

JIRA-Tosca Integration using Tosca Connect

For Tasktop Version 17.3 and above

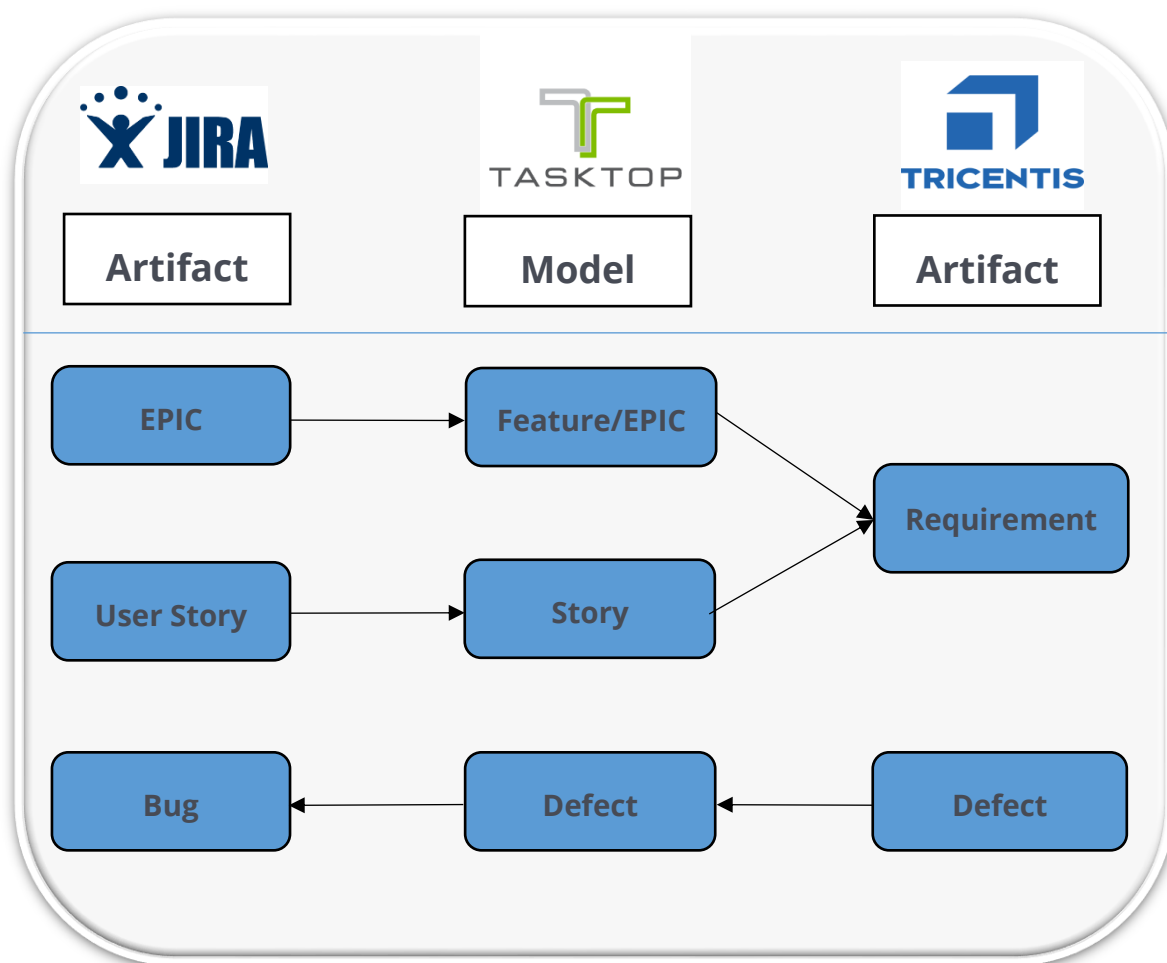
TRICENTIS

Table of Contents

1	Introduction	3
2	Prerequisite.....	4
3	JIRA-Tosca Integration Process	5
4	Hands-on Example	41

1 Introduction

JIRA-Tosca integration is a process where we sync the artifacts between both tools. For this integration, we use a third-party tool Tasktop Integration Hub. Tasktop provides the integration between several tools with its effective model feature. Following picture will explain how JIRA-Tosca integration happens using Tasktop.



EPIC :- It is essentially a large user story that can be broken-down into a number of smaller stories.

User Story :- A story or user story is a software system requirement that is expressed in a few short sentences, ideally using non-technical language.

Bug :- An issue could represent a software bug

In this integration, we are connecting EPIC and UserStory of JIRA with the Requirement session of Tosca so that whenever a new Epic or user story get created in JIRA it will automatically get linked to the Requirement of Tosca. Vice versa whenever a defect is created in Tosca it will automatically reflected as a new issue in JIRA.

2 Prerequisite

Following are the prerequisite from each tool.

1. JIRA

- JIRA should be installed locally or at a remote server with at list one project created inside it.
- A separate user should be available for the Tasktop.

2. Tasktop

- Tasktop should be installed locally or at a remote server with valid license.

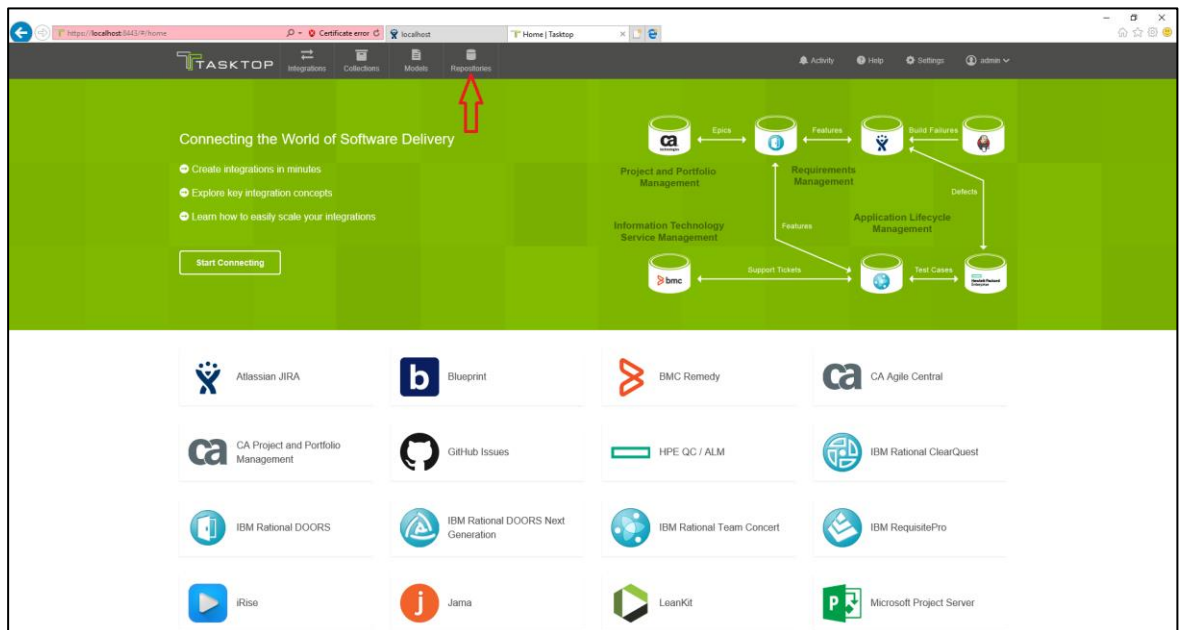
3. Tosca

- One multiuser workspace
- A dedicated user for Tasktop
- Rest API should be working properly [To communicate with workspace]
- One empty requirement set

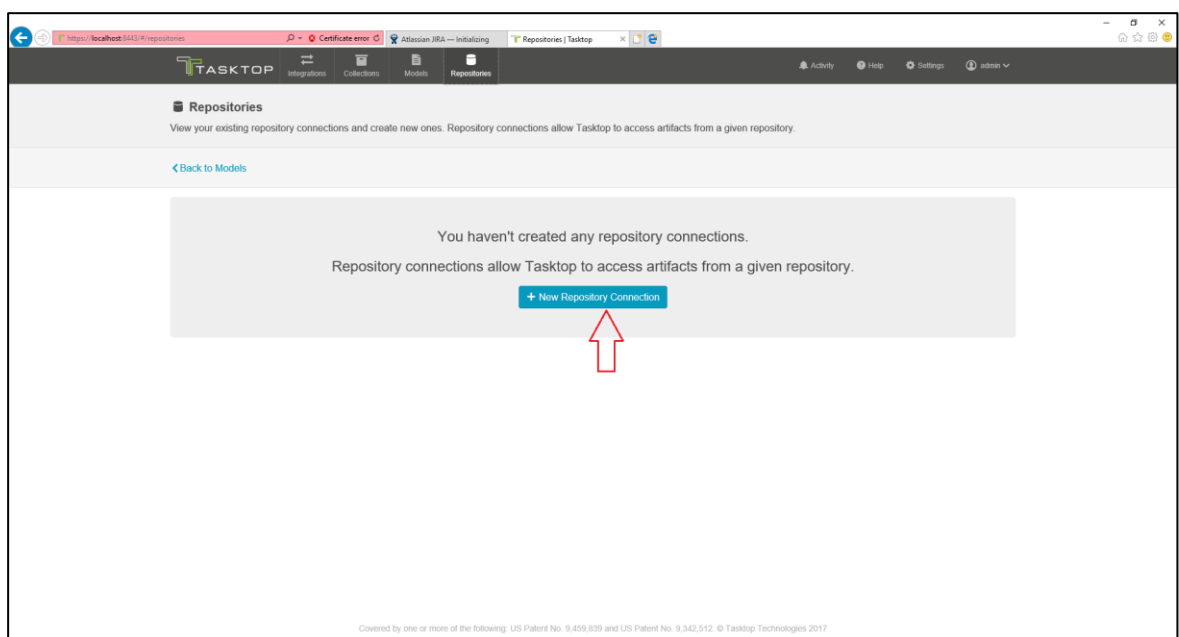
3 JIRA-Tosca Integration Process

To start the JIRA-Tosca integration first we require to login to the Tasktop. To create the integration, we use right to left approach in Tasktop, it means that first we create the Repositories then collections and finally we do the integrations. Following are the step by step process of integration after logging in to the Tasktop.

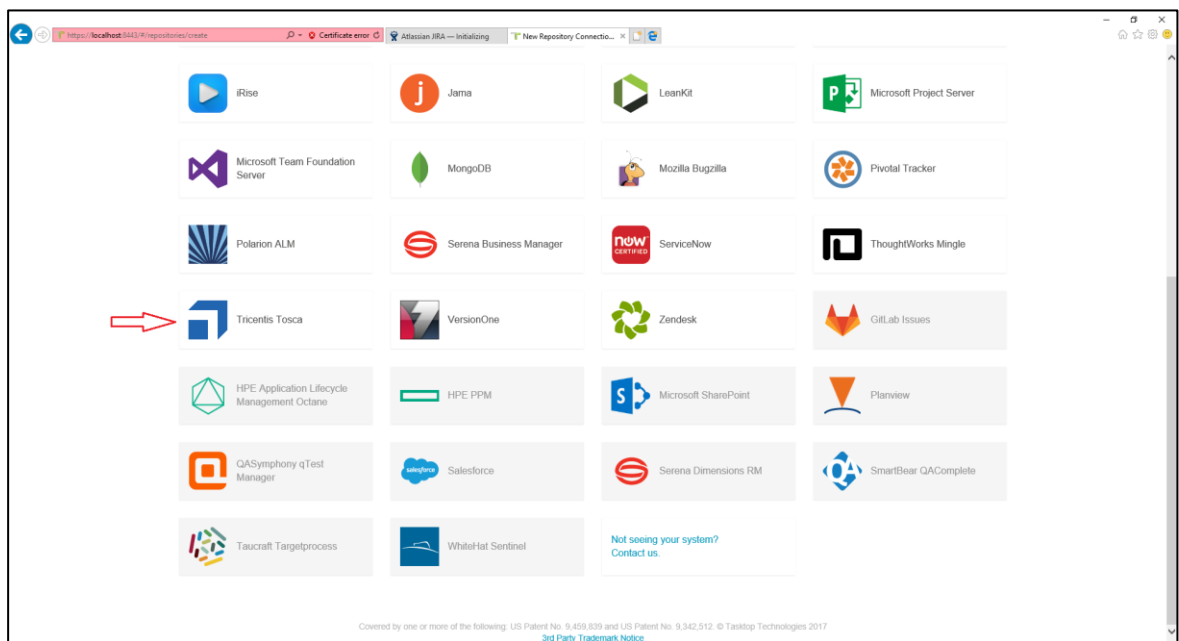
1. Click on the **Repositories** menu



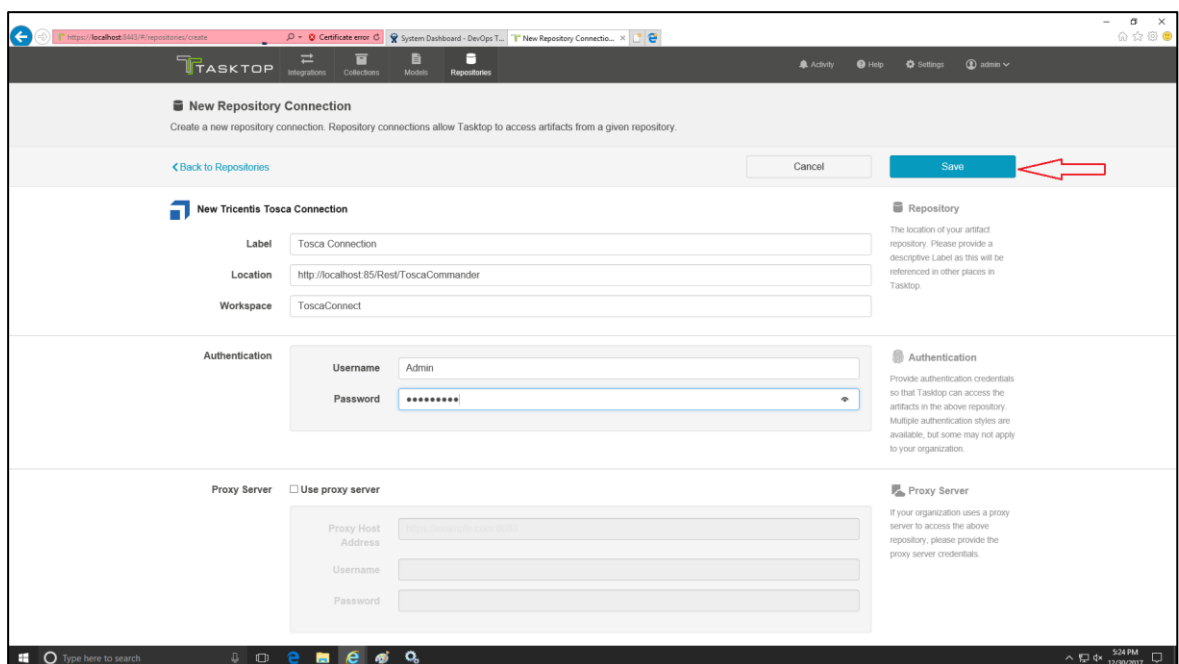
2. Repository is a connection between your tool and Tasktop, it allows Tasktop to communicate with your tool. Here click on **New Repository Connection** button



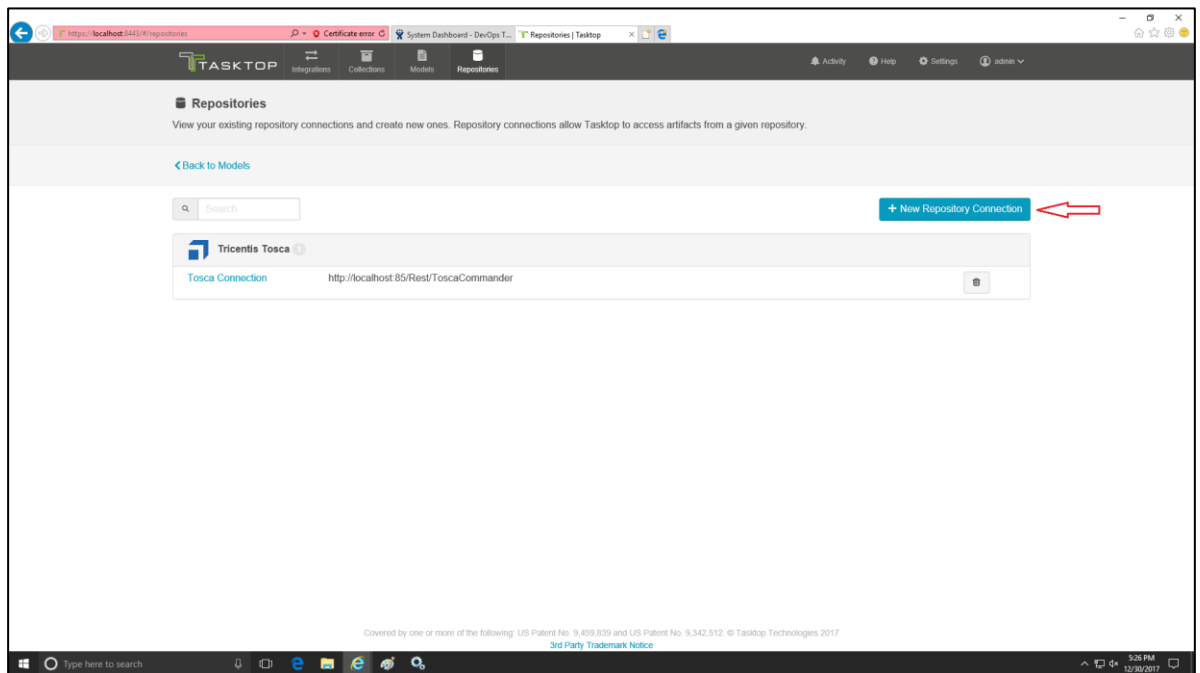
- First we will create the connection between Tosca and Tasktop, for that we will select the option **Tricentis Tosca**



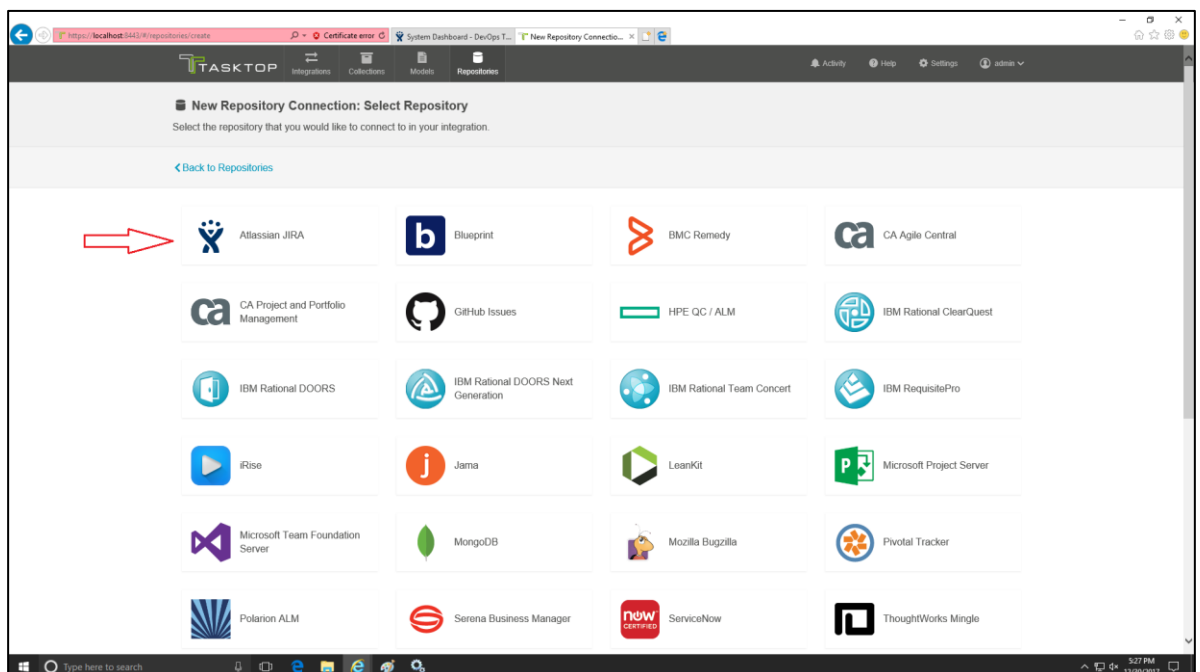
- Now we will provide the connection details,
Label: -Name of the connection
Location: -Provide the Rest API path [Make sure your Rest API is working properly]
Workspace: -Provide the workspace name
Username: -Username of workspace [Dedicated user of Tasktop]
Password: -Password of the user
 After this click on **Save** button and wait for establishing the connection, then click on **Done** button



- Now you can see that your connection is created, similarly we will create the connection for JIRA for that again click on **New Repository Connection** button



- Now we will select the option **Atlassian JIRA**



- Now provide the connection details of JIRA
 - Label:** -Provide the name for the connection
 - Location:** -Provide the URL of JIRA
 - Authentication:** -Select Standard Authentication from the drop-down list
 - Username:** -Provide the username of JIRA [Dedicated user of Tasktop]
 - Password:** -Provide the password of the user
 After this click on **Save** button and wait for establishing the connection, then click on **Done** button

New Repository Connection
Create a new repository connection. Repository connections allow Tasktop to access artifacts from a given repository.

