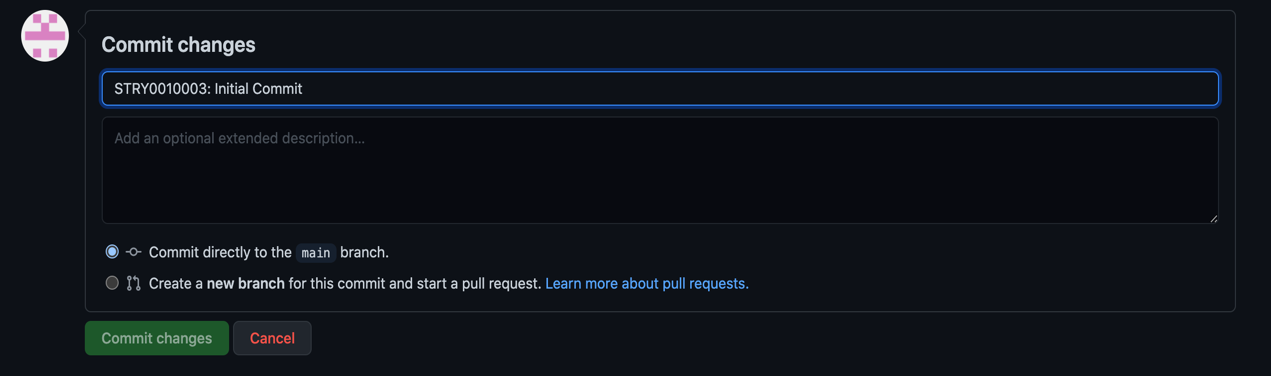


1. To map Commits to work items the commit message be of format:

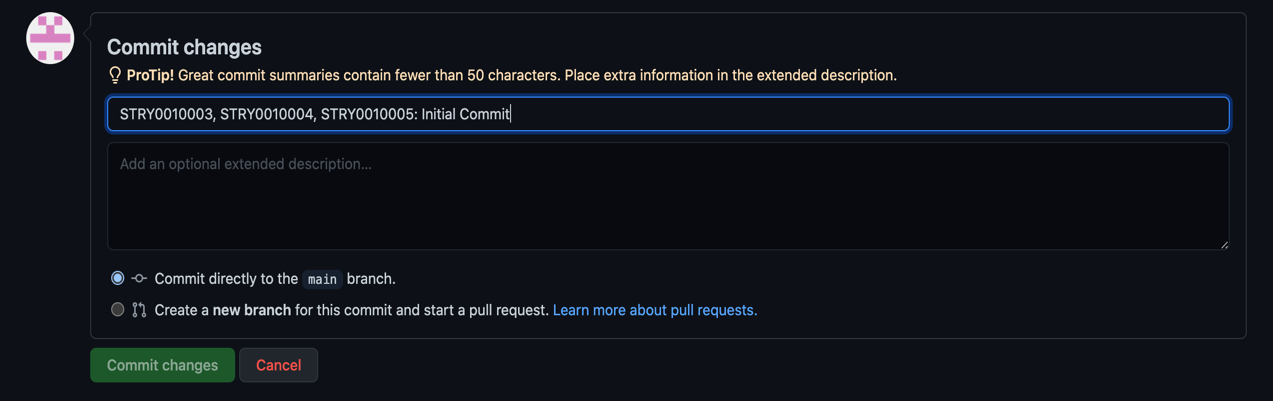
*Work Item 1 Native ID, Work Item 2 Native ID,…: Commit Message*



