Venkatesh Prasad

Venkatesh.prasad@ikea.com

Abstract

Changes are inevitable and with the rapid pace with which the software industry is transforming itself, companies have to make certain tough decisions with implementing changes to their tool sets that they are using. Any migration from one tool to another will come at a cost and several limitations. In most of the scenarios, we end up losing huge chunk of data as a result of migration, as the two tools have different architecture and cannot have a one to one mapping of data between them. Companies should make decisions keeping this in mind and should be well prepared with these limitations and impediments of migration.   
  
By reading several blogs, I realized that admins have tough time migrating data from Rational Team Concert to Jira. I therefore decided to do a POC on this project and I have provided a detailed step by step guide and documented my experiences here. Though this may not cover all the topics related to RTC to Jira migration, but this will surely get you started with migration and provide you helping hand with most of the scenarios.

**Detailed Instructions for Migrating Artifacts from Rational Team Concert to Jira**

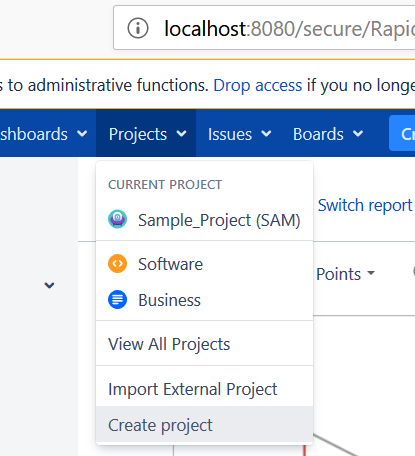
Applicable for Jira Server

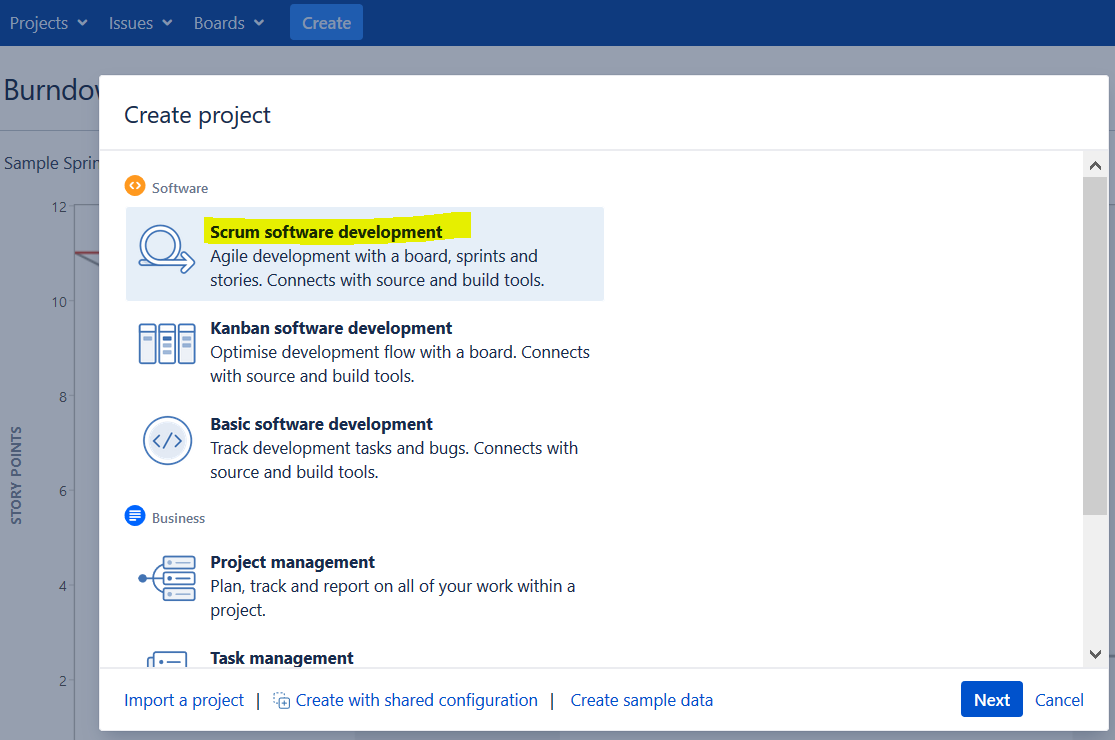
**Detailed Instructions for Migrating Artifacts from Rational Team Concert to Jira**