[Back to Repositories](#) Cancel Save

New Atlassian JIRA Connection

Label: JIRA Connection

Location: http://localhost:8085

Zephyr access Key:

Zephyr shared secret:

Authentication: Standard Authentication

Username: Admin

Password: *****

Proxy Server: ☐ Use proxy server

Proxy Host Address:

Username:

Repository
The location of your artifact repository. Please provide a descriptive Label as this will be referenced in other places in Tasktop.

Authentication
Provide authentication credentials so that Tasktop can access the artifacts in the above repository. Multiple authentication styles are available, but some may not apply to your organization.

Proxy Server
If your organization uses a proxy server to access the above repository, please provide the proxy server credentials.

- Now you can see that both the repository has created

Repositories
View your existing repository connections and create new ones. Repository connections allow Tasktop to access artifacts from a given repository.

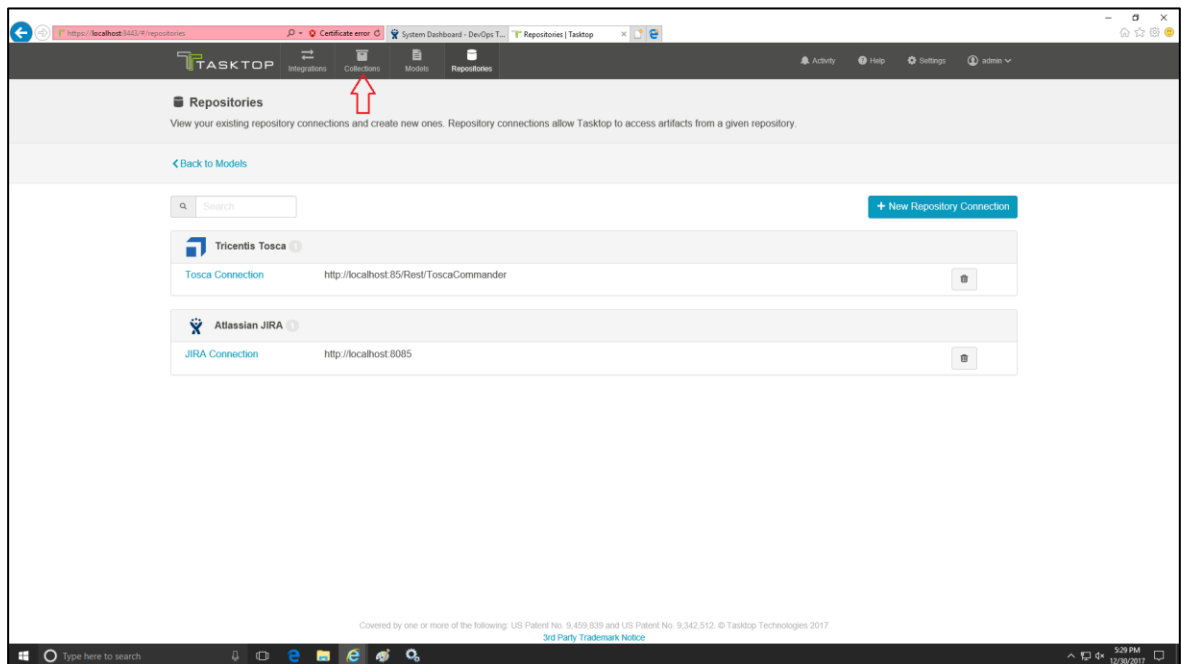
[Back to Models](#) + New Repository Connection

Search:

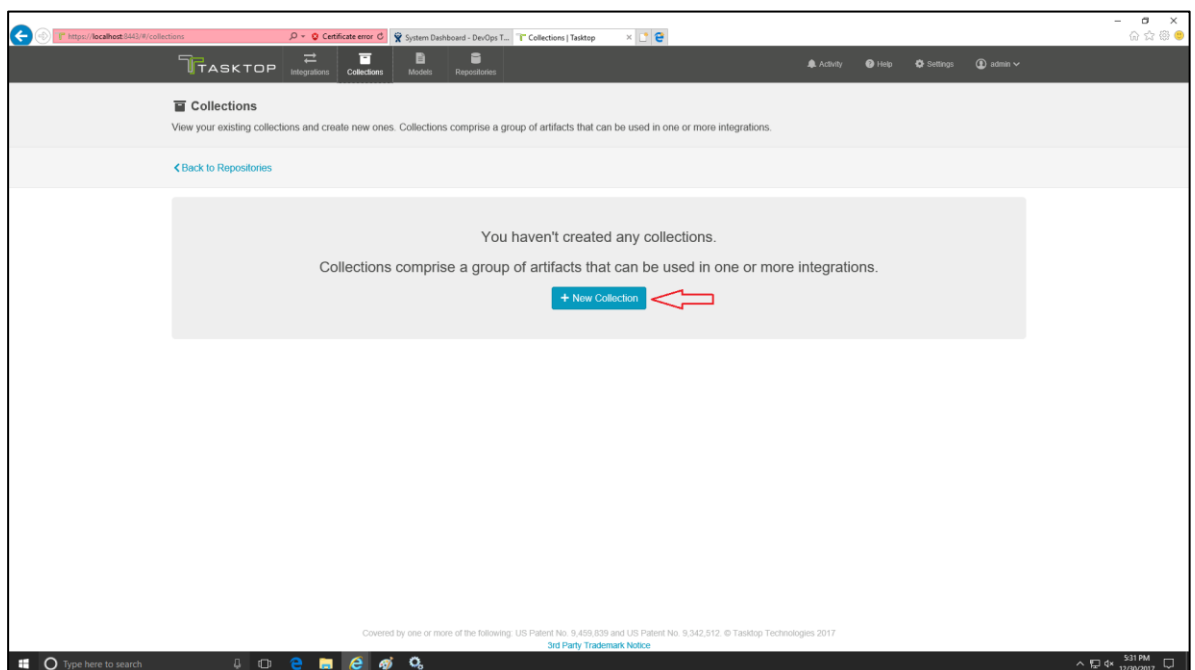
Tricentis Tosca	Tosca Connection	http://localhost:85/Rest/ToscaCommander	✕
Atlassian JIRA	JIRA Connection	http://localhost:8085	✕

Covered by one or more of the following: US Patent No. 9,419,879 and US Patent No. 9,342,512. © Tasktop Technologies 2017
3rd Party Trademark Notice

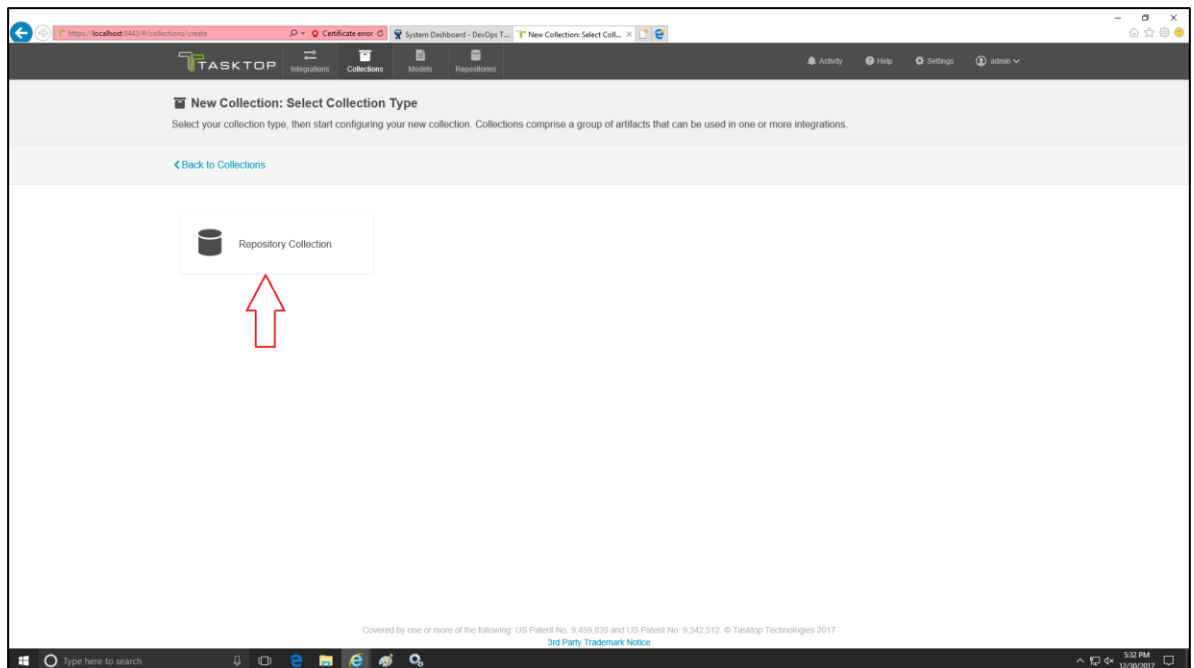
- Now click on **Collections** menu. Collections comprise a group of artifacts that can be used in one or more integrations.



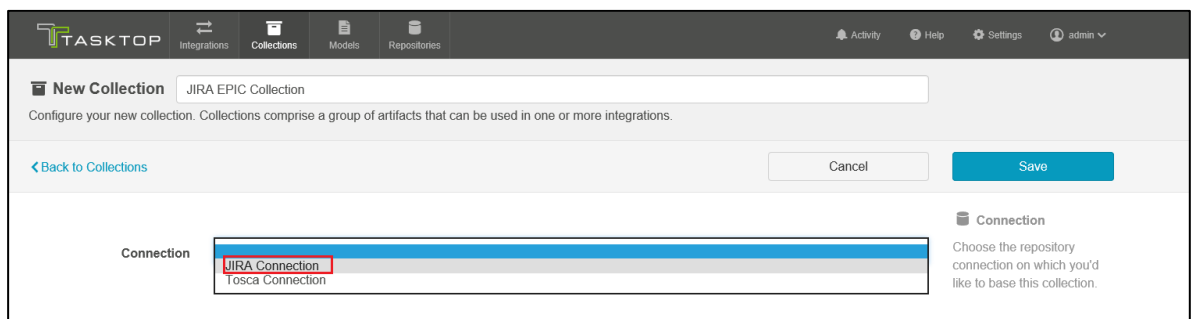
- Now we will require to create 6 collections 3 of JIRA [EPIC, UserStory, Bug] and 3 for Tosca [EPIC, UserStory, Bug]. For creating collections, we will click on **New Collection** button



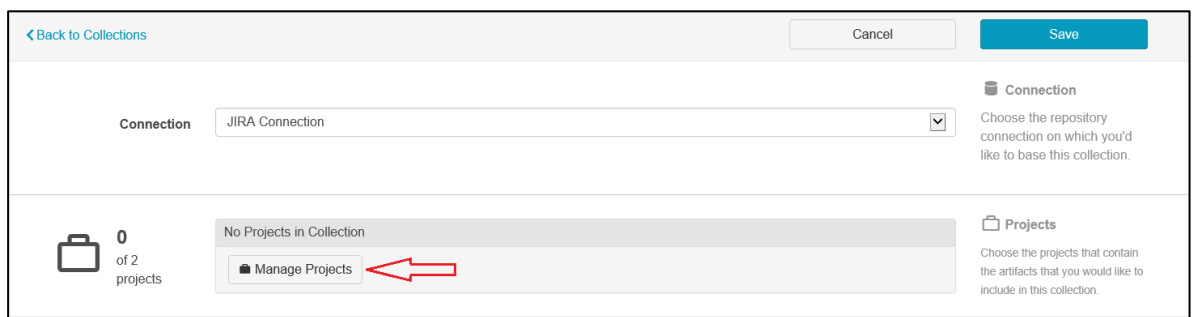
11. Now we will choose collection type as **Repository Collection**



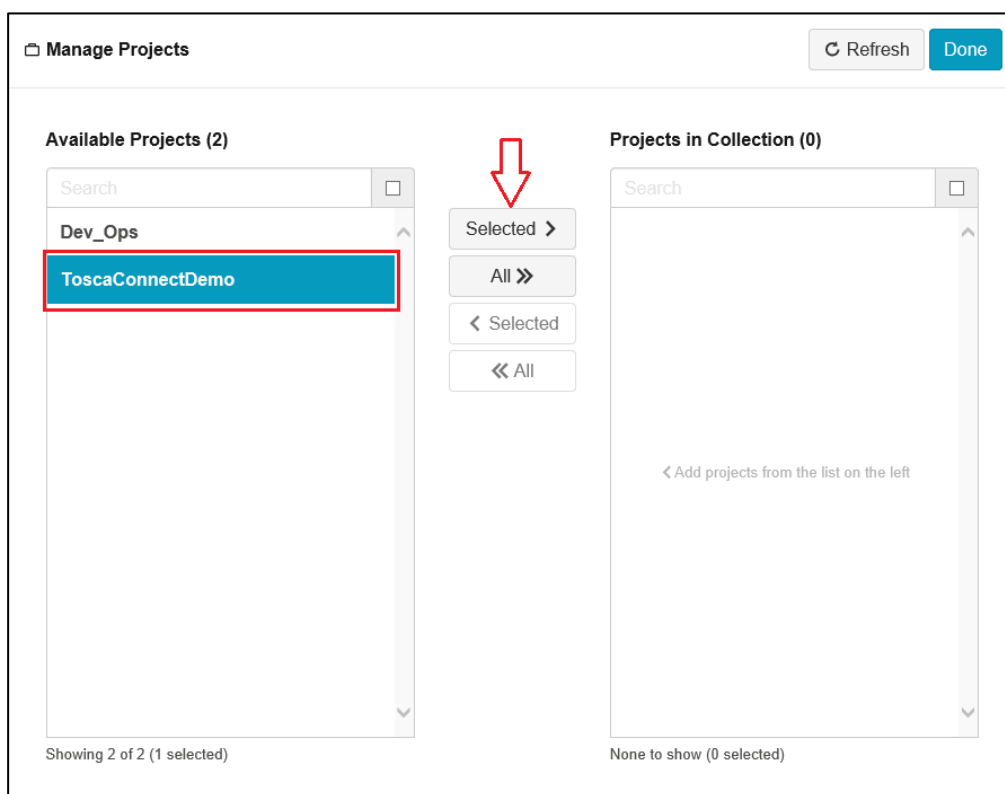
12. First we are creating the collection for JIRA EPIC, so we will provide the appropriate name and from connection we will select the **JIRA Connection**



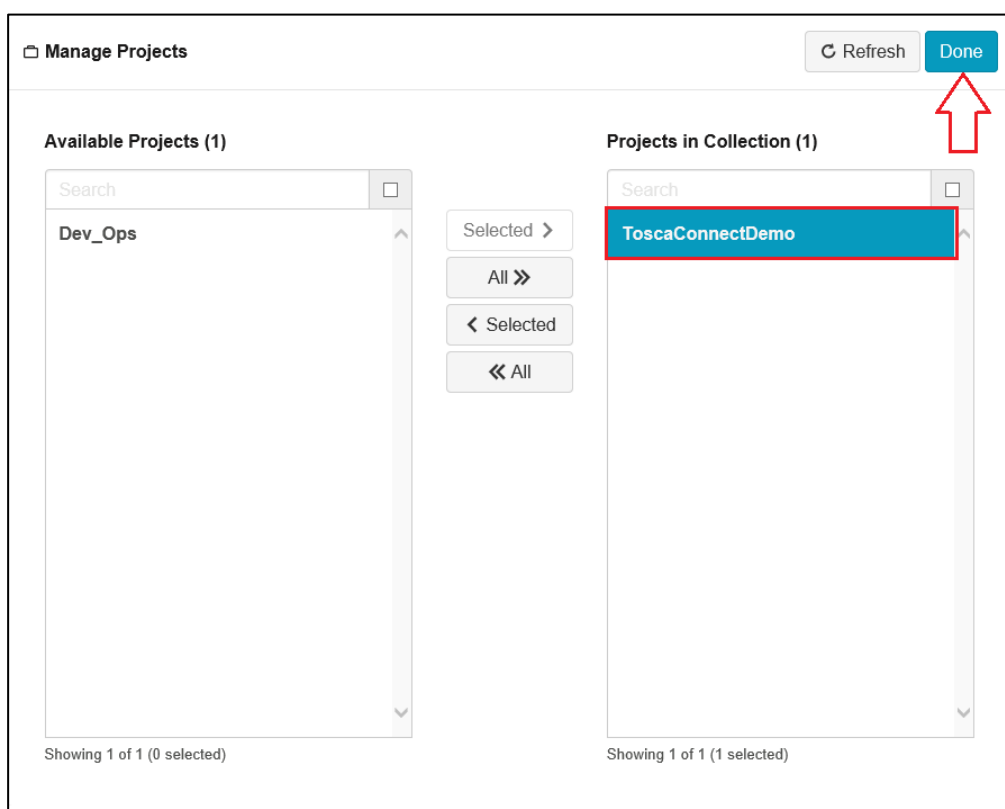
13. Now click on **Manage Projects**



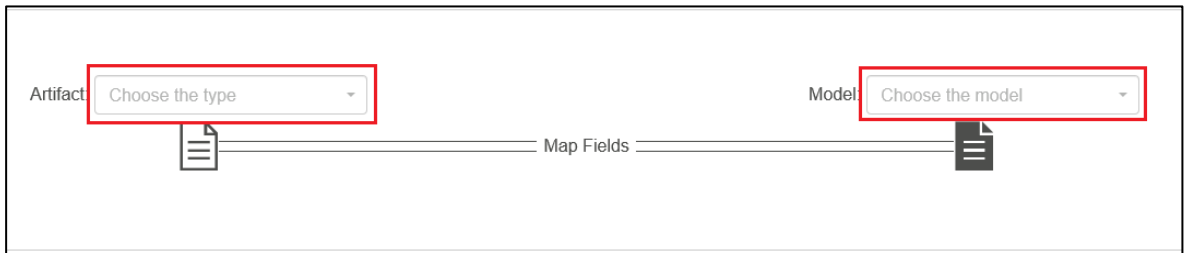
14. All the projects available inside JIRA will be displayed here, from the list choose your project and click on **Selected** button