1. In this case commit message will be:
   1. To map single work item to commit

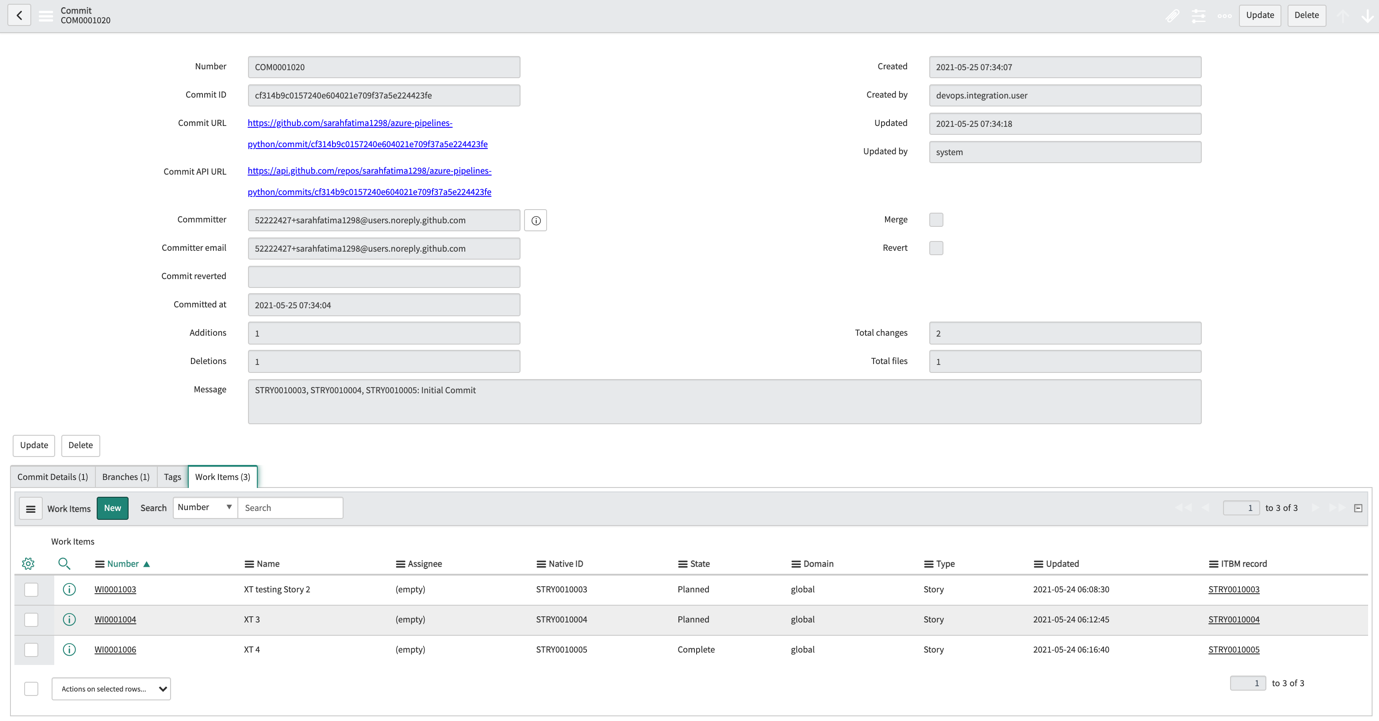


* 1. To map multiple work items to commit



**Expected Results:**

1. The commits will be mapped to work items which can be viewed under the Work Items related list for commit



## **Bulk processing of commits:**

**Pre-requisites:**

1. Gitlab Project, with a repository in it. The following gitlab instance can be used:

URL : <http://gitlab2.sndevops.xyz>

Credentials: root/DevOps1!

Personal Access Token : B957zKxr\_yuv2p1shJck

1. **ServiceNow DevOps Tool -** created to integrate with the above project. Below are the steps to achieve it:

**a. Navigate to DevOps > Tools > Create New and create a record.**

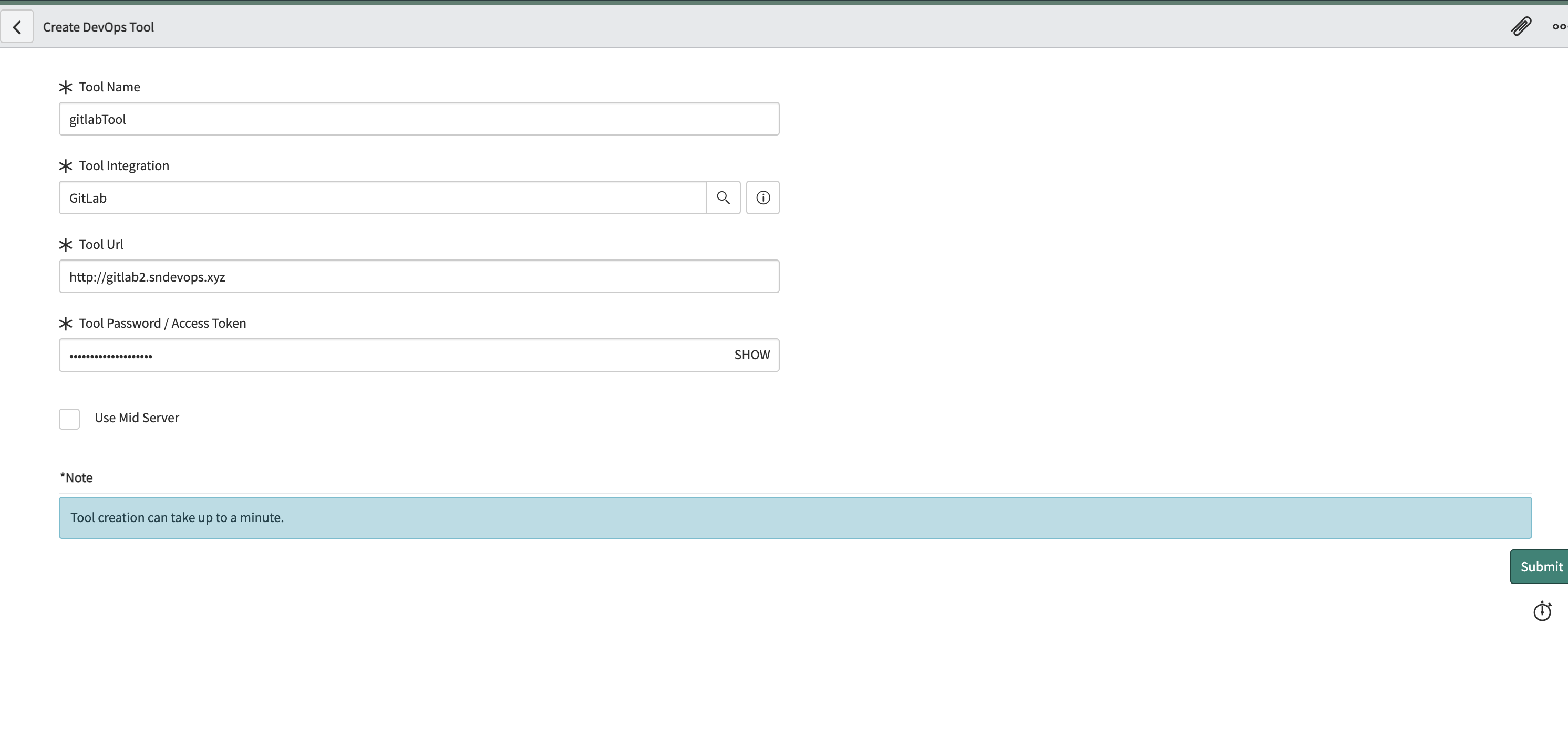
**b. Enter the following Details:**

**Tool Name – <Any Desired Name>**

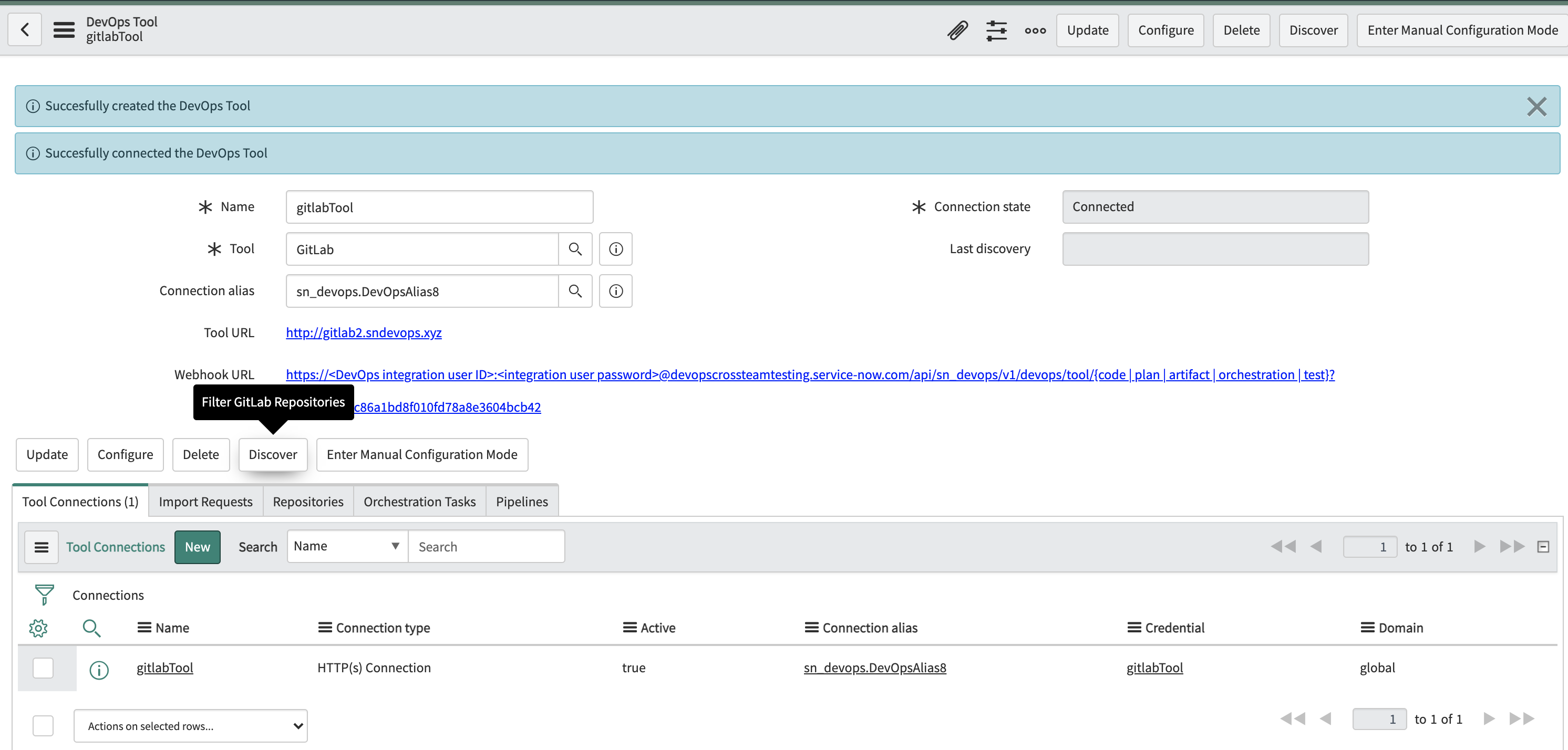
**Tool Integration - GitLab**

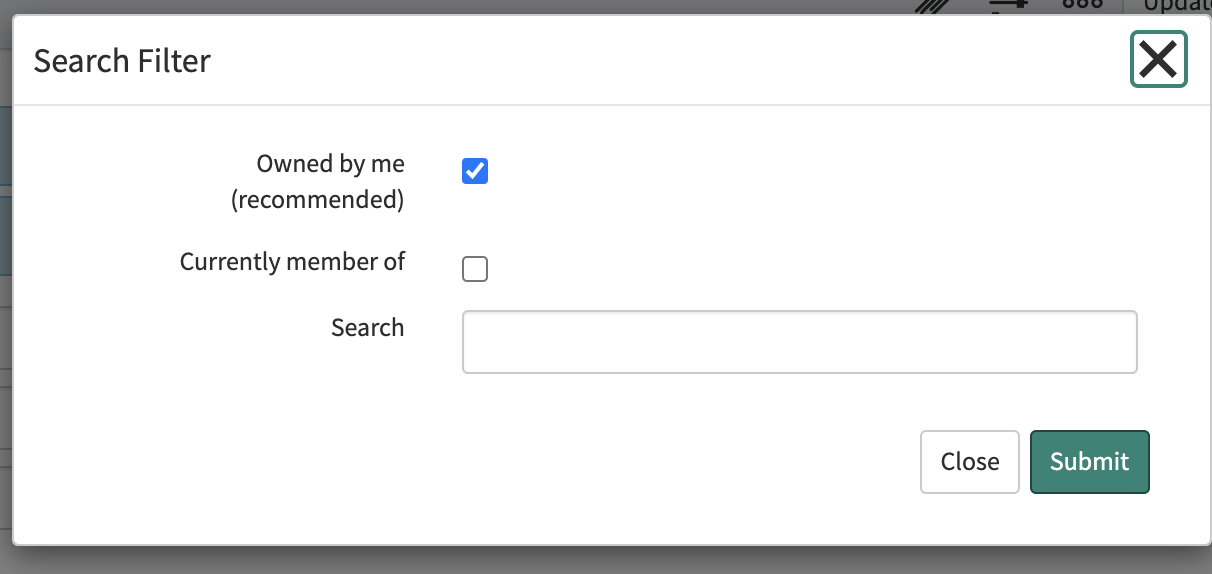