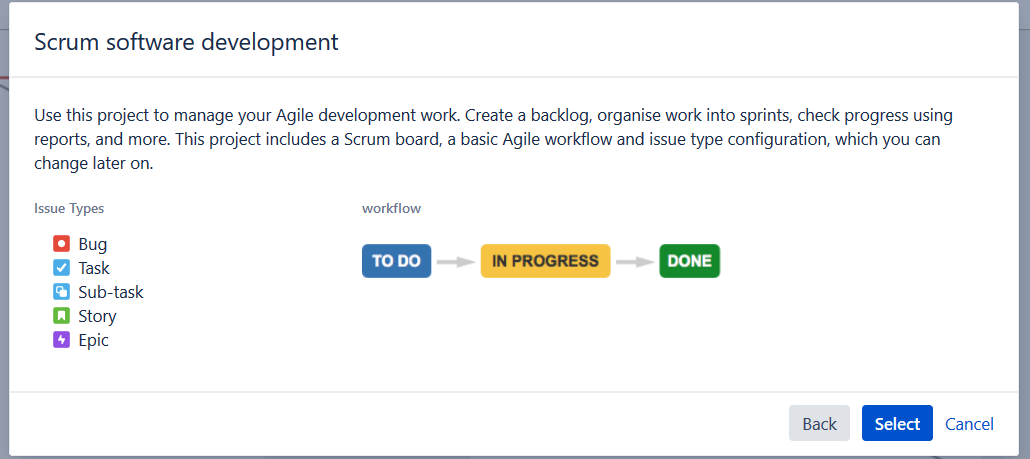
**This document works for Jira Server. Jira Cloud platform is almost same but with a few changes. Please refer the Atlassian Jira Documentation for different configuration changes between Jira Server and Jira Cloud.   
To proceed with the steps below, the assumption is that the Jira Server is installed and you have the Jira admin rights. Configuring the server is discussed below.**

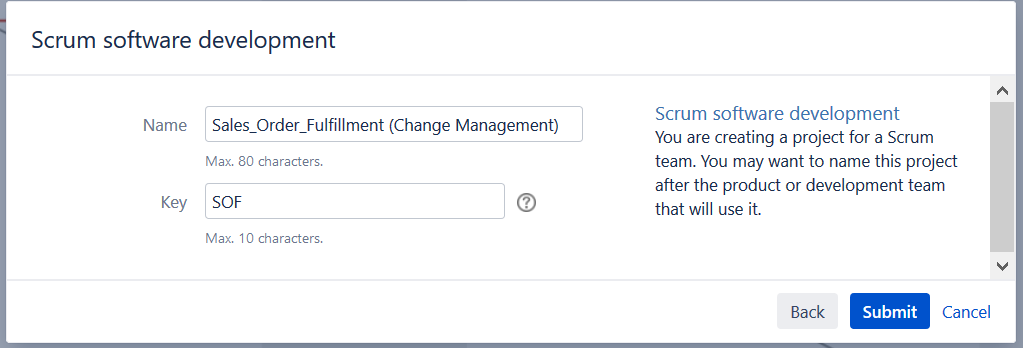
**Please also note that this guide will not cover metadata migration, i.e History (transactional logs) and Attachments. Migrating Links are covered in this document but with a workaround approach which will involve manual steps.**

1) Create a Software project in Jira with Scrum Software Development template and call it as Sales\_Order\_Fulfillment (Change Management).