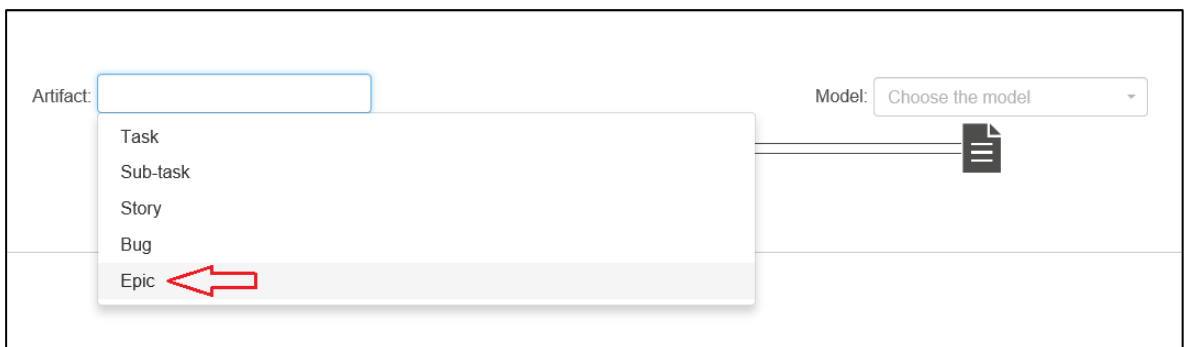
15. Once you select your project, it will move on the right-hand side of the window, now click on **Done** button



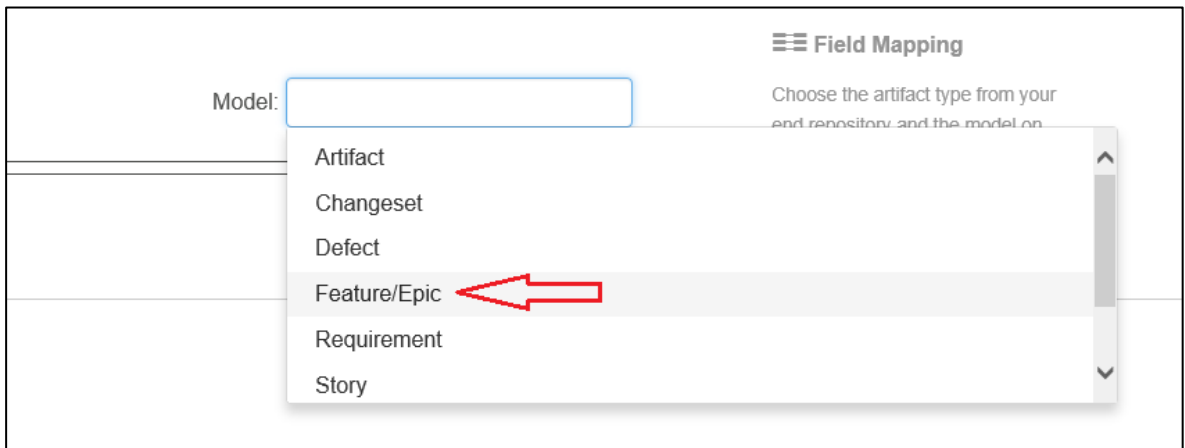
16. Once you select your project you will get two more options one for selecting Artifact of JIRA and second is for choosing the Model of Tasktop



17. Here we want to create the collection for JIRA Epic, so we will choose the Artifact **Epic** from the drop-down list.



18. After this we will choose the **Feature/Epic** Model of Tasktop



19. Now click on **Save** button

Tasktop Integrations Collections Models Repositories Activity Help Settings admin

New Collection JIRA EPIC Collection

Configure your new collection. Collections comprise a group of artifacts that can be used in one or more integrations.

[Back to Collections](#) Cancel Save

Connection JIRA Connection

Choose the repository connection on which you'd like to base this collection.

Projects 1 of 2 projects

ToscaConnectDemo

[Manage Projects](#)

Choose the projects that contain the artifacts that you would like to include in this collection.

Field Mapping

Artifact: Epic Model: Feature/Epic

Map Fields

Choose the artifact type from your end repository and the model on which you'd like to base this collection. Map between the fields on each side to form the field mapping for this collection.

20. Now your collection is created and you can see different type of sub artifacts are mapped

Atlassian JIRA JIRA Connection http://localhost:8085

Connection

This is the repository connection on which this collection is based.

Projects 1 of 2 projects

ToscaConnectDemo

[Manage Projects](#)

Add or remove projects from the collection. Tasktop can access the artifacts in these projects when this collection is used in one or more integrations.

Field Mapping

Artifact: Epic Model: Feature/Epic

Map Fields

11 of 42 fields mapped 11 of 16 fields mapped

Specify how the fields on your repository artifact align to fields in your model. The mapping between the fields on each side forms the field mapping for this collection.

Relationship Specification

Inbound Artifacts Relating Model

Configure Relationships

0 of 19 relationships mapped 0 of 5 relationships mapped

Specify how your repository artifact aligns to the relationship types in your model.

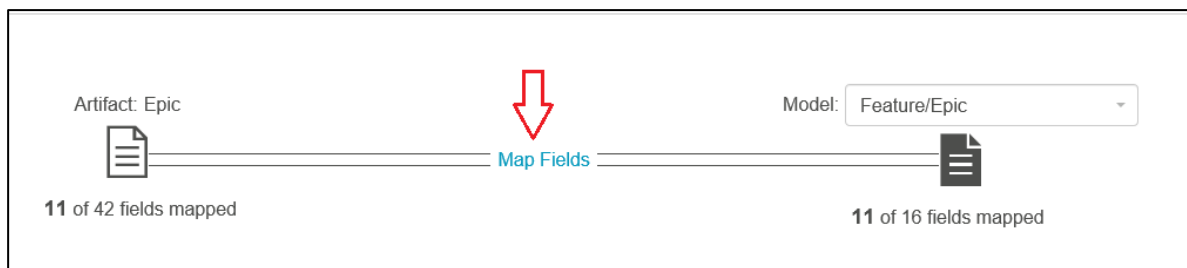
Person Resolution Strategy

Inbound Person Person Model

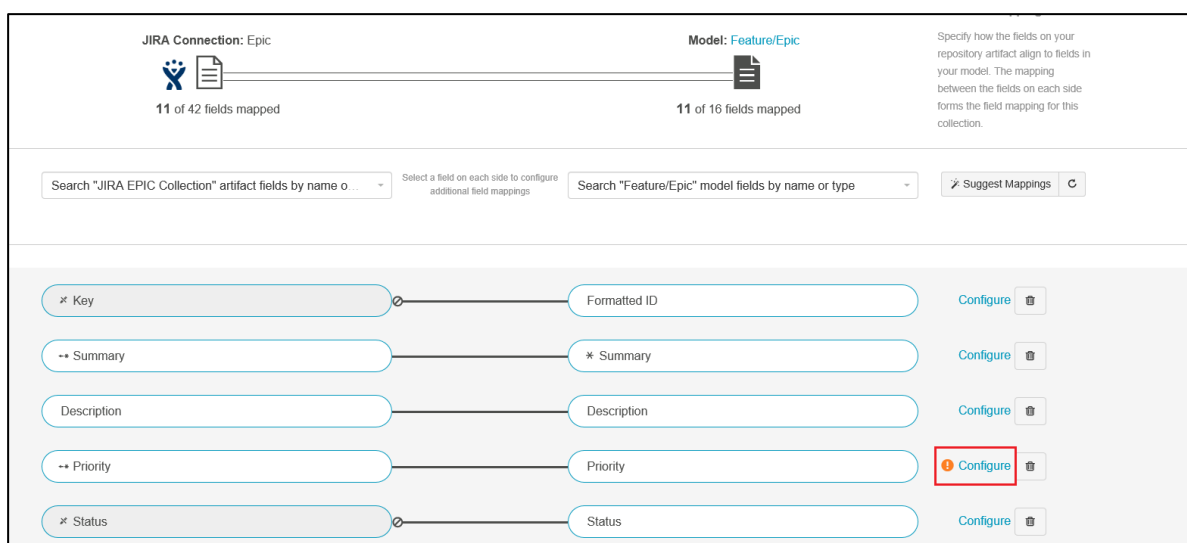
Person Resolution Strategy

Specify the strategy you'd like to use to reconcile persons between your repository artifact and your model.

21. Now click on **Map Fields** link



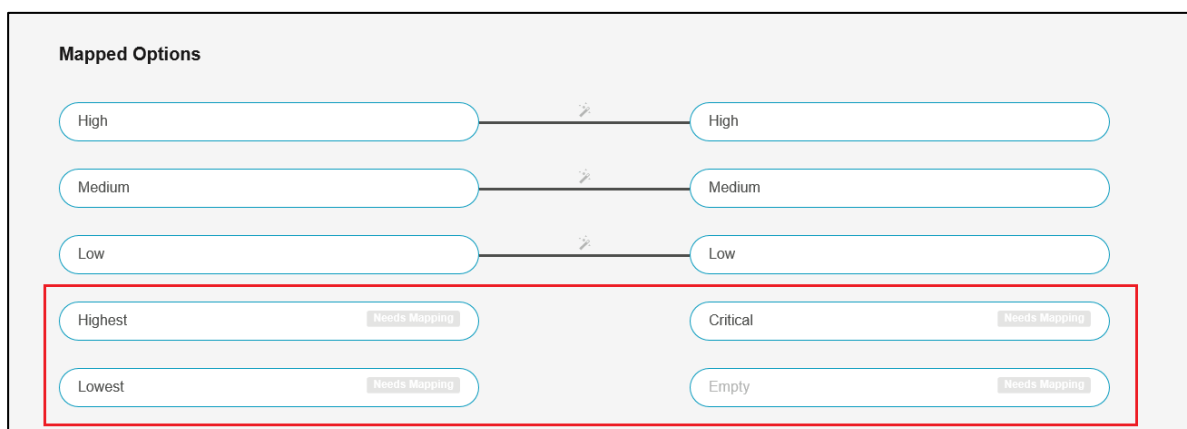
22. Now you can see the error mark at Priority field which means that it is not mapped properly



23. Now click on **Configure** link to see the configuration



24. Here we can see that Highest and Lowest priority are not mapped. The reason behind that is we do have a priority Highest and Lowest available in EPIC schema of JIRA but we don't have these two priorities in Feature/Epic model of Tasktop.



25. Now we are required to connect this priority type with the available priority type in Feature Epic model, for that first we will select **Highest** from JIRA

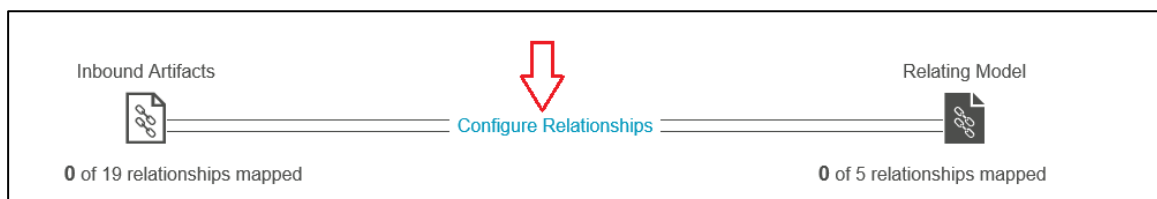
26. Next we will map this with High priority of Tasktop, for that we will choose **High** from Tasktop, then we will click on **Connect** button to map both the fields

27. In similar way, we will map Lowest with Low, after this click on **Save** button and then click on **Done** button

28. Now you will see all the fields are mapped properly

* Key	Formatted ID	Configure	
** Summary	* Summary	Configure	
Description	Description	Configure	
** Priority	Priority	Configure	
* Status	Status	Configure	
* Sprint	Sprint	Configure	
Reporter	Created By	Configure	
Assignee	Owner	Configure	
* Created	Created	Configure	
* Updated	Modified	Configure	
* URL	URL	Configure	

29. Now click on **Configure Relationship** link



30. Here choose **Sub Task(Relationships)** from JIRA Connection

[Back to Collection](#)

JIRA Connection: Epic

0 of 19 relationships mapped

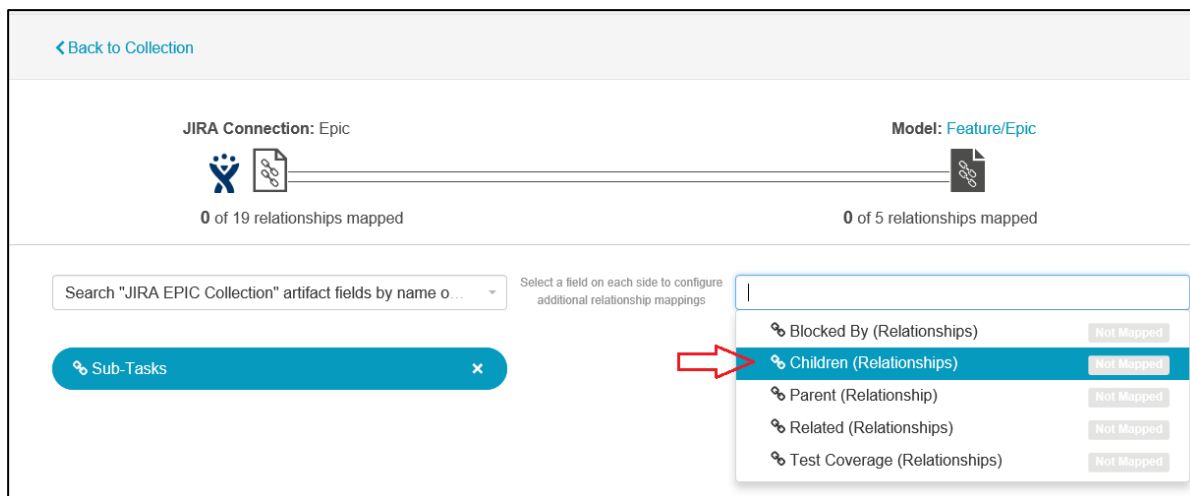
Model: Feature/Epic

0 of 5 relationships mapped

- * Issue ID (String) Not Mapped
- * Rank (String) Not Mapped
- % relates to (in) (Relationships) Not Mapped
- % relates to (out) (Relationships) Not Mapped
- * % Sub-Tasks (Relationships) Not Mapped
- Web Links (Web Links) Not Mapped

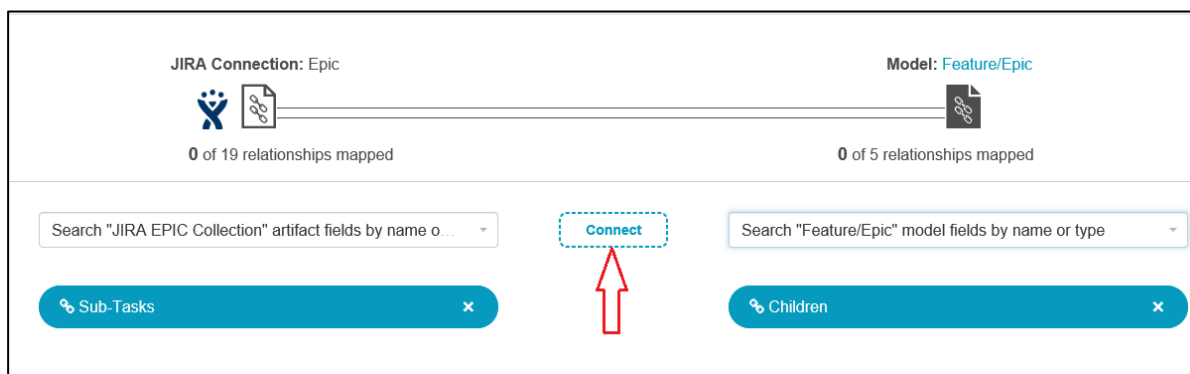
Select a field on each side to configure additional relationship mappings

31. Now choose **Children(Relationships)** from model

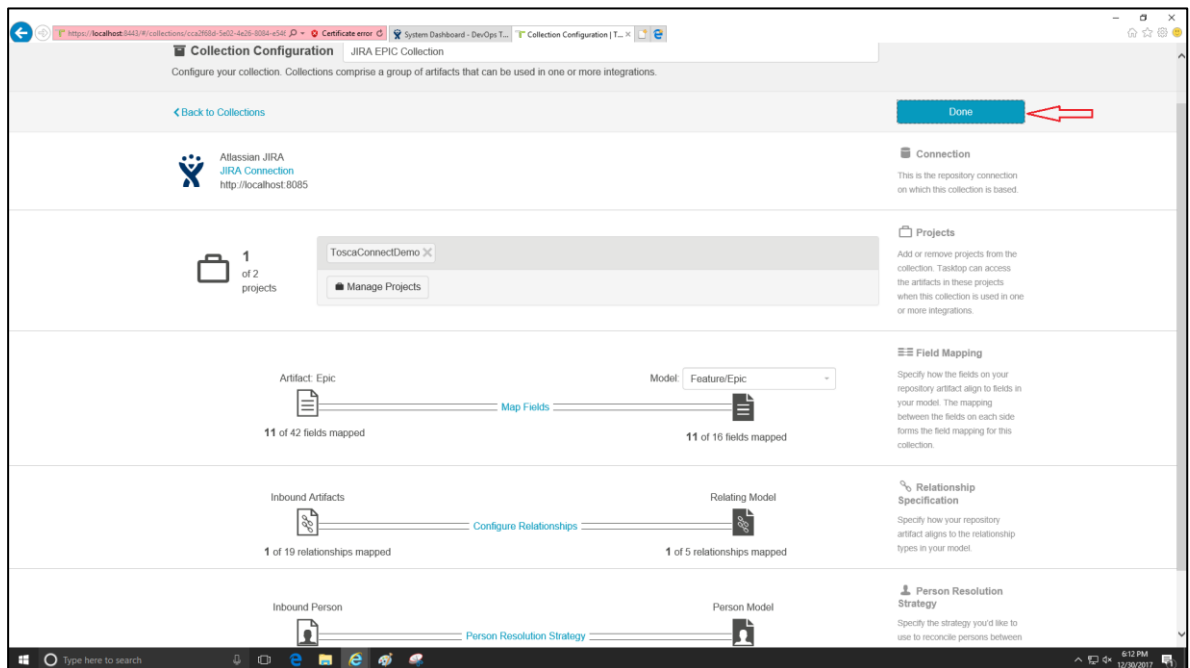


32. Now click on **Connect** button, now your relationship is created. After this click on **Save** and then **Done** button.