**Tool URL -** <http://gitlab2.sndevops.xyz>

**Personal Access Token – Created with DevOps Defined Scopes to Authorize.**



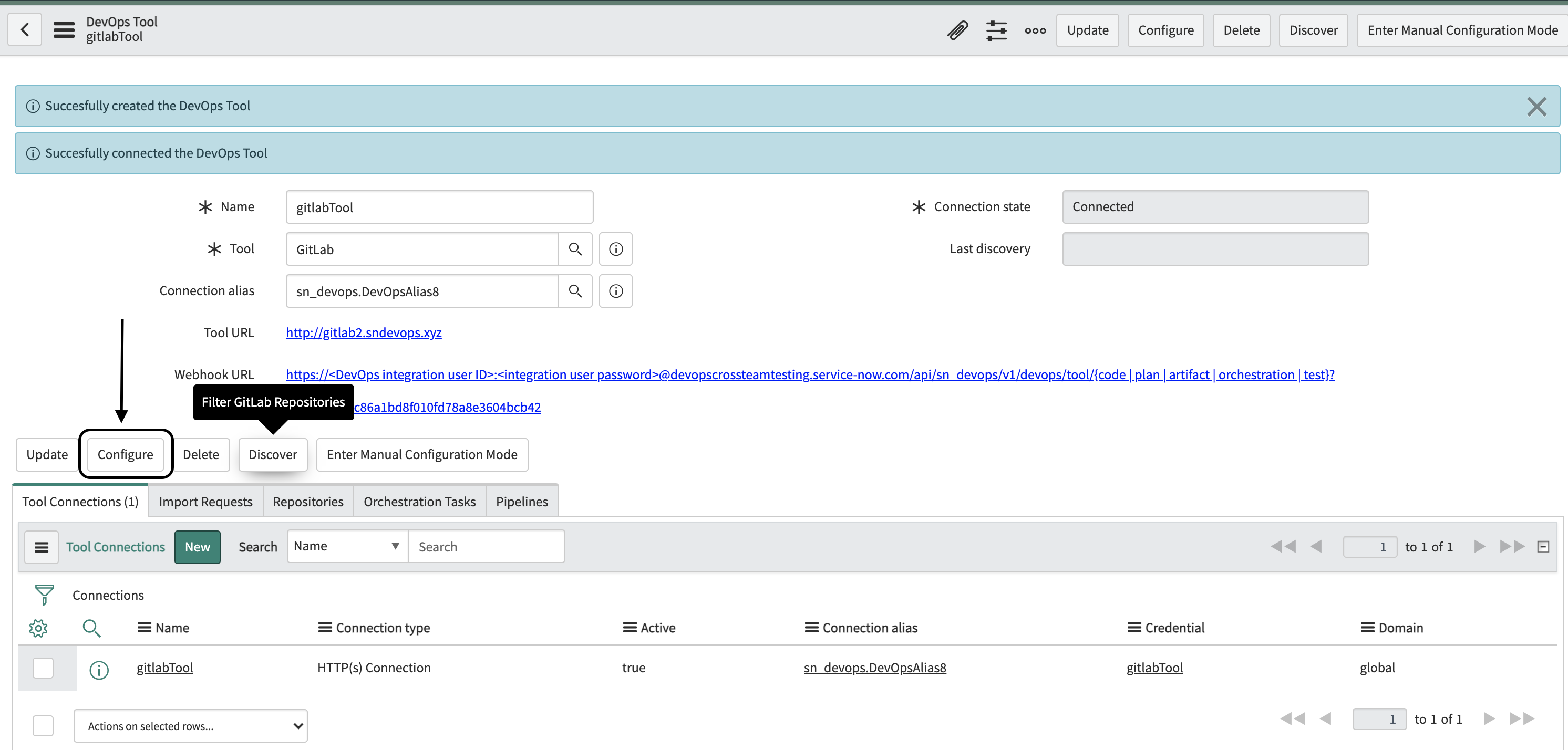
## **Once the Tool is connected, click on ‘Discover’ to import the plans, repositories, and pipelines owned by you.(This can be selected in the Filter popup that comes up after clicking on Discover.)**