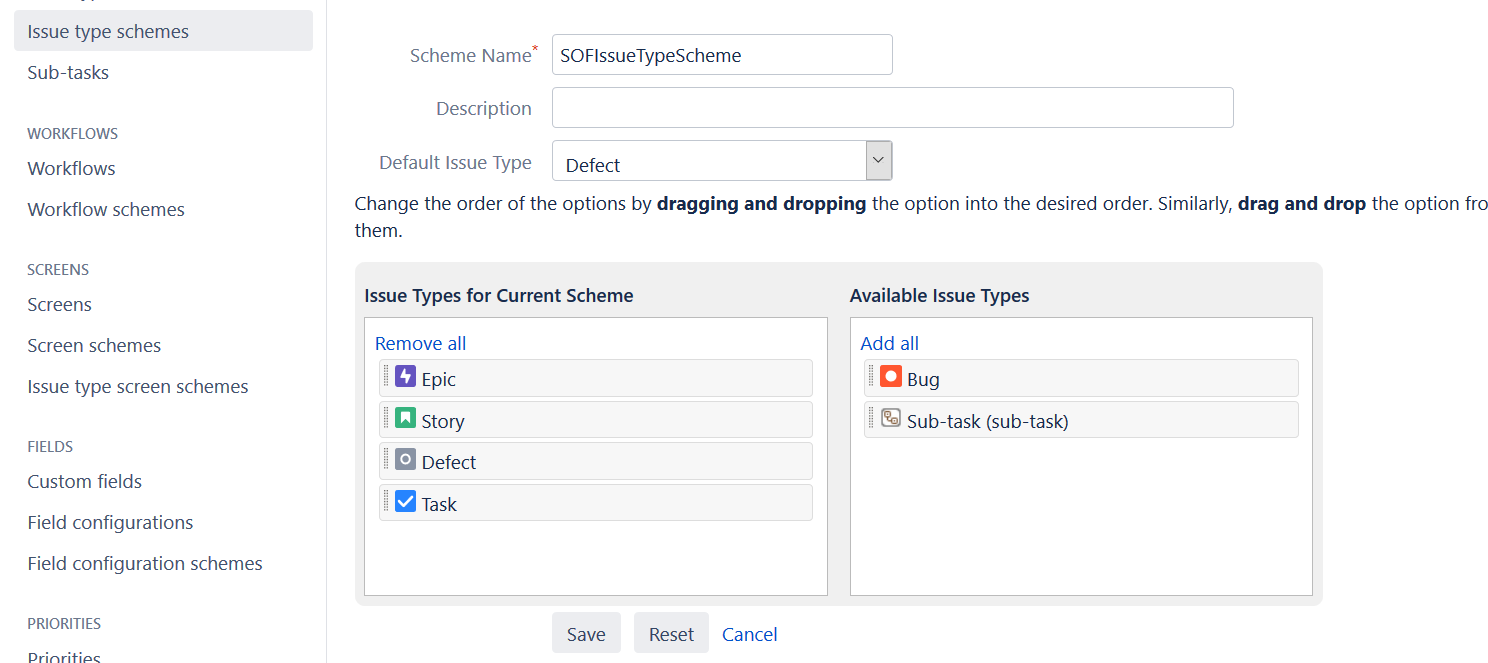


2) Navigate to Administration Page --> System --> and click on Issues Tab

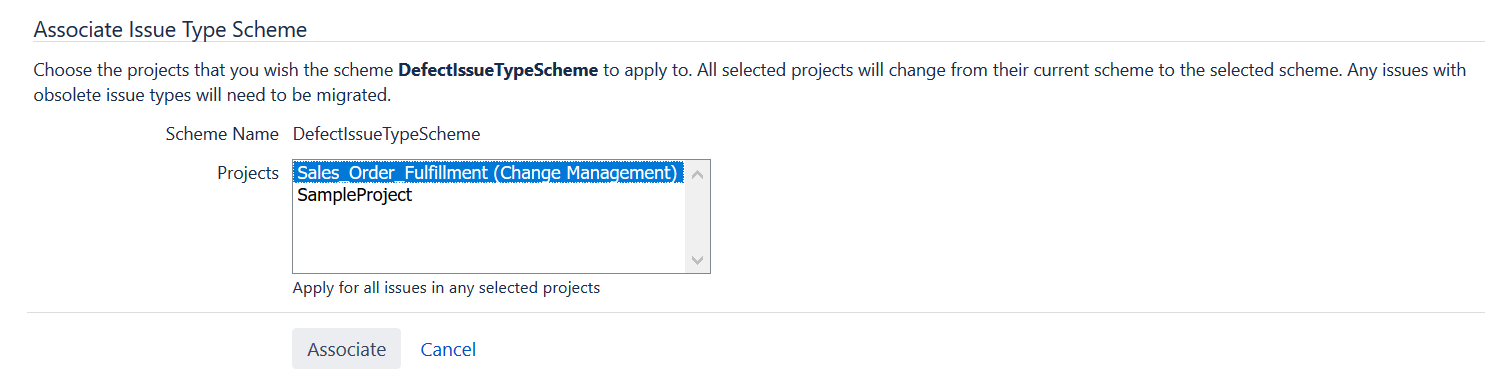
Here we will be presented with several customizable options for Issue Types, Workflows, Screens and Fields