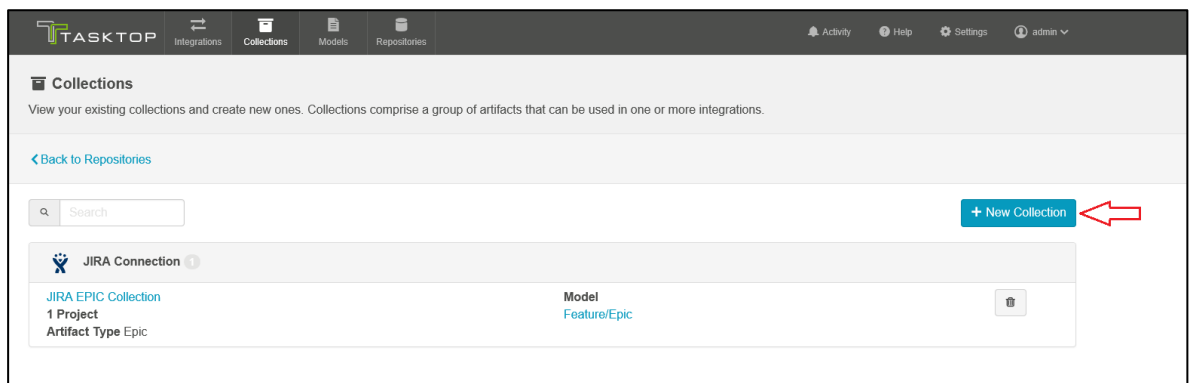
Here we are telling the Tasktop that that this Epic may contain the sub-task also which can be considered as children of the epic.



33. Now our collection is created properly, after this click on **Done** button to complete the collection

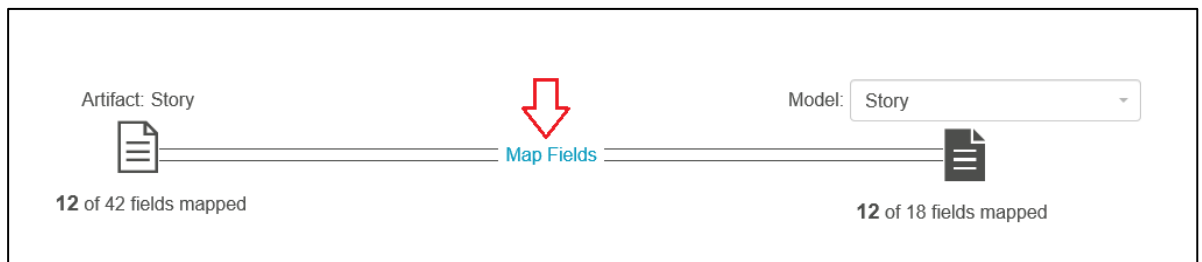


34. JIRA EPIC Collection is ready, in similar way we will create the remaining collection, for that we will again click on **New Collection** button



35. Now we will create collection for JIRA UserStory for that we will choose **Story** artifact from JIRA and **Story** model from Tasktop

36. Here we will click on **Map Fields** link



37. We will again click on **configure** on priority field

38. In same way, we will map the priority field for story

Field Mapping: JIRA STORY Collection
View and manage the field mapping for this collection. The field mapping specifies how fields from your repository artifact map to fields in your model.

[Back to Field Mapping](#) Cancel Save

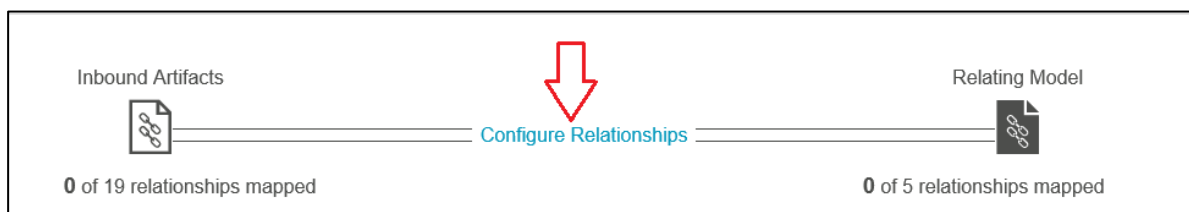
Priority ————— Priority

Transform Single Select Copy Transform Single Select Copy

Mapped Options

Highest , [High]	—————	High	
Lowest , [Low]	—————	Low	
Medium	—————	Medium	
		Critical	Needs Mapping
		Empty	Needs Mapping

39. Now we will configure the **Relationship**



40. Here we will connect Epic Link to Parent. We will save this collection and our JIRA User story collection is ready.

JIRA Connection: Story 0 of 19 relationships mapped

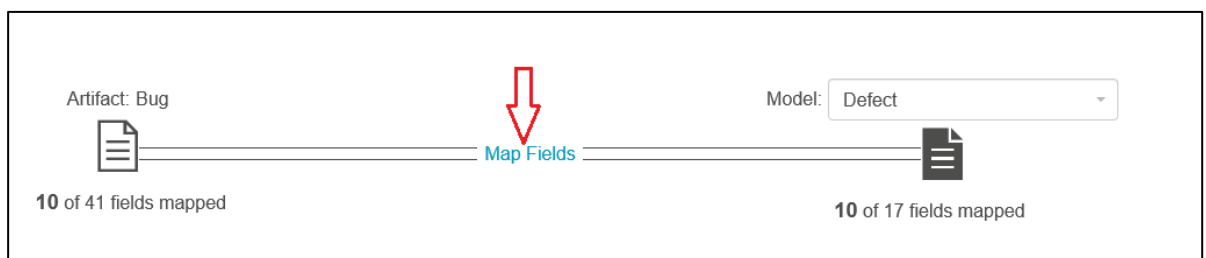
Model: Story 0 of 5 relationships mapped

Search "JIRA STORY Collection" artifact fields by nam... Connect Search "Story" model fields by name or type

Epic Link × Parent ×

41. Now we will create the JIRA Bug collection, for this we will map **Bug** artifact of JIRA with **Defect** model of Tasktop

42. We will map the fields



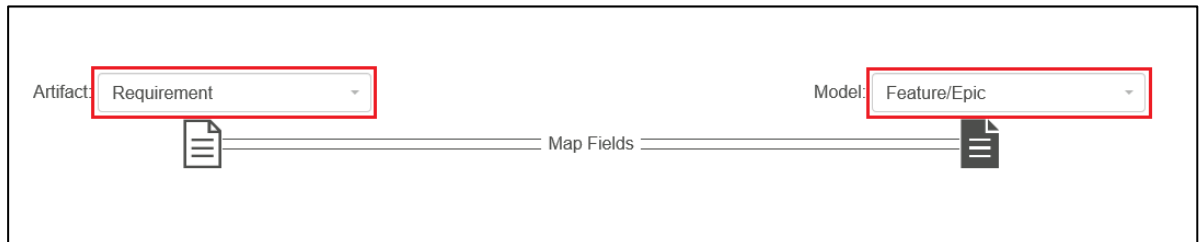
43. Here we can see that two fields are not mapped properly, as we are not going to use Resolution field, we will simply delete this field. After this we will map the Priority field like we map for EPIC and User Story collections and at the end we will save this collection. Now our JIRA Bug collection is ready to use.

44. Now our JIRA Collections are ready and we will create the Tosca Collections. First we will create the Tosca EPIC Collection, for this we will choose the Tosca Connection.

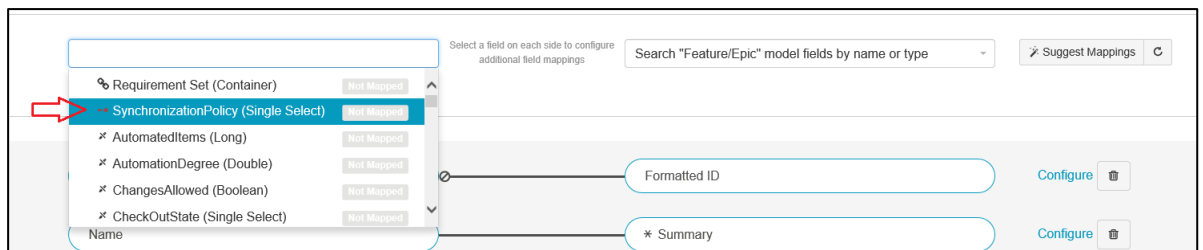
45. Now we will again click on **Manage Projects** to select project

46. From the list choose the project you want to use and then click on **Done** button

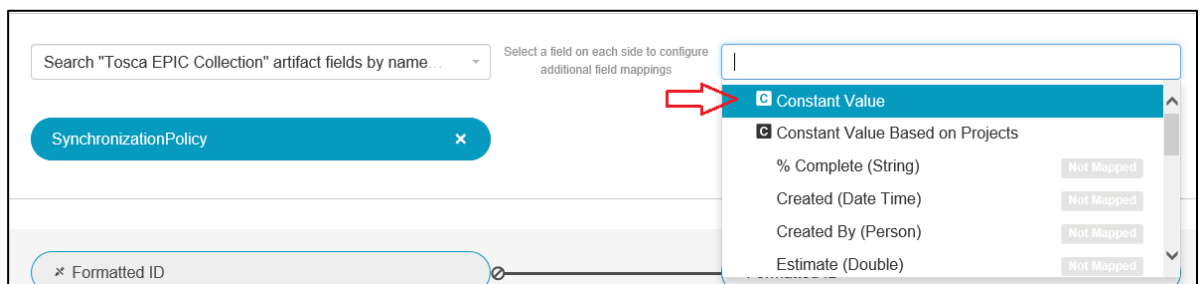
47. For this collection, we will choose **Requirement** as artifact from Tosca and **Feature/Epic** model from Tasktop. For JIRA EPIC Collection, we have already chosen the model Feature/Epic and for Tosca also we have choose the Feature/Epic Model, so this model is standing in-between this artifact to flow the data from one end to another end.



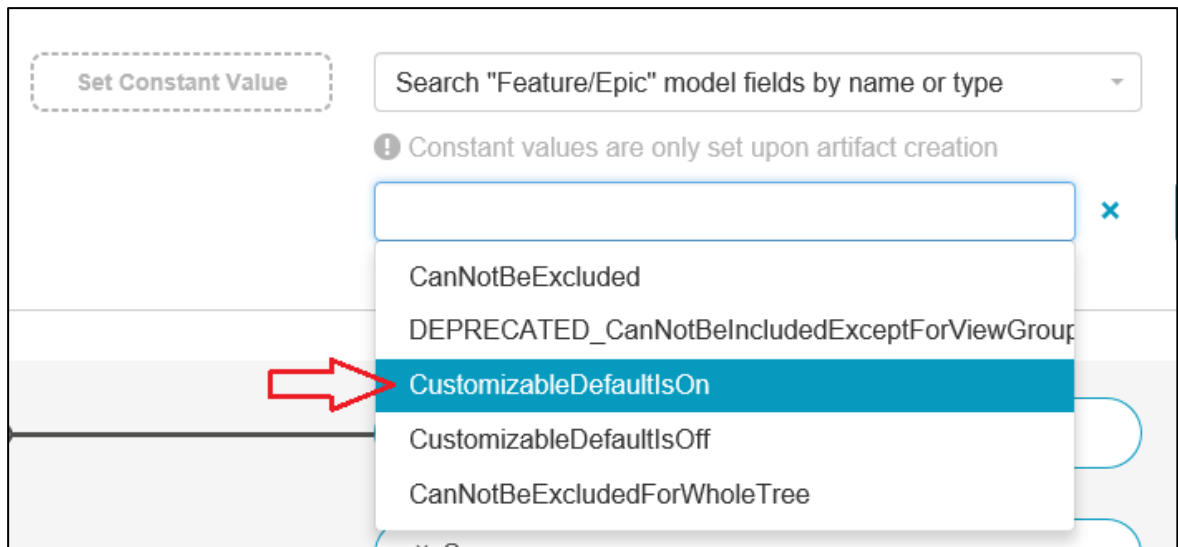
48. Next thing is to **Map Fields**, here we are required to map one extra field. For this we will choose the field **SynchronizationPolicy**



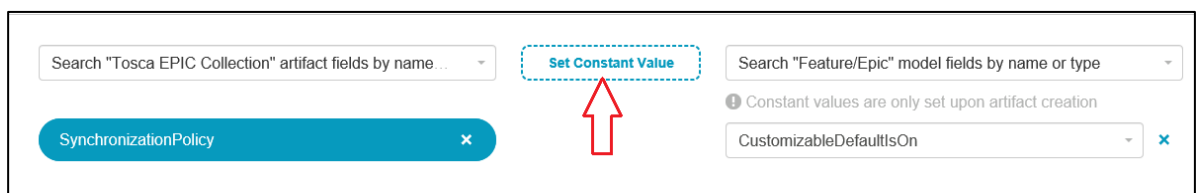
49. From **Feature/Epic** model we will choose **Constant Value** and wait for some time to load the constant values



50. From the constant values choose **CustomizableDefaultsOn**



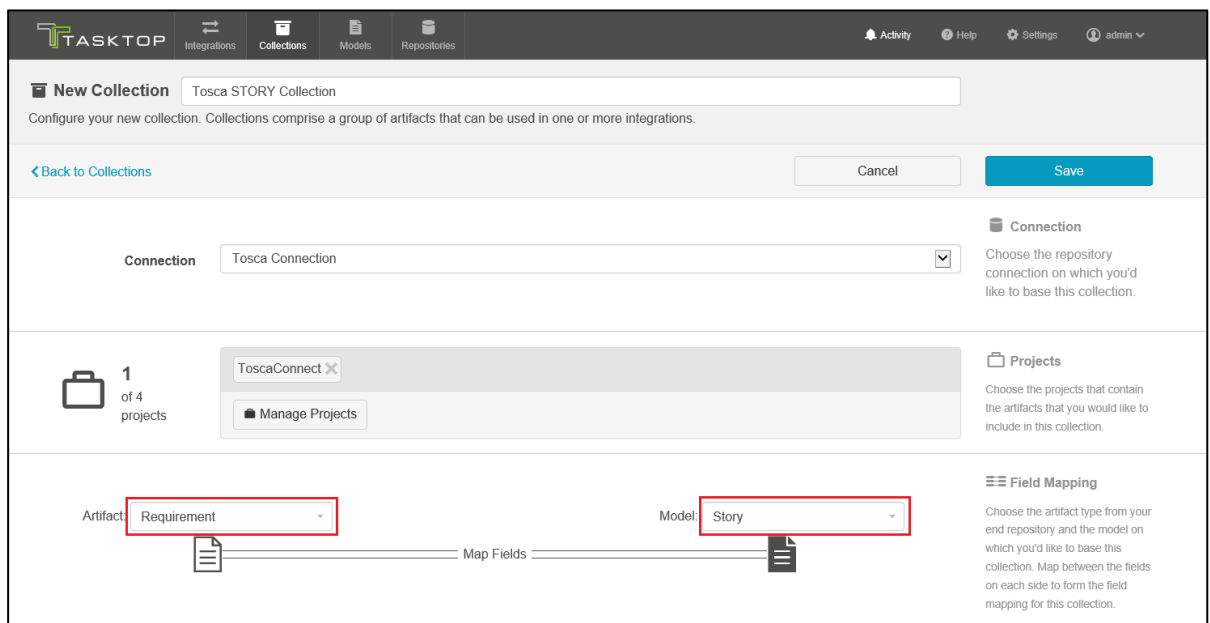
51. Now we will click on **Set Constant Value** button, and save the collection. Now Tosca EPIC Collection is ready to use.



Here we are adding one extra mapping **SynchronizationPolicy** to know more about this read this KB article

<https://support.tricentis.com/community/article.do?number=KB0011292>

52. Now we will create Tosca STORY Collection, For this we will choose the Requirement artifact from Tosca and Story model from Tasktop.



53. Now again we will map the extra field i.e. SynchronizationPolicy. Now save the collection and our Tosca Story Collection is ready to use.

Field Mapping: Tosca STORY Collection
View and manage the field mapping for this collection. The field mapping specifies how fields from your repository artifact map to fields in your model.

[Back to Collection](#) Cancel Save

Tosca Connection: Requirement 5 of 52 fields mapped

Model: Story 4 of 18 fields mapped

Search "Tosca STORY Collection" artifact fields by name or type Select a field on each side to configure additional field mappings Search "Story" model fields by name or type Suggest Mappings C

Tosca Connection: Requirement	Model: Story	Configure	Trash
SynchronizationPolicy	CustomizableDefaultIsOn	Configure	
Formatted ID	Formatted ID	Configure	
Name	Summary	Configure	
Description	Description	Configure	
URL	URL	Configure	

54. Now we will create our last collection i.e. Tosca Bug Collection, for this we will choose Defect artifact from Tosca and Defect model from Tasktop.

Tasktop Integrations Collections Models Repositories Activity Help Settings admin

New Collection Tosca BUG Collection
Configure your new collection. Collections comprise a group of artifacts that can be used in one or more integrations.

[Back to Collections](#) Cancel Save

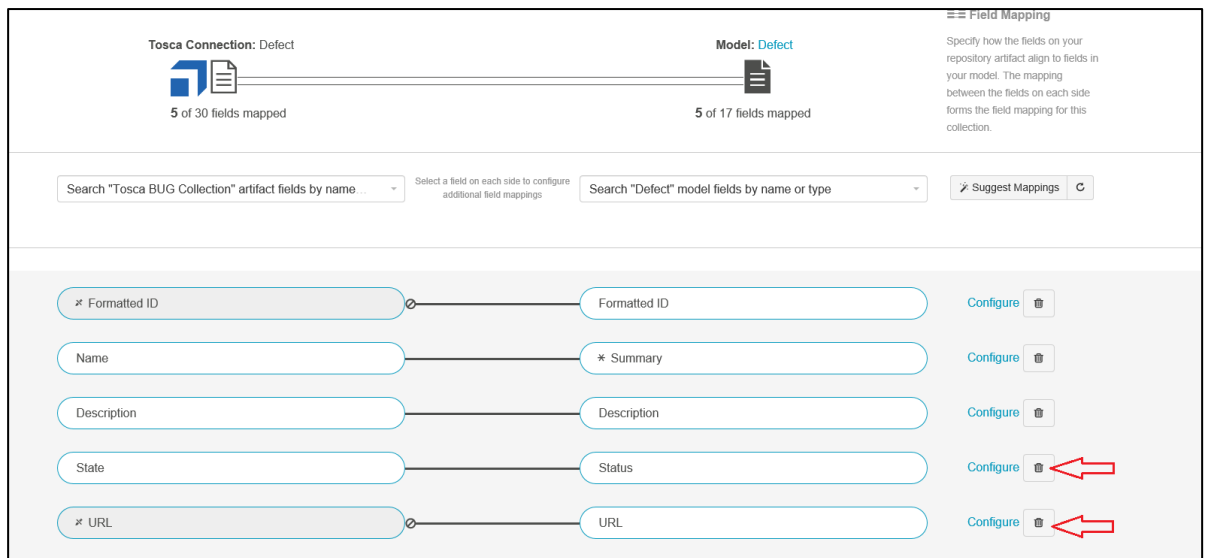
Connection Tosca Connection ▼
Choose the repository connection on which you'd like to base this collection.

Projects 1 of 4 projects
ToscaConnect ×
Manage Projects
Choose the projects that contain the artifacts that you would like to include in this collection.