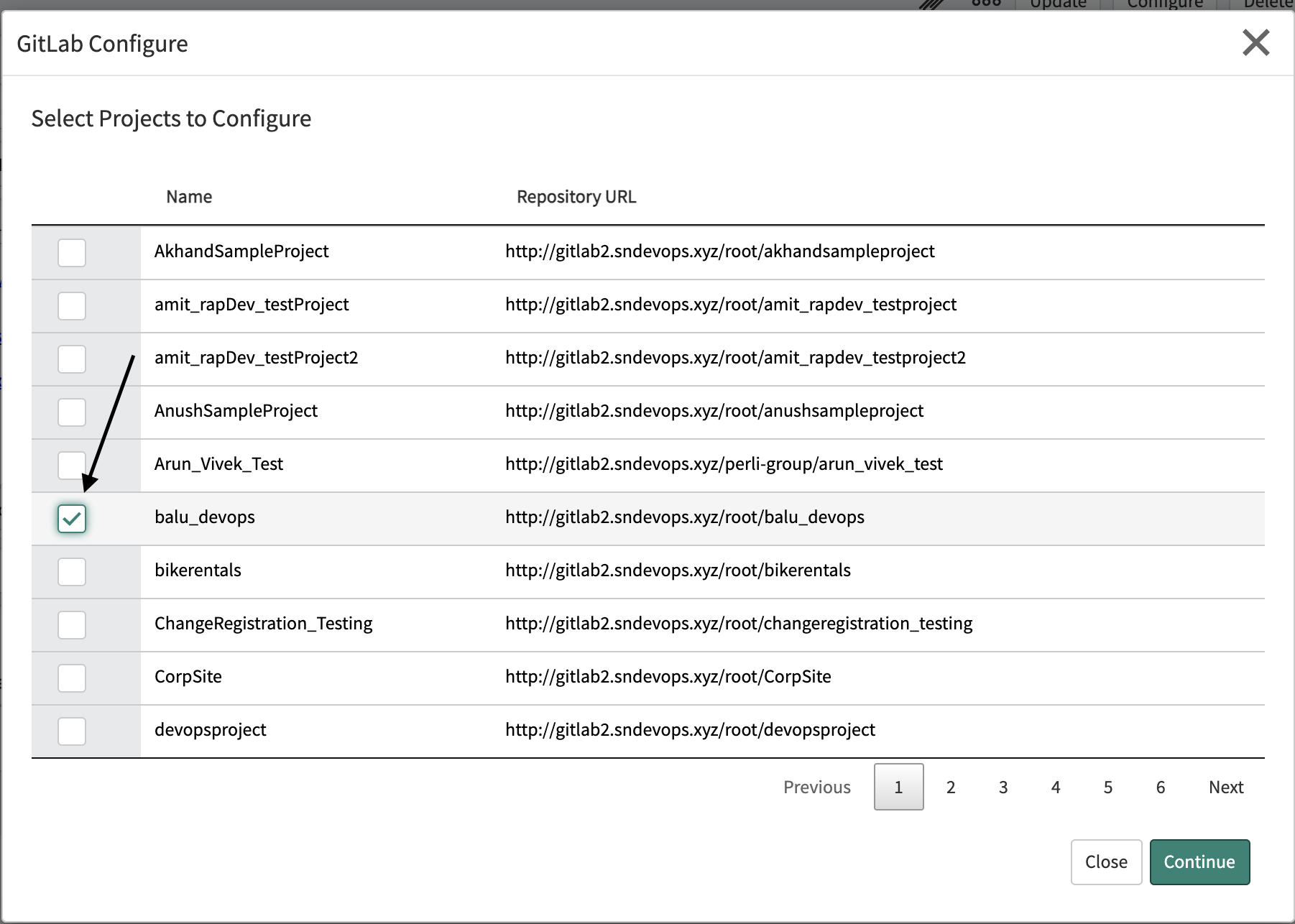


1. **Configure** – To create a Webhook. Once the tool is connected and Discover Action is successful click on ‘Configure’.