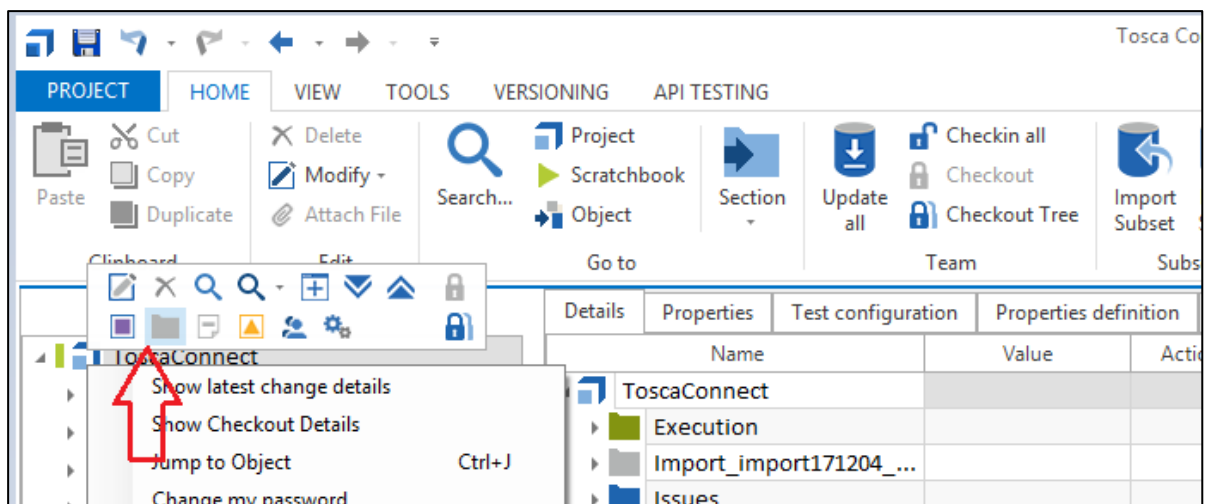
Artifact Defect ▼ **Model** Defect ▼
Map Fields

Field Mapping
Choose the artifact type from your end repository and the model on which you'd like to base this collection. Map between the fields on each side to form the field mapping for this collection.

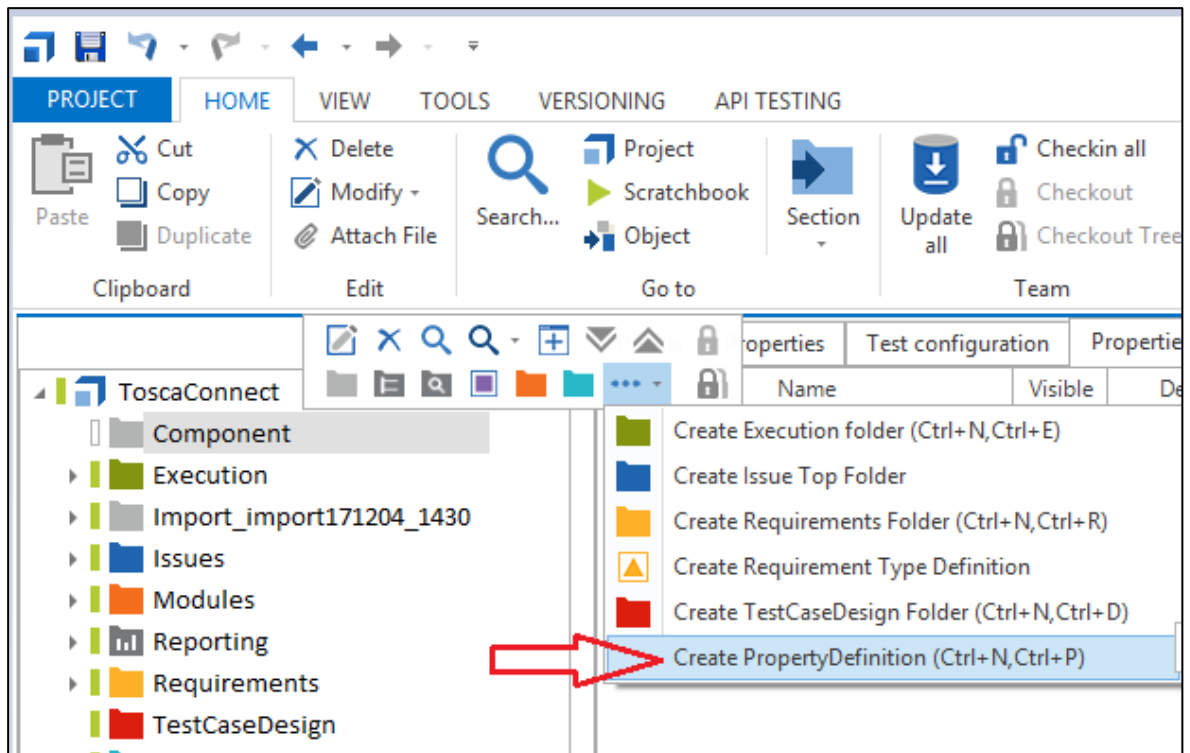
55. Here two extra field are mapped and we don't require that so we will delete state and URL fields. Now we need to map the Priority field, but if we search in the artifacts field of Tosca then we will not be able to see this field because it is not available inside Tosca. So, we required to create this filed inside Tosca.



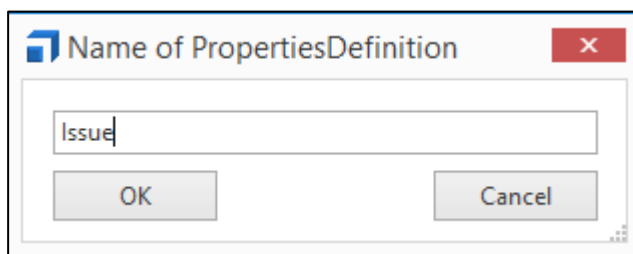
56. To create this filed inside Tosca open the workspace you have choose for this integration and checkout the workspace. Now create one Component folder inside workspace.



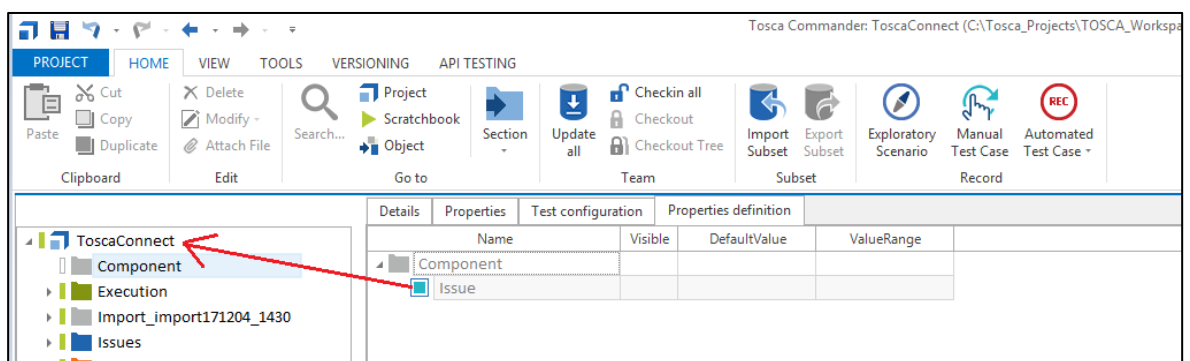
57. Now create one Property Definition for this folder



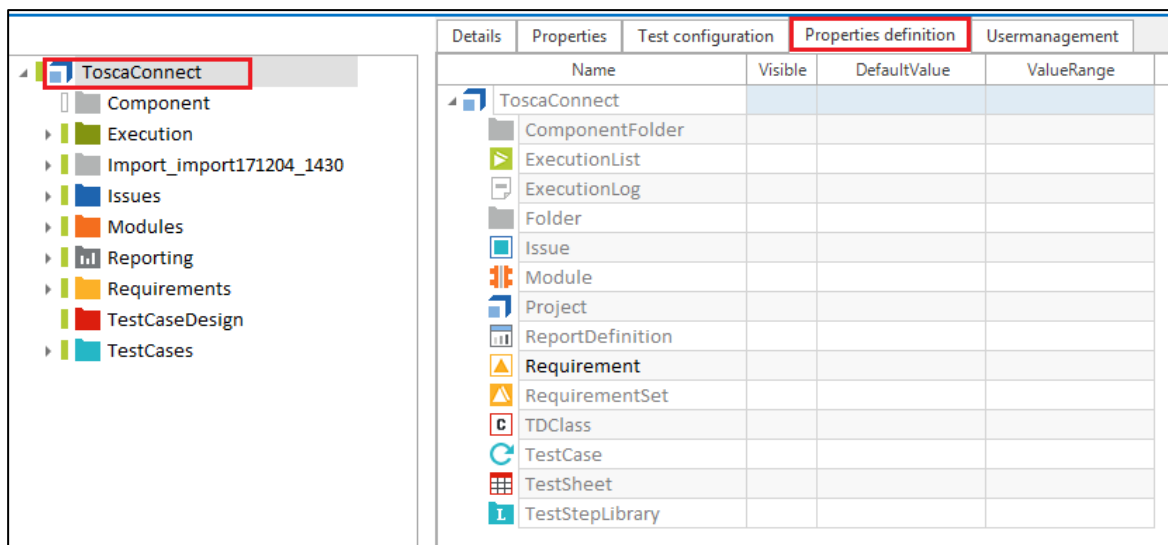
58. Name this PropertiesDefinition to **Issue** and click on OK



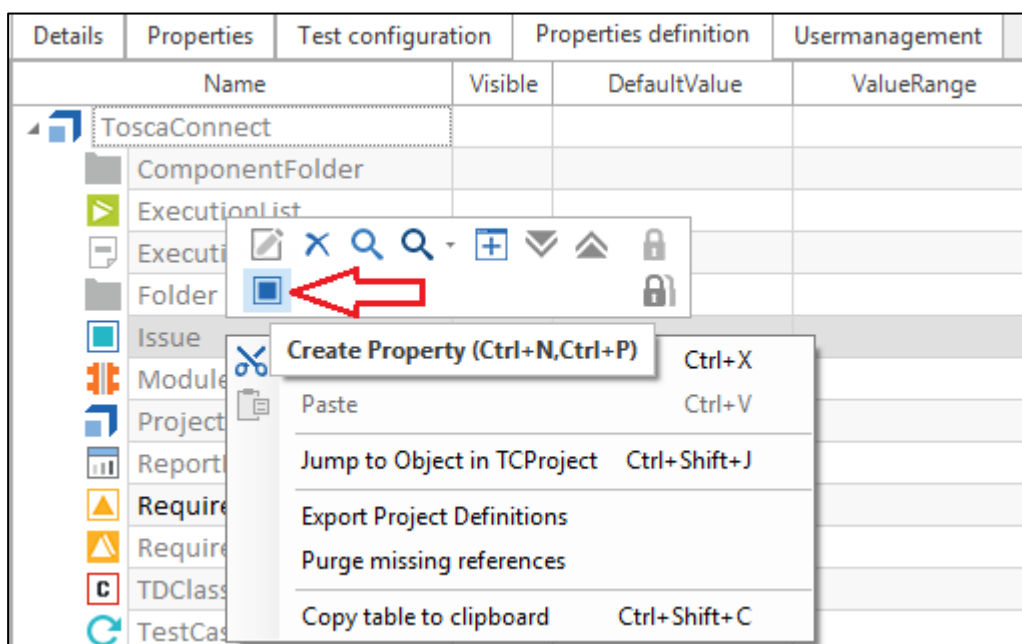
59. Now drag and drop this property definition to Project level of the workspace



60. Now you will be able to see this property definition at root level of project



61. Now right click on property definition and choose the option create property



62. Now rename this property to Priority and Put the **ValueRange** as "Highest;High;Medium;Low;Lowest". Give "Medium" as default value and save the workspace. Now check-in all so that changes will get reflected to all the users.

Details	Properties	Test configuration	Properties definition	Usermanagement	
Name	Visible	DefaultValue	ValueRange		
ToscaConnect					
ComponentFolder					
ExecutionList					
ExecutionLog					
Folder					
Issue					
Priority	<input checked="" type="checkbox"/>	Medium	Highest;High;Lowest;Low;Lowest		
Module					

63. Go back to Tasktop again and click on Refresh button which is just next to Suggest Mappings

[Back to Collection](#)
Cancel
Save

Tosca Connection: Defect

3 of 29 fields mapped

Model: Defect

3 of 17 fields mapped

Search "Tosca BUG Collection" artifact fields by name ...

Select a field on each side to configure additional field mappings

Search "Defect" model fields by name or type

Suggest Mappings

Formatted ID

Formatted ID

Configure

Name

Summary

Configure

Description

Description

Configure

64. You will be able to see Priority field, Now connect this mapping with Priority field of Defect model

Search "Tosca BUG Collection" artifact fields by name...

Connect

Search "Defect" model fields by name or type

Suggest Mappings

Priority

Priority

Clear Workspace

65. Now also map the SynchronizationPolicy field and save the collection

Field Mapping: Tosca BUG Collection
View and manage the field mapping for this collection. The field mapping specifies how fields from your repository artifact map to fields in your model.

[Back to Collection](#) [Cancel](#) [Save](#)

Tosca Connection: Defect **Model: Defect**

5 of 30 fields mapped 4 of 17 fields mapped

Search "Tosca BUG Collection" artifact fields by name ... Select a field on each side to configure additional field mappings Search "Defect" model fields by name or type [Suggest Mappings](#)

SynchronizationPolicy	CustomizableDefaultsOn	Configure
Priority	Priority	Configure
Formatted ID	Formatted ID	Configure
Name	Summary	Configure
Description	Description	Configure

66. Our all 6 collections are ready to use, now next thing is to do the integration between this collection so that they can synchronized the data. Click on Integrations menu to start the integrations

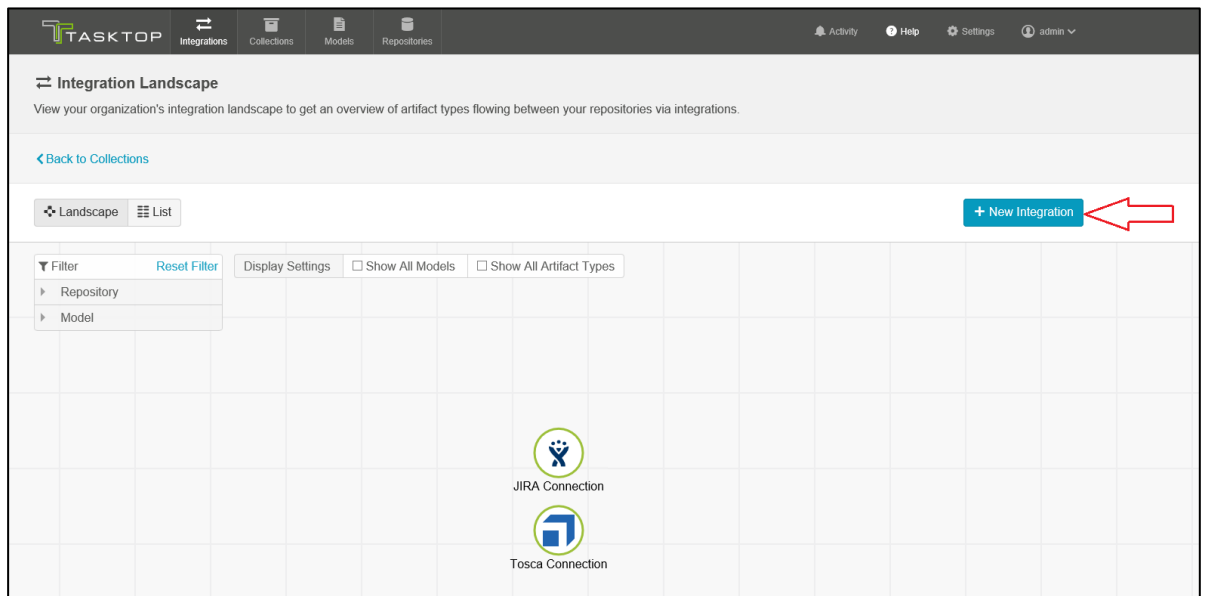
TASKTOP [Integrations](#) [Collections](#) [Models](#) [Repositories](#) [Activity](#) [Help](#) [Settings](#) [admin](#)

Collections
View your existing collections and create new ones. Collections comprise a group of artifacts that can be used in one or more integrations.

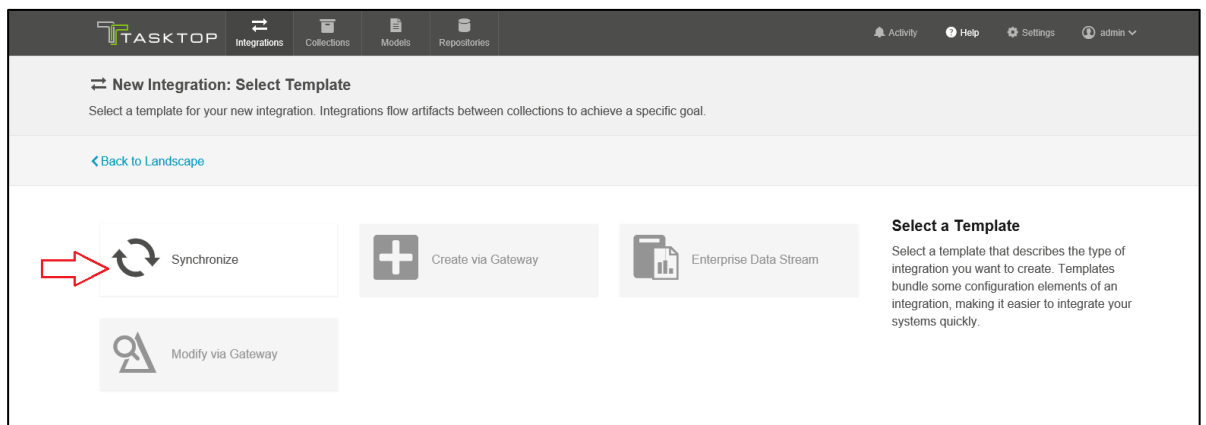
[Search](#) [+ New Collection](#)

Collection	Model	Action
JIRA Connection		
JIRA BUG Collection 1 Project Artifact Type Bug	Defect	Delete
JIRA EPIC Collection 1 Project Artifact Type Epic	Feature/Epic	Delete
JIRA STORY Collection 1 Project Artifact Type Story	Story	Delete
Tosca Connection		
Tosca BUG Collection 1 Project Artifact Type Defect	Defect	Delete
Tosca EPIC Collection 1 Project Artifact Type Requirement	Feature/Epic	Delete
Tosca STORY Collection 1 Project	Story	Delete

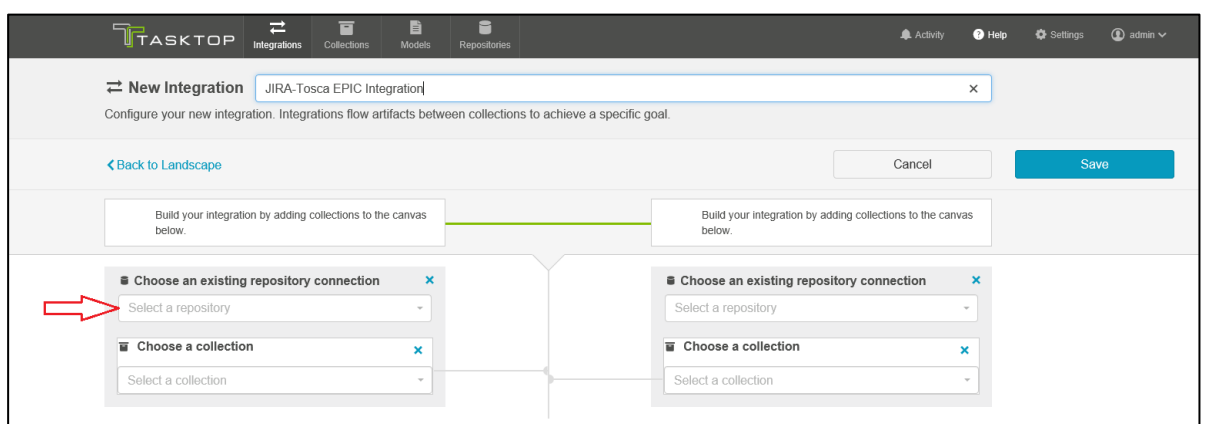
67. Here we are required to create the 3 integrations, one is for Epic collection, second is for Story collection and last is for Bug Collection. Click on New Integration



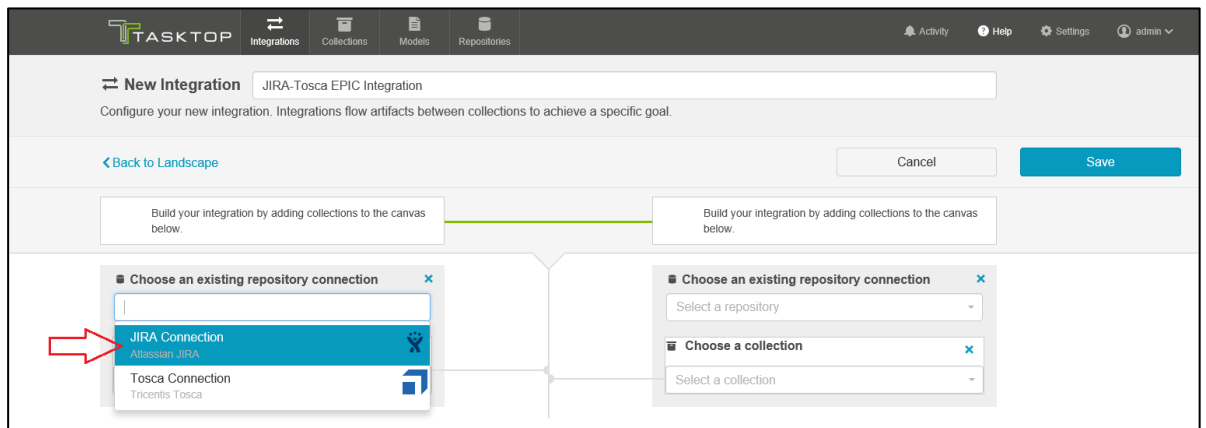
68. From the options choose Synchronize



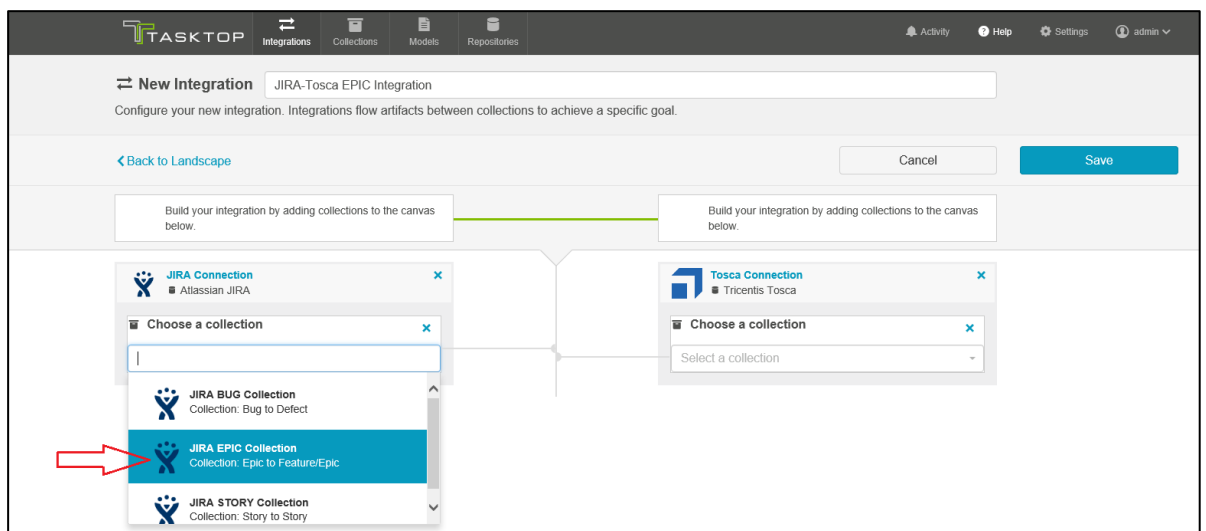
69. First we will create integration for Epic Collection. Provide the appropriate name. Now click on "Choose an existing repository connection".



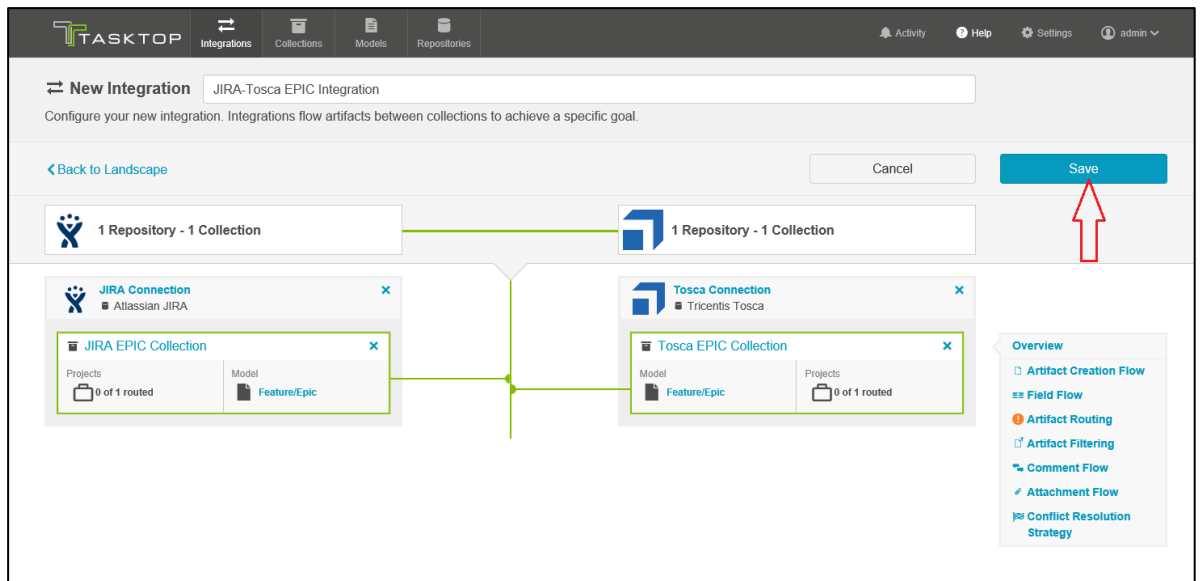
70. Now select JIRA Connection from the list and same way choose the Tosca Connection from right hand side.



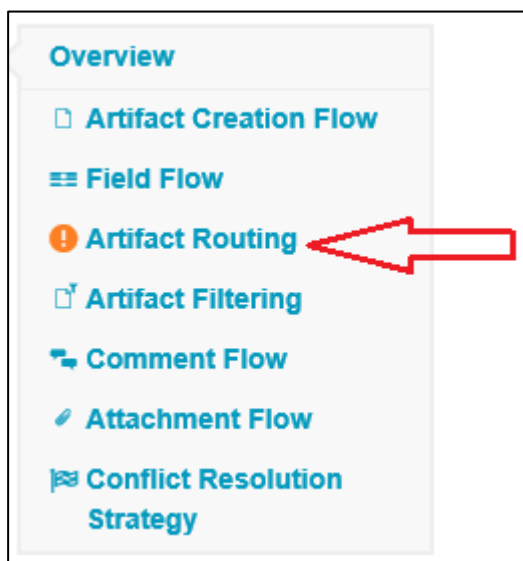
71. Now choose JIRA Epic Collection from left hand side and Tosca Epic collection from right hand side



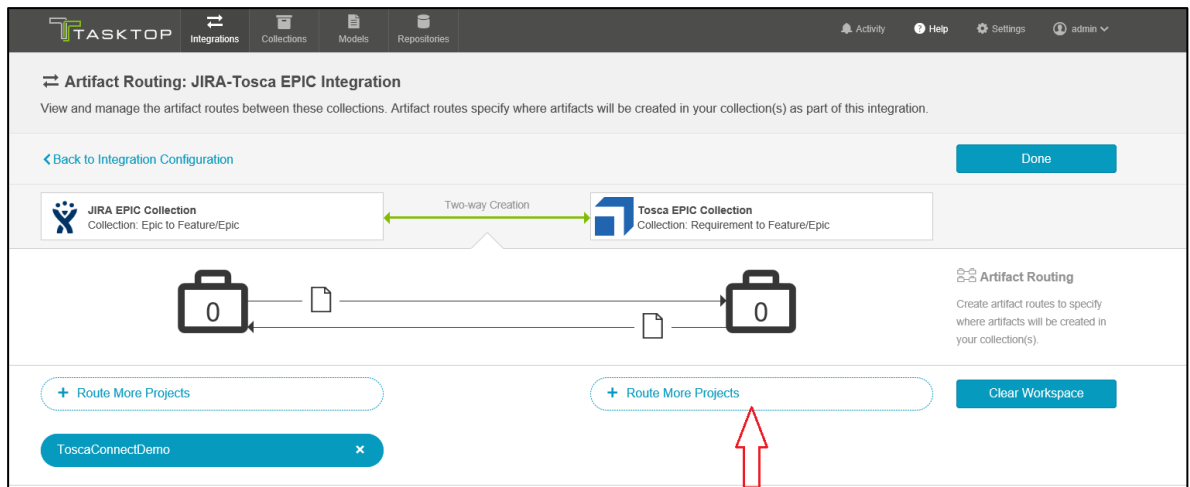
72. Click on Save button



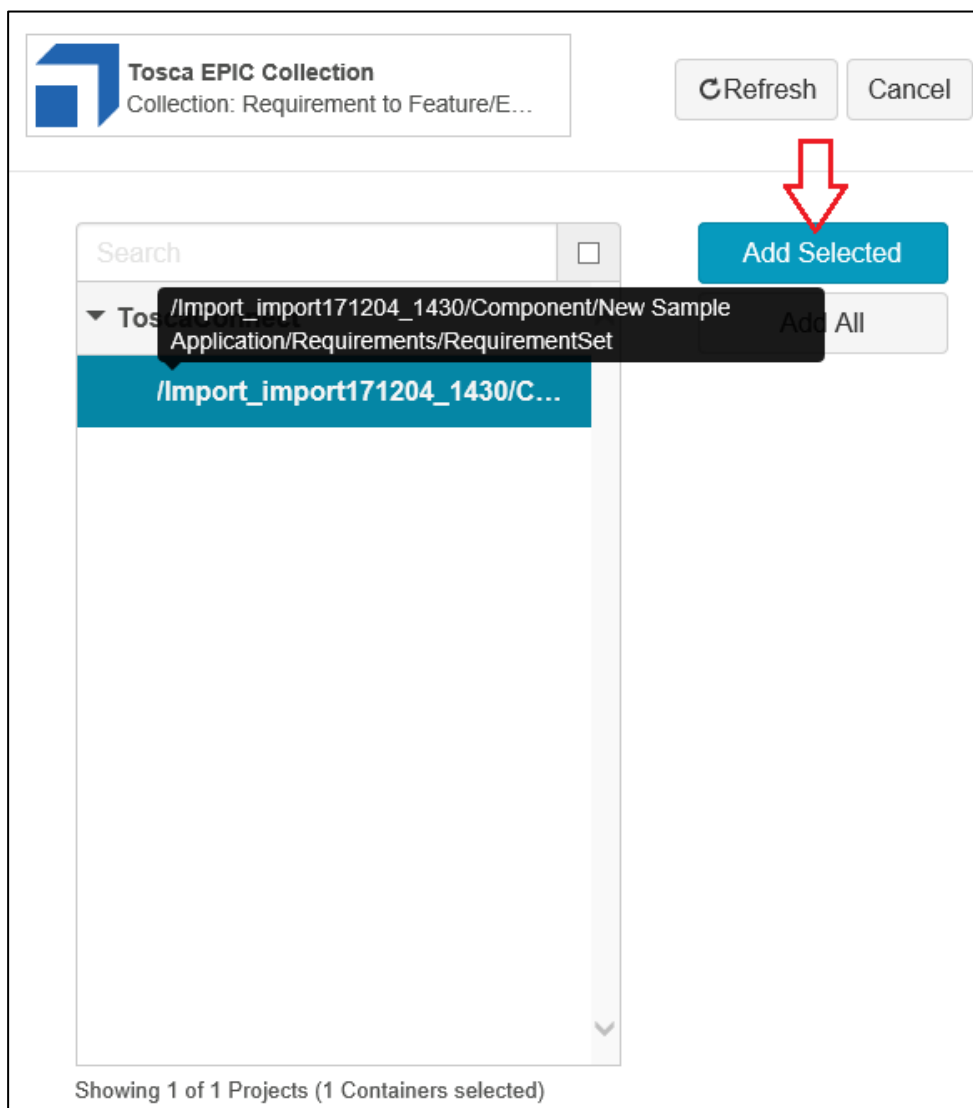
73. Now you will see an error at Artifact Routing, click on it to configure it correctly



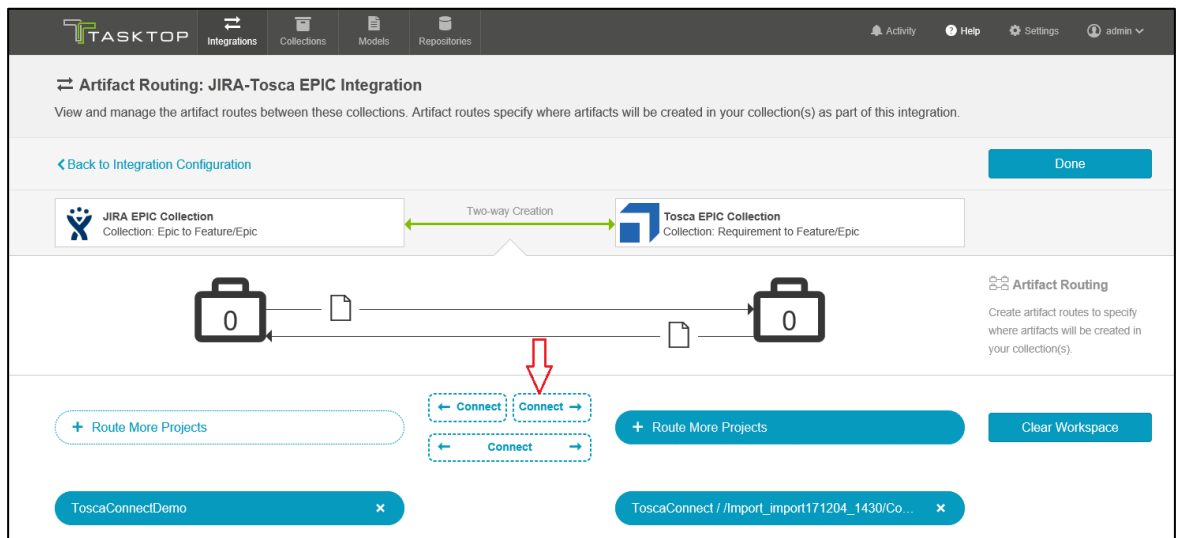
74. Click on **Route More Projects** from Tosca Epic Collection side



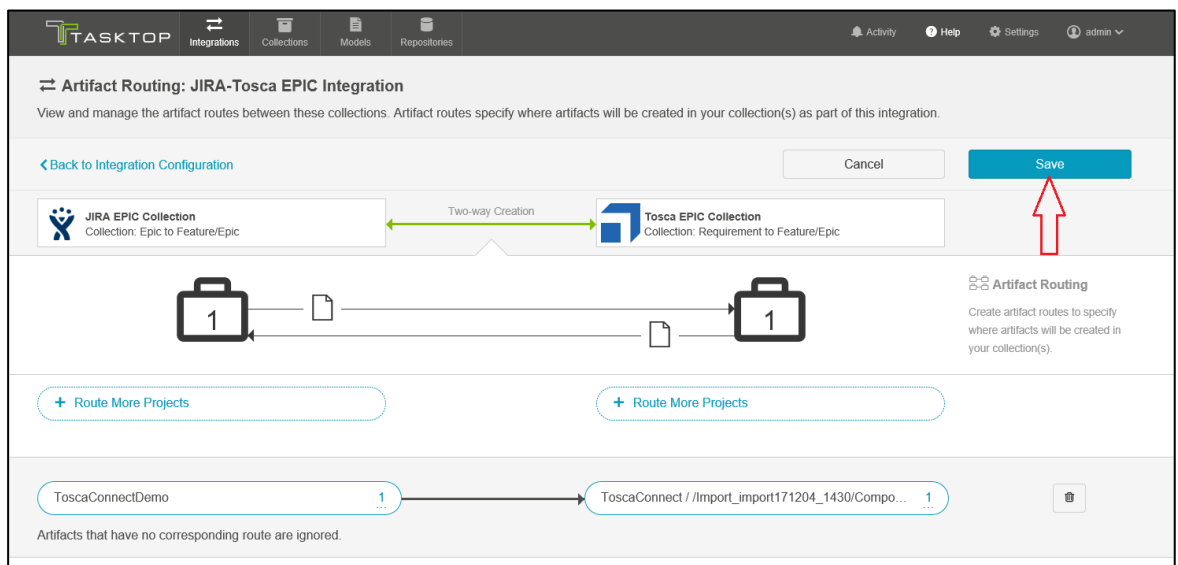
75. Choose the RequirementSet from the list and click on **Add Selected**



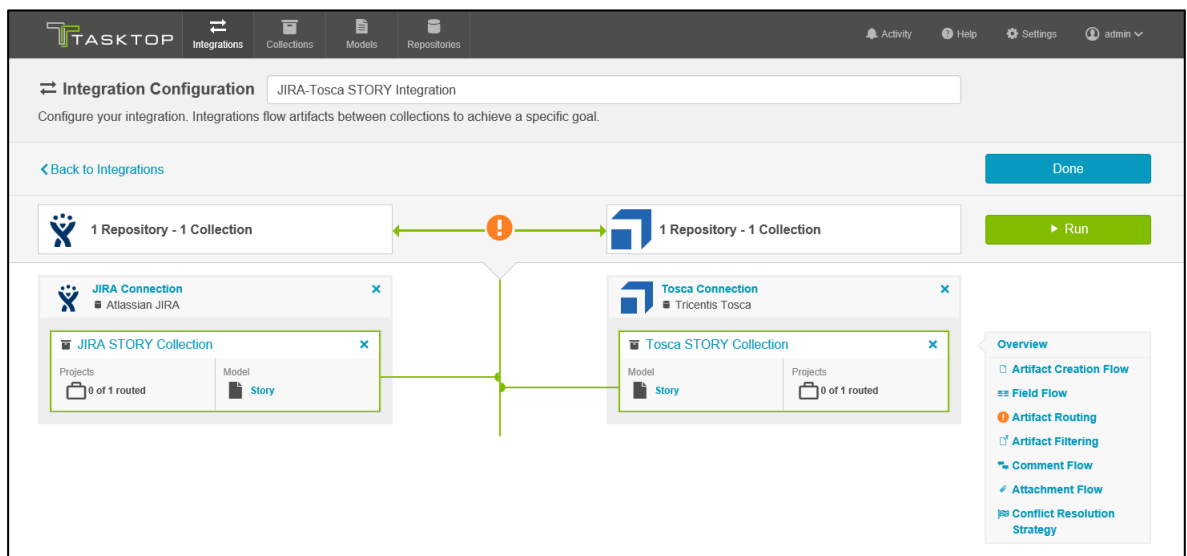
76. Now you can see three connection options are available, one for both way connect and two for single way connect. Here we are connecting JIRA Epic to Tosca Requirement so we will select one way connect which will connect from JIRA Epic to Tosca Requirement