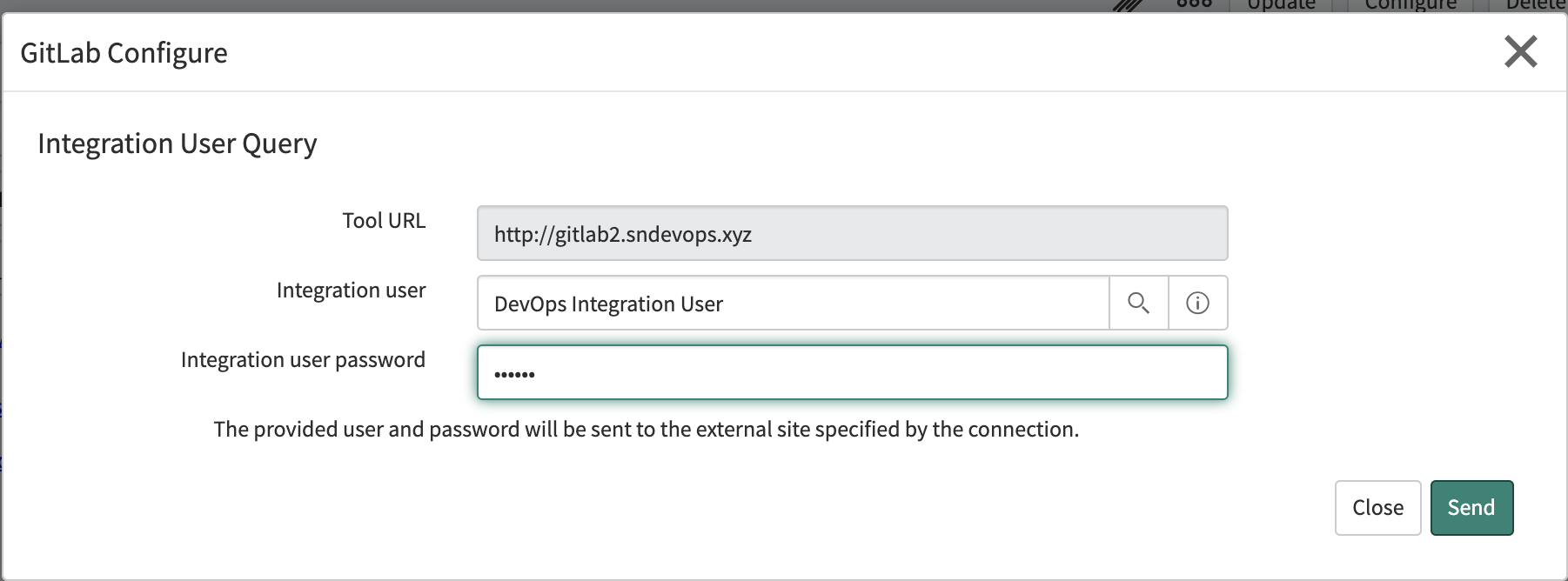
(Username: *devops.integration.user* and it’s password)



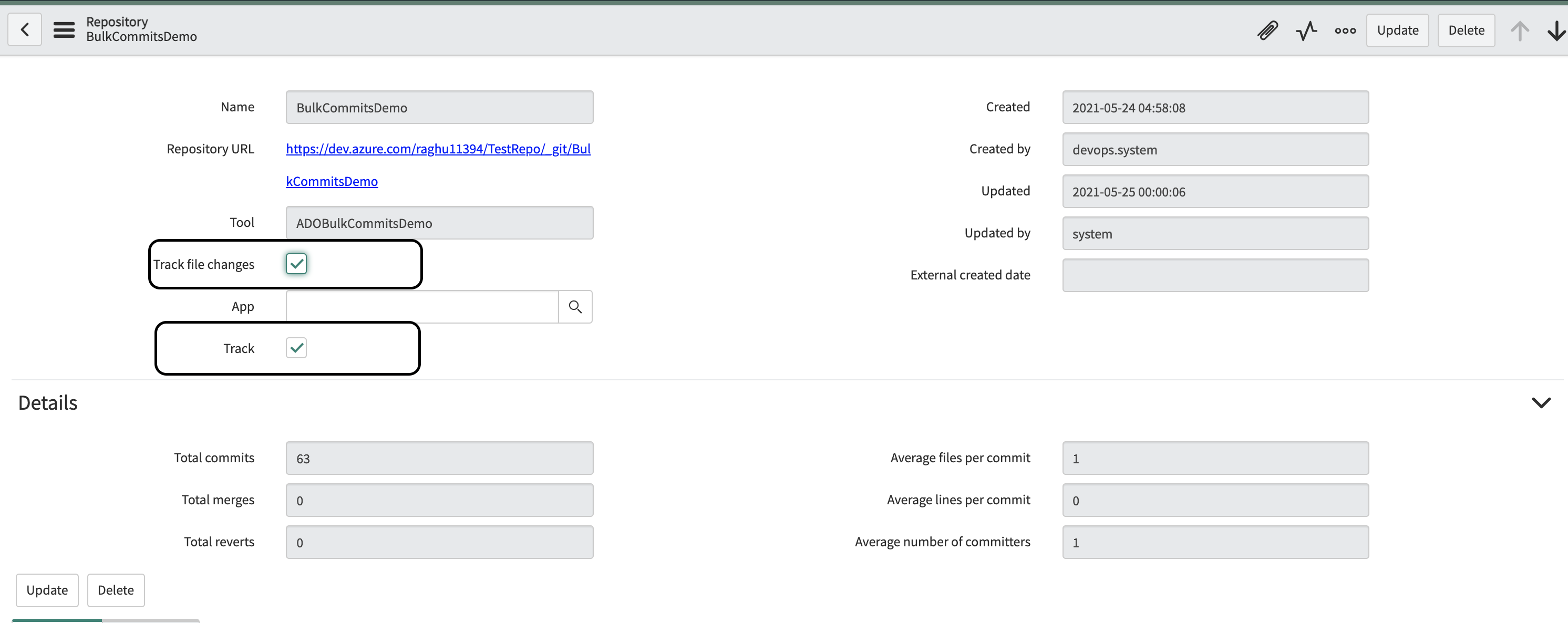
Now, select the repository to be configured in the popup that looks as below:



In the next screen, after clicking on Continue, enter the credentials of DevOps Integration User to Configure the repository selected.



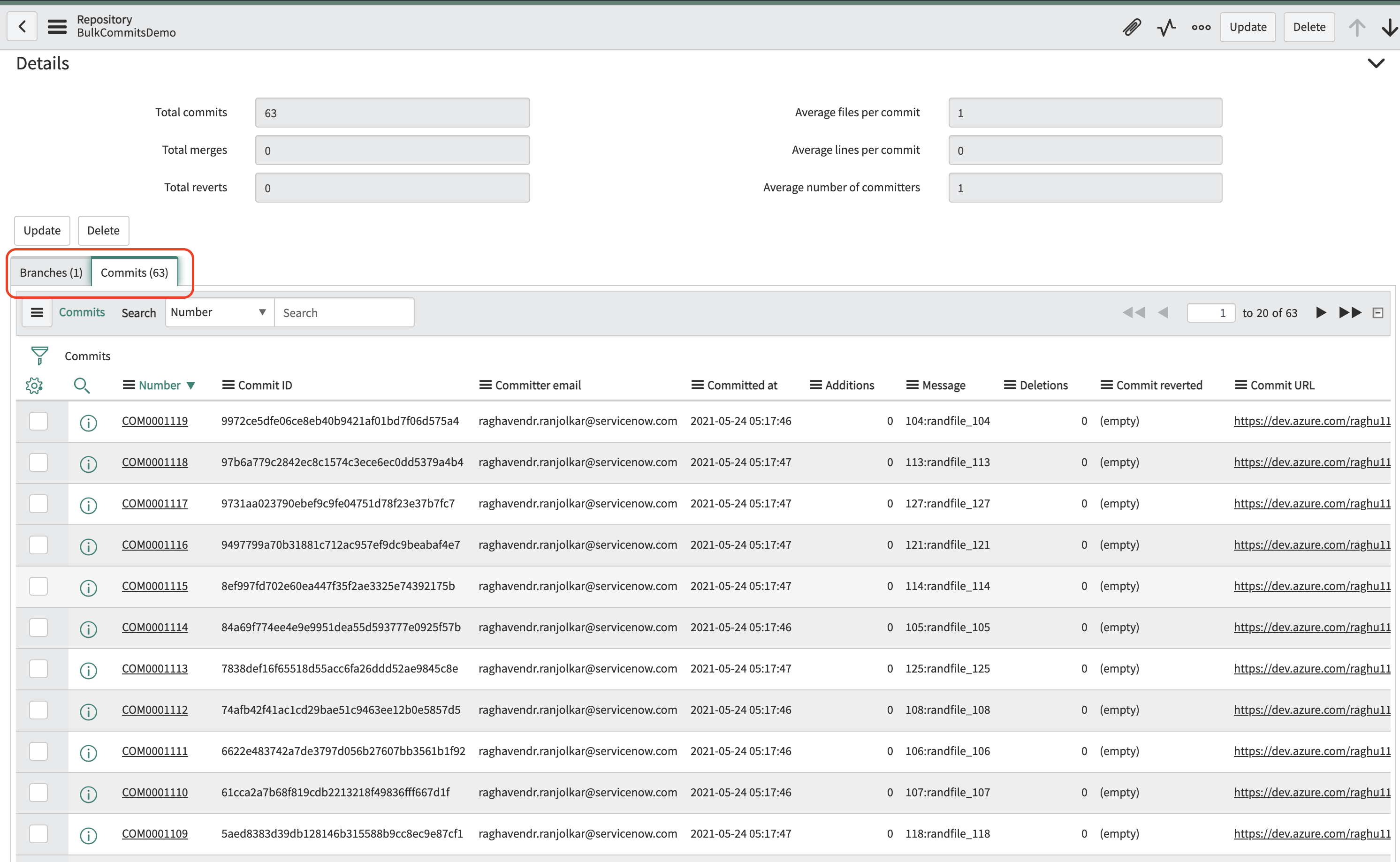
1. Enable tracking for the repository by checking the *Track* and *Track file changes* checkboxes for real-time tracking of the repository where Bulk Commits are to be created.