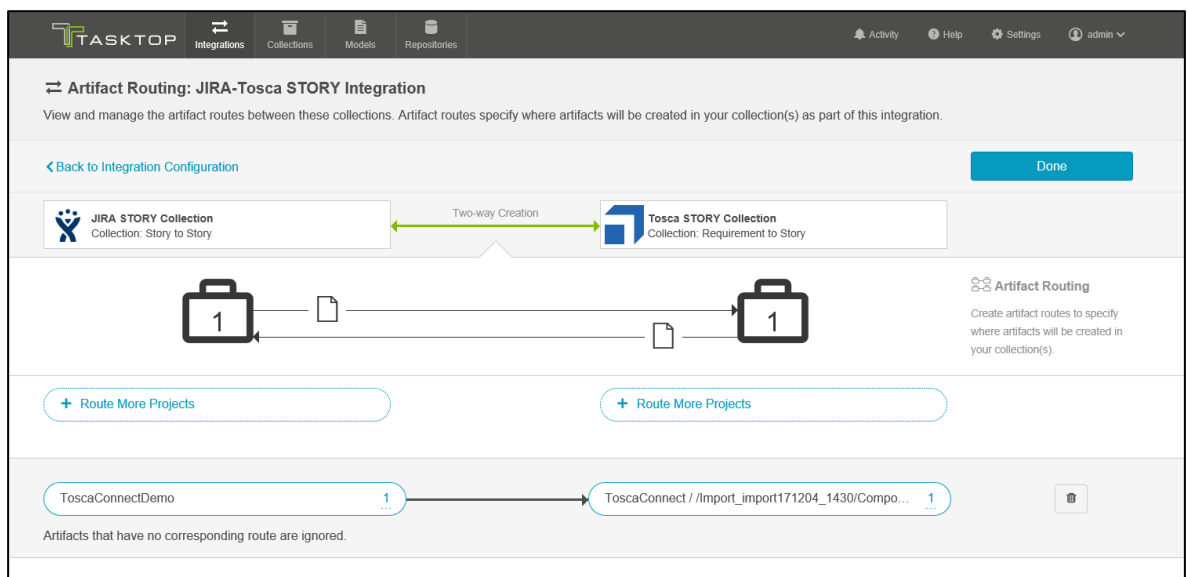
77. Once you click on connect button you will see the connection, now click on **Save** button and then click on **Done** button. Now our JIRA-Tosca Epic is ready to use



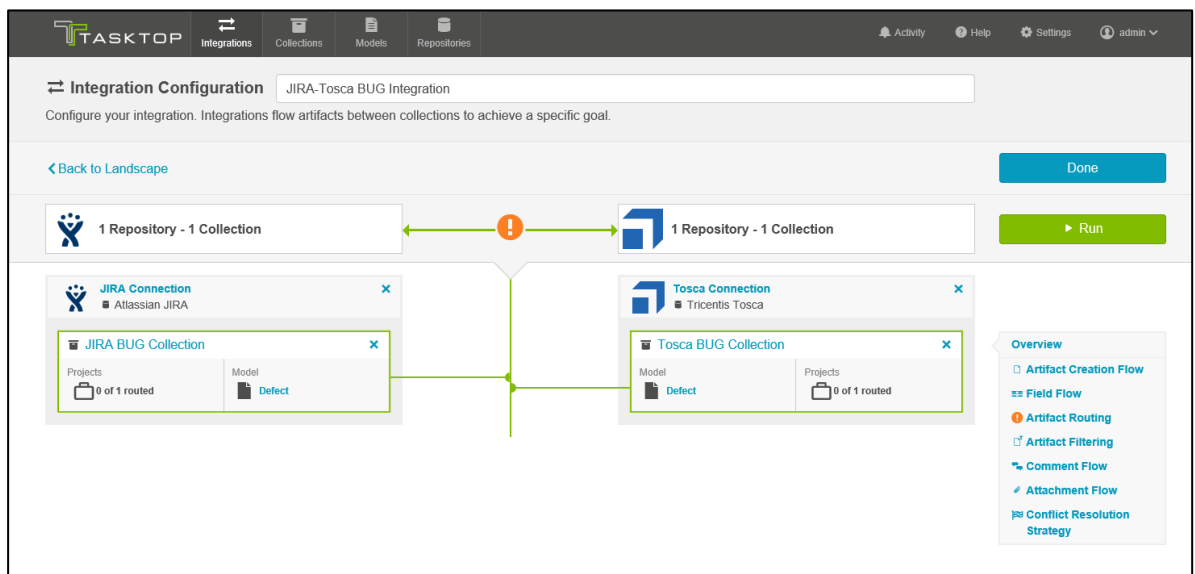
78. Same way we will create the integration for JIRA-Tosca Story collection, here we will choose JIRA Story collection and Tosca Story Collection from the respective side



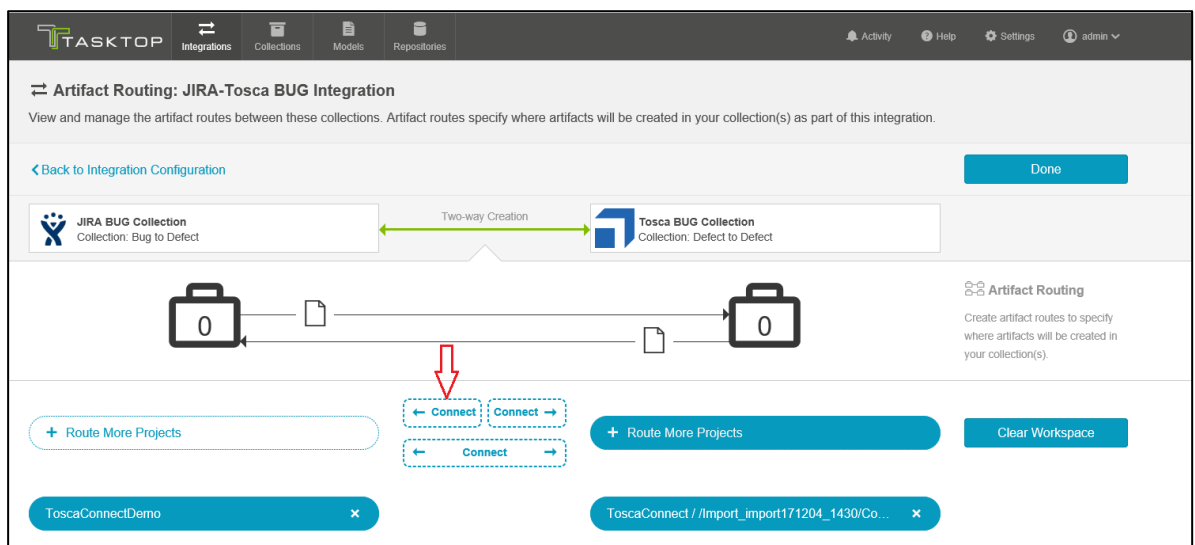
79. Now we will do the artifact routing and we will choose single direction routing. Now save this integration and our second integration is ready to use



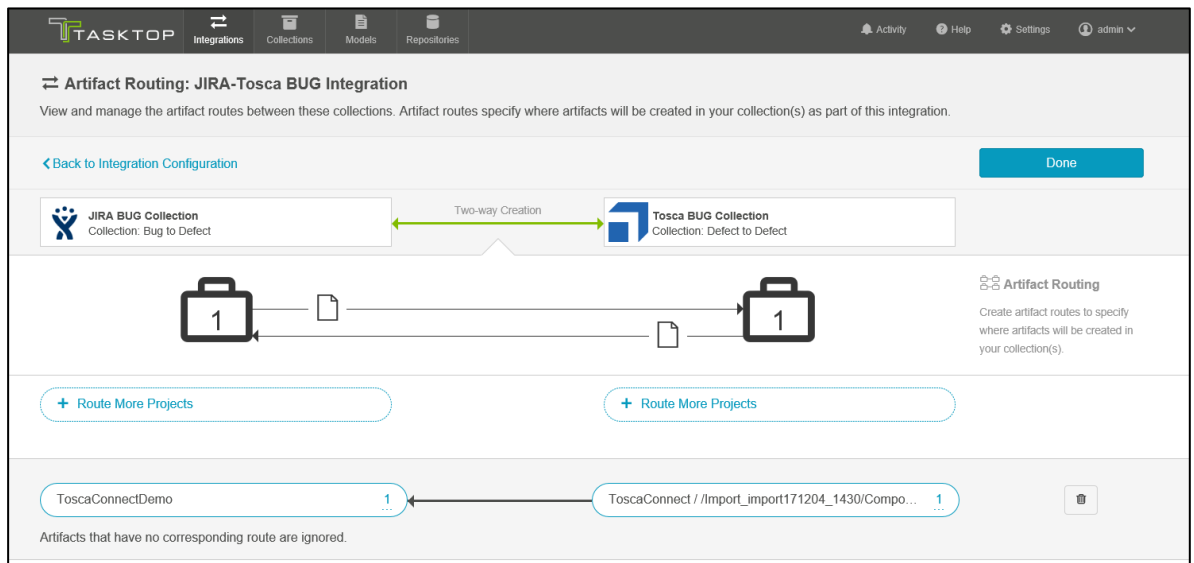
80. Now we will create the last integration for JIRA-Tosca Bug Integration, here choose JIRA Bug Collection and Tosca Bug Collection from the respective side



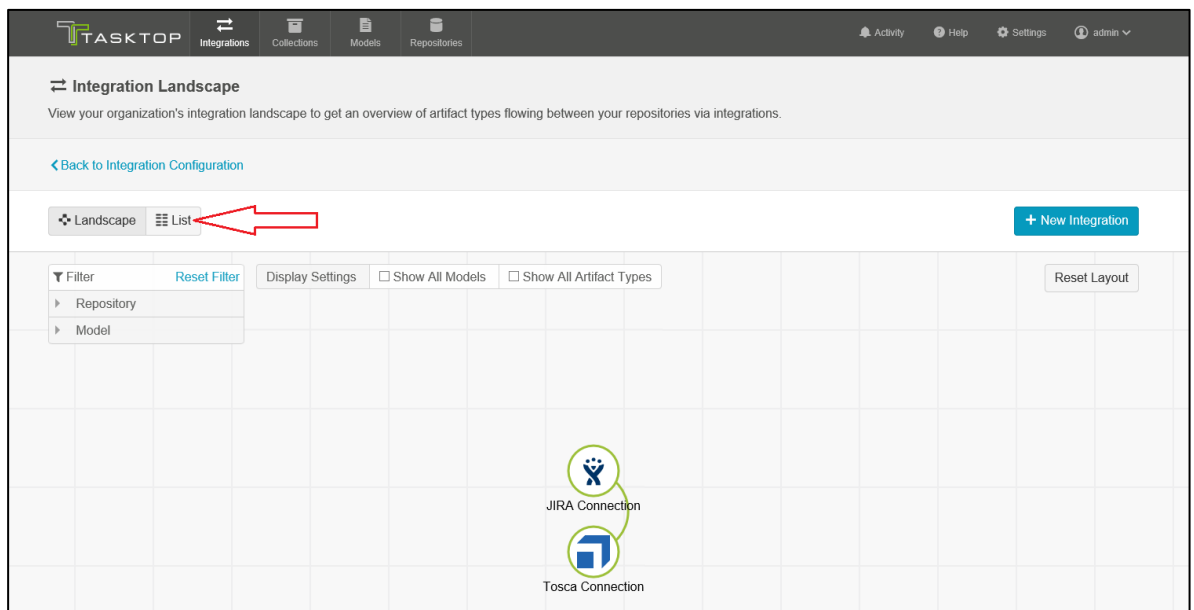
81. In defect integration, we will sync defect of Tosca with Bug of JIRA, so we here we will choose left single connect.



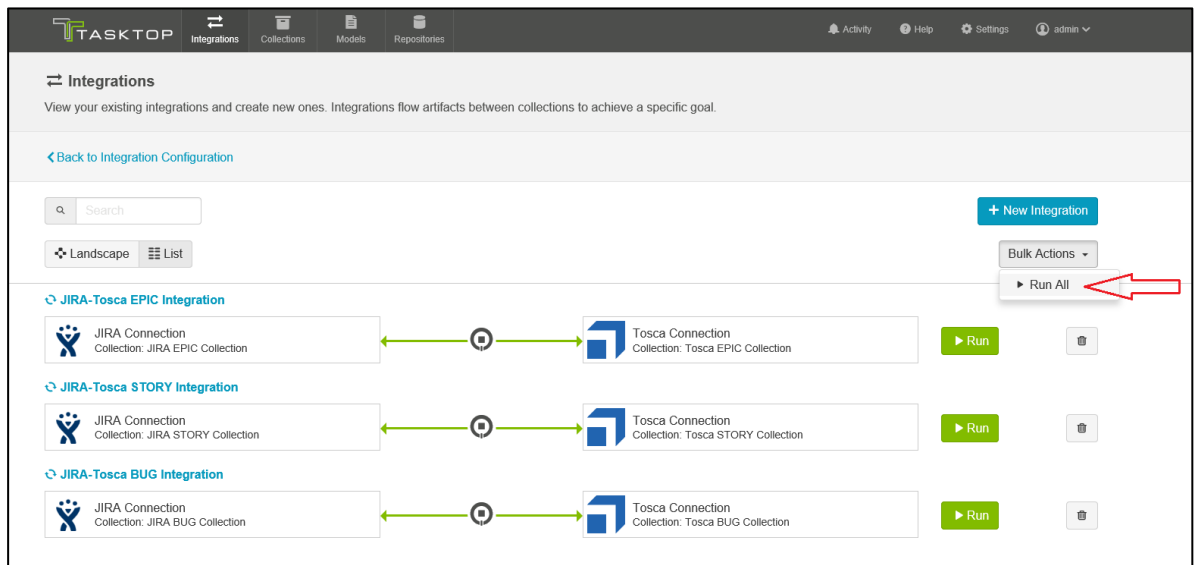
82. Once you select the connection click on Save and Done and your last integration is ready to use



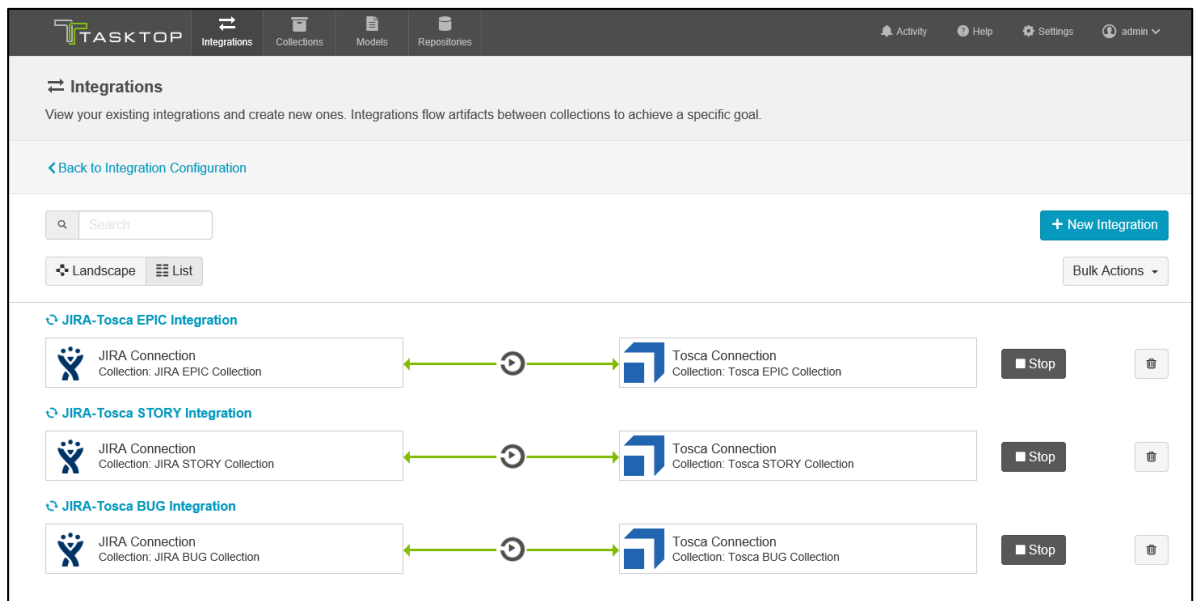
83. Go back to the integration and click on **List** button



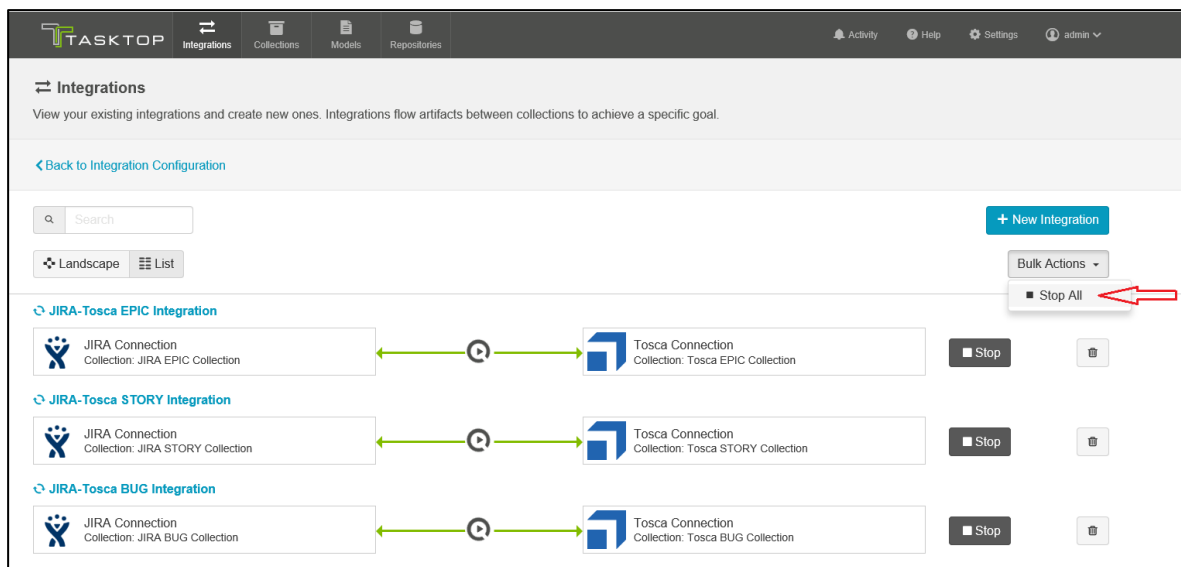
84. You will be able to see all integrations in single window, now click on Bulk Actions and then Run All.