## **Creating Bulk Commits:**

1. The following script can be executed in the terminal to create Bulk Commits in the ADO Repository:
2. **string=""**  
   **x=1**  
   **j=00**  
   **while [ $x -le <CommitCount>]**  
   **do**  
    **string="touch randfile$((j+x)) && git add . && git commit -m '$((j+x)):randfile\_$((j+x))'"**  
    **echo $string**  
    **eval "$string"**  
    **x=$[$x+1]**  
   **done**
3. **git push**

## Once the Commits are Processed, they will be processed and will be persisted in ServiceNow instance.



## **Tags:**

**Prerequisites:**

1. Coding tool in *connected* state and repository with *track* and *track file changes* enabled and *configured.*

*Note: For the guide purpose we will use GitHub Repository created in initially*

1. *Basic Understanding of Tags in git.*

[*Basics Of Tagging in git*](https://git-scm.com/book/en/v2/Git-Basics-Tagging)

**Creation of Tags :**

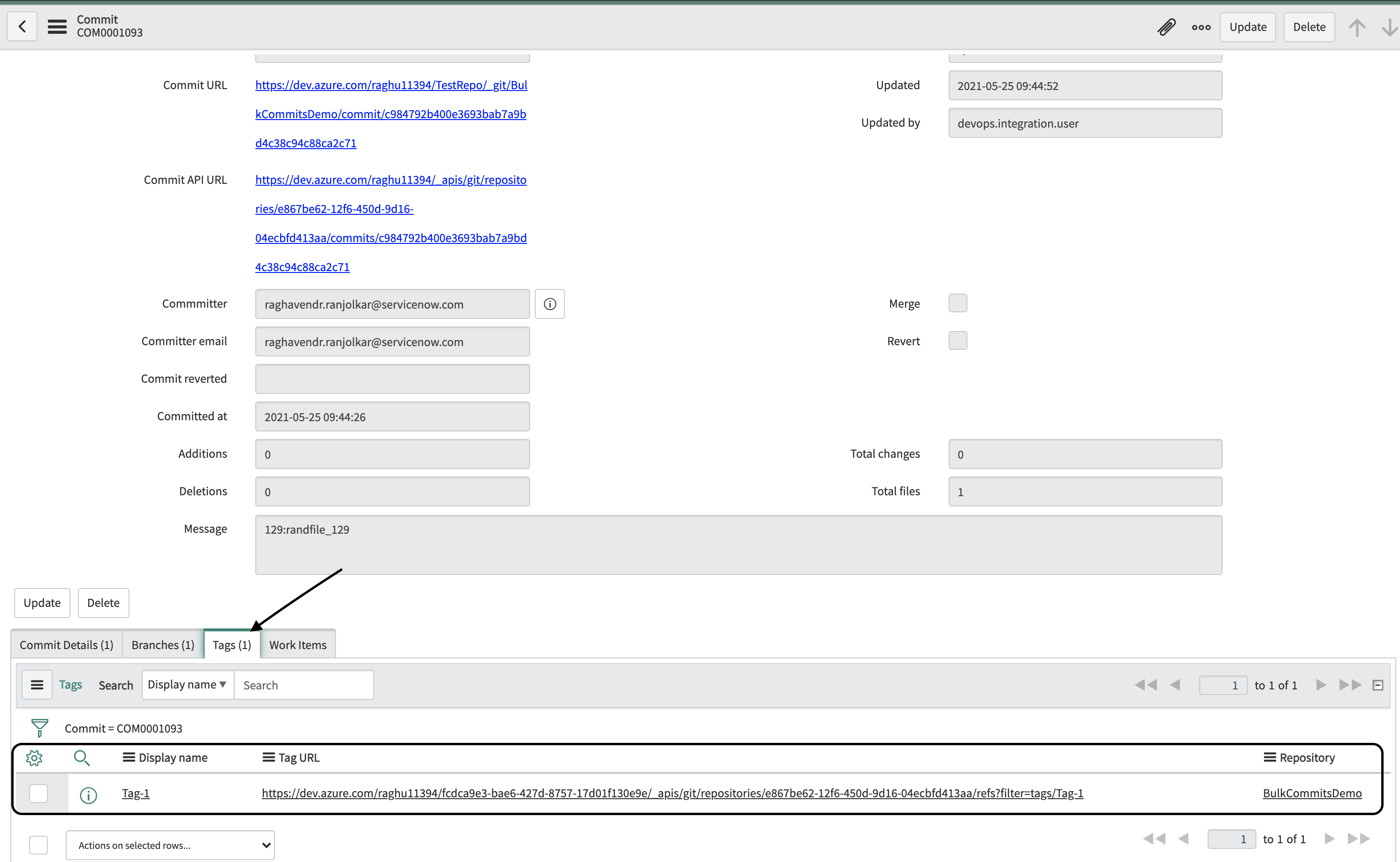
1. **Creation of Lightweight Tags:**

git tag Tag-1

git push origin Tag-1

1. The above step creates a lightweight tag and maps it to the last commit made in the GitHub Repository.

1. The same gets reflected in the servicenow instance as a related list in the Commit to which the tag is mapped to.



**Different Tag Use Cases:**

1. **Lightweight Tags (Can be created as above)**
2. **Annotated Tags**

git tag -a <TagName> -m "Tag-3 annotated"

git push origin <TagName>

1. **Tags to map to a past commit**

git tag -a <TagName> <commit id for C6> -m "Tag-4 for past commit"

git push origin <TagName>