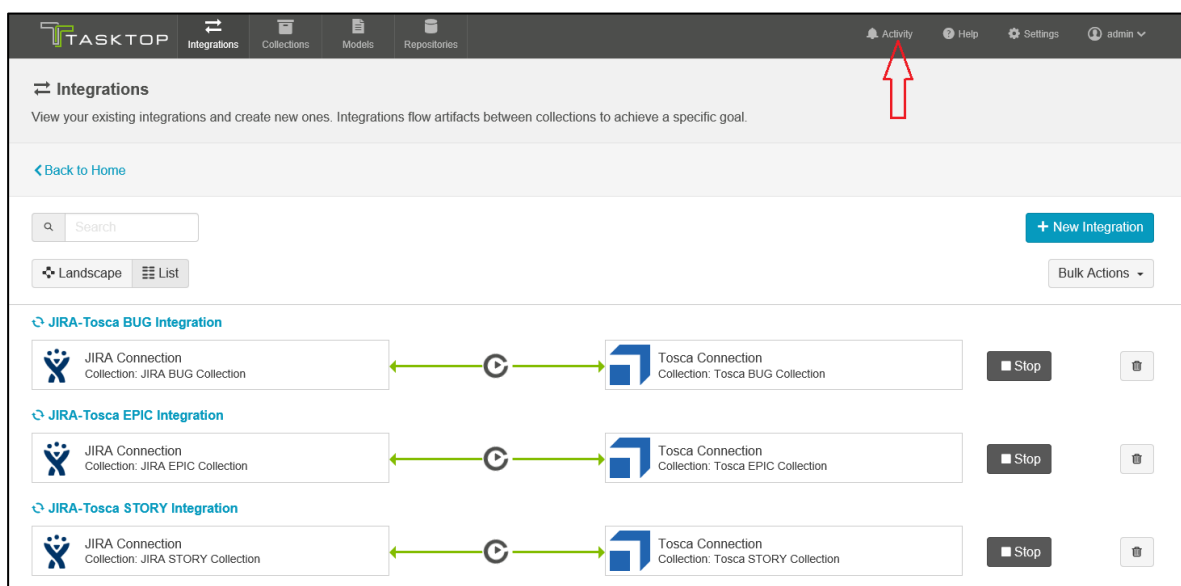
85. Now you all integrations are running, you can start or stop them individually



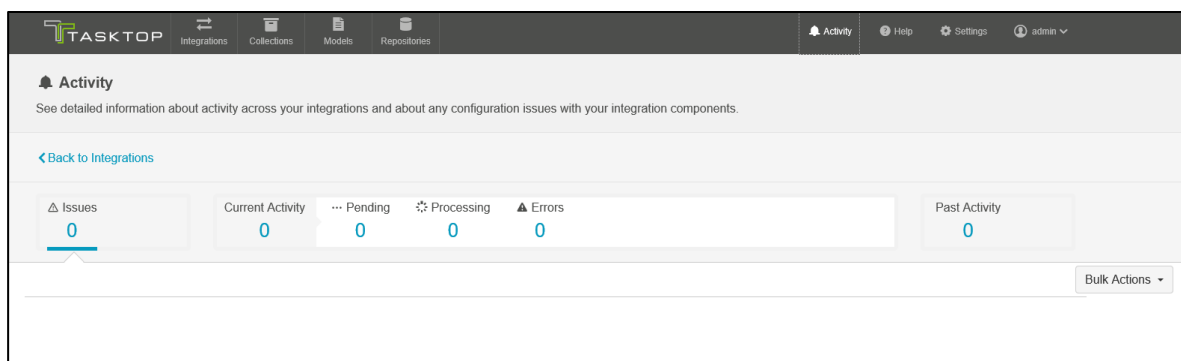
86. Now click on Bulk Actions and then Stop All, this will stop all the integrations



87. You can track the current activity by clicking on the Activity link

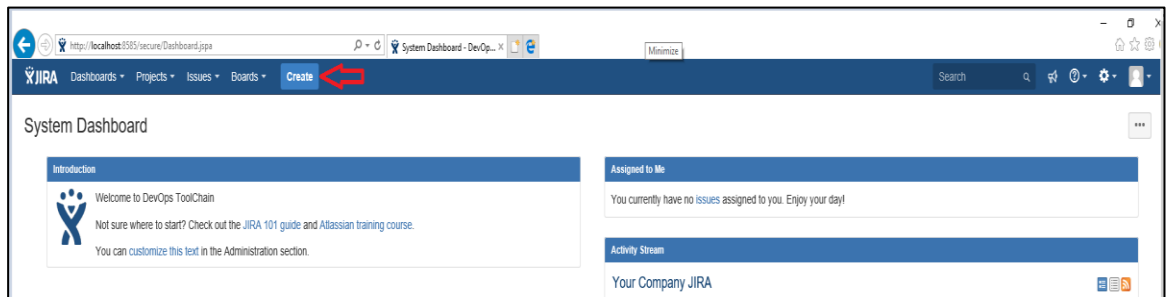


88. Here you can track all the activity details



4 Hands-on Example

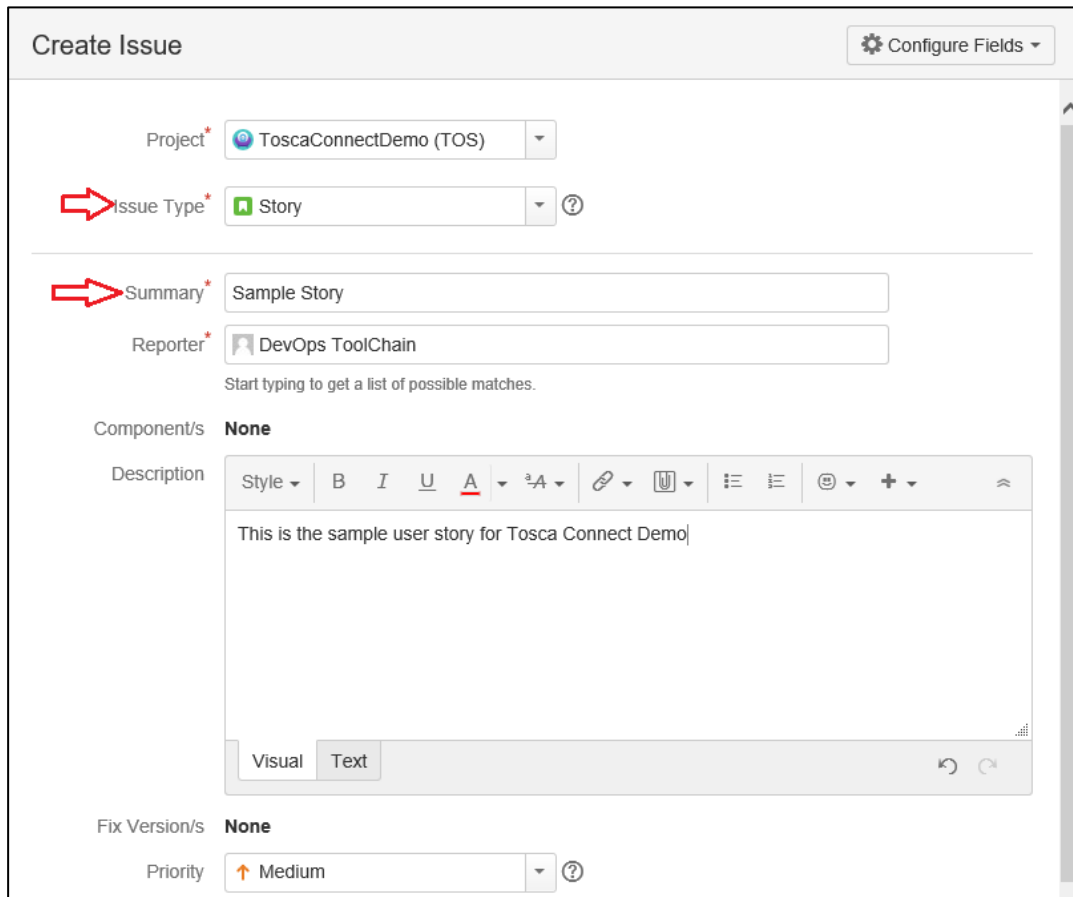
1. Log in to the JIRA and click on **Create**



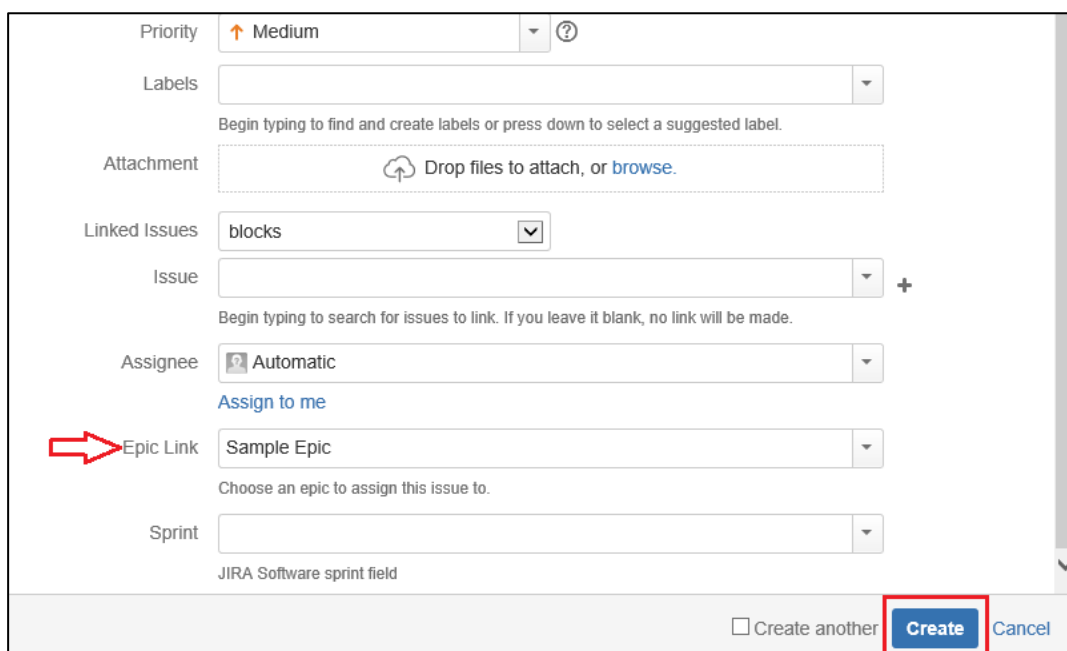
2. Now choose **Issue Type** as Epic, Put **Epic Name**, **Summary**, **Description** and then click on **Create** button, your Epic is created

The screenshot shows the 'Create Issue' form in JIRA. The form is titled 'Create Issue' and has a 'Configure Fields' button. The 'Project' is set to 'ToscaConnectDemo (TOS)'. The 'Issue Type' is set to 'Epic'. The 'Epic Name' field is filled with 'Sample Epic'. The 'Summary' field is also filled with 'Sample Epic'. The 'Reporter' is set to 'DevOps ToolChain'. The 'Component/s' is set to 'None'. The 'Description' field is filled with 'This is the sample epic for Tosca Connect Demo'. The 'Fix Version/s' is set to 'None'. The 'Priority' is set to 'Medium'. The 'Labels' field is empty. The 'Attachment' section shows a 'Drop files to attach, or browse.' button. At the bottom right, there is a 'Create' button highlighted with a red box, and a 'Cancel' button next to it. There is also a checkbox for 'Create another'.

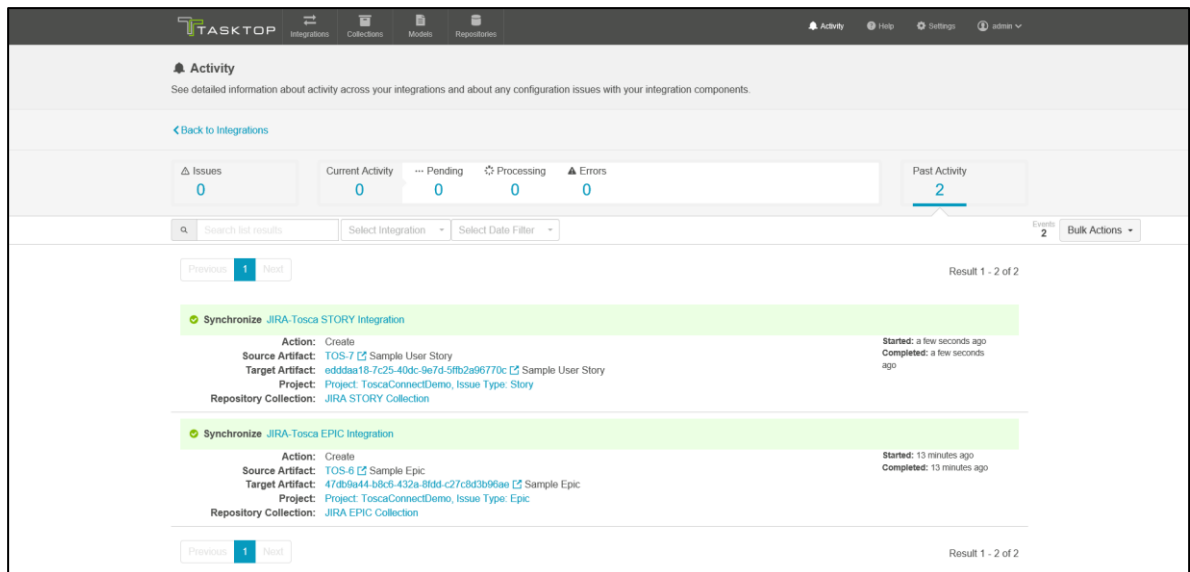
- Now once again click on **Create** and now choose **Issue Type** as Story, put **Summary** and scroll down



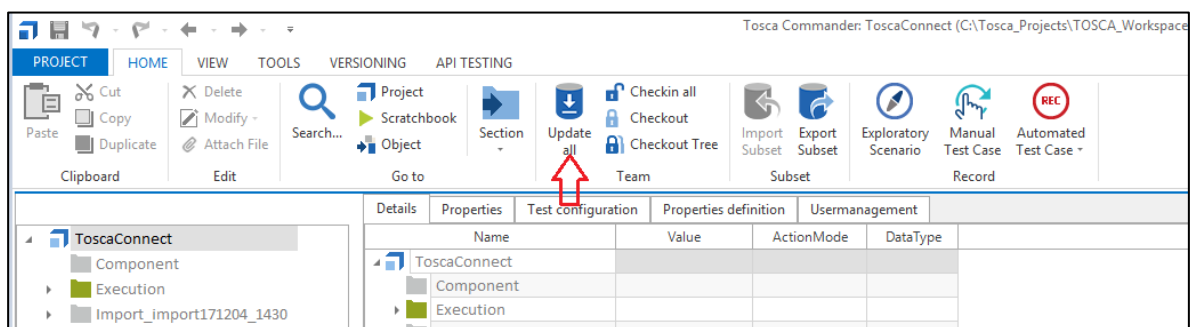
- Now click on Epic Link and select Epic you just have created and then click on **Create** button



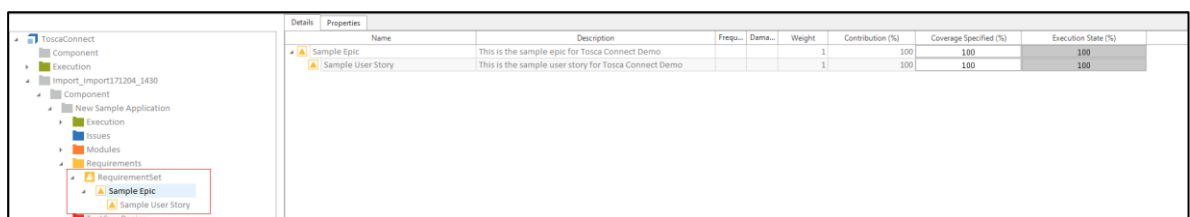
5. You can see the progress of the integration in **Activity** window of Tasktop



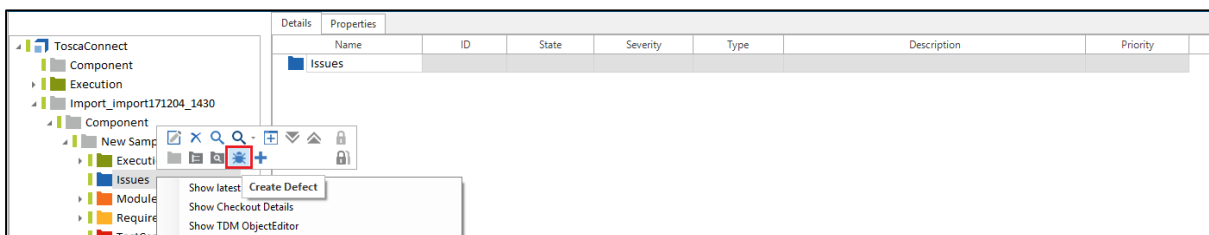
6. Open Tosca workspace and click on **Update all**



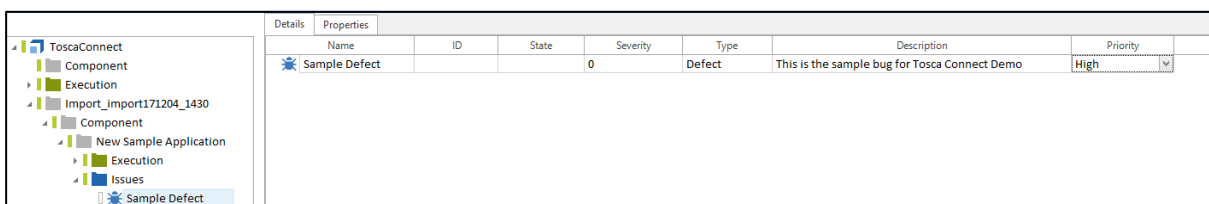
7. You can see the Epic and User Story are now linked to the Requirement set of Tosca and you can find the more information in Description



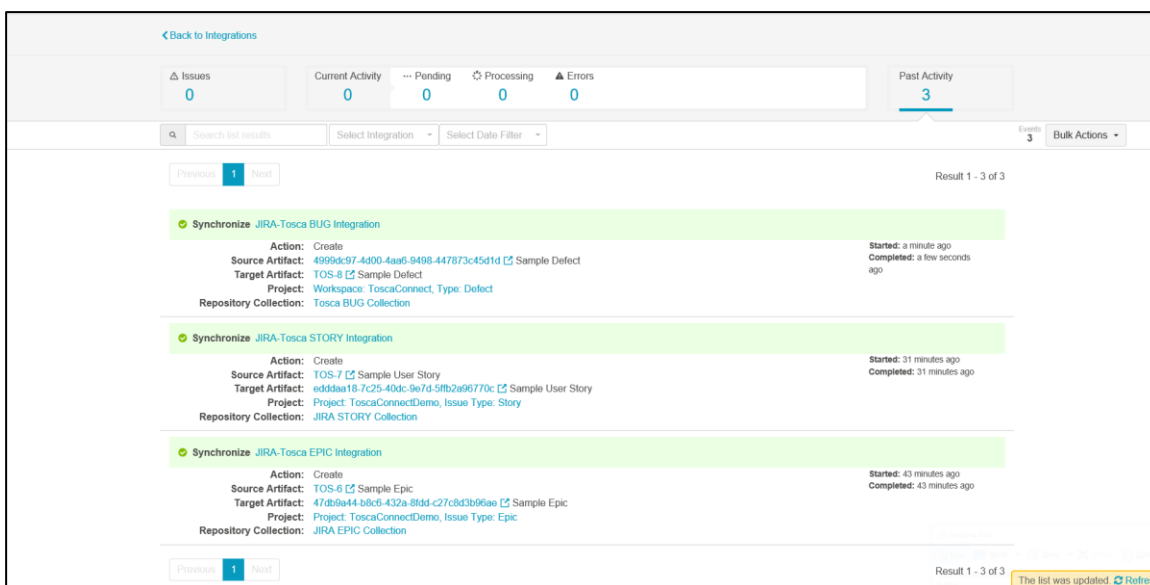
- Now checkout the workspace, create a defect by right clicking on Issue folder and then selecting the **Create Defect** option



- Fill the **Description** of it, also select the **Priority**. Save and checkin all the changes



- You can see the details of defect integration in **Activity** window of Tasktop



11. Once you log-in to the JIRA you can see that your new defect is created as a Bug inside JIRA