3) Navigate to Issue Types and create a issue type called Defect by clicking on 'Add issue type'  
Similarly, create Tasks, Story, Features, Risk, Release, Impediment, Project Change Request, PI Objective, Retrospective, etc as you require

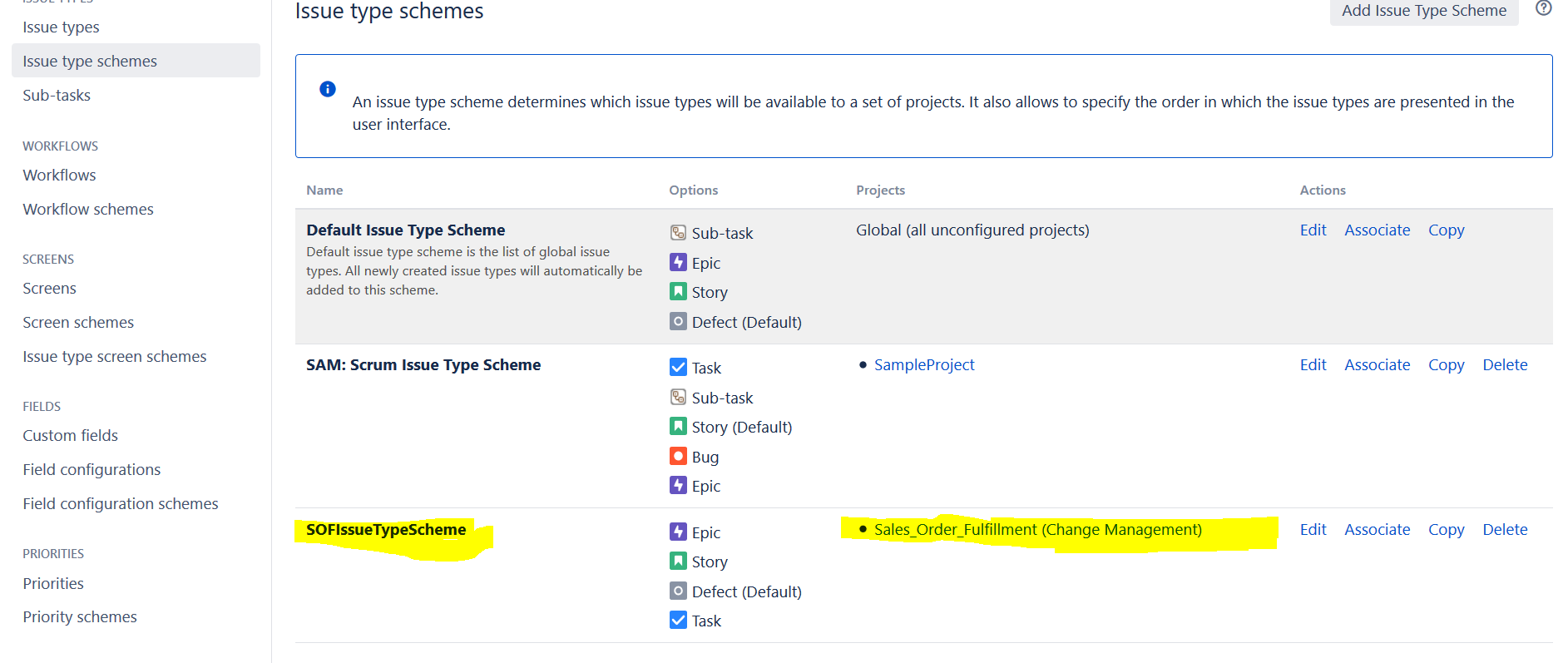
4) Now create the issue type scheme. Issue type schemes will hold the issue types. Give it the name SOFIssueTypeScheme and add Defect, Story, Epic and Task issue types under this issue type scheme



5) Now click on Issue Type Schemes and it will display the Scheme you just created in the above step, but it won’t be associated to the project. Associate it with the project by clicking on the Associate button next to the Issue type Scheme name



The screen should look something like this after associating the scheme to the project, but we can add more Issue types to the scheme. For now, we have added only the below shown issue types.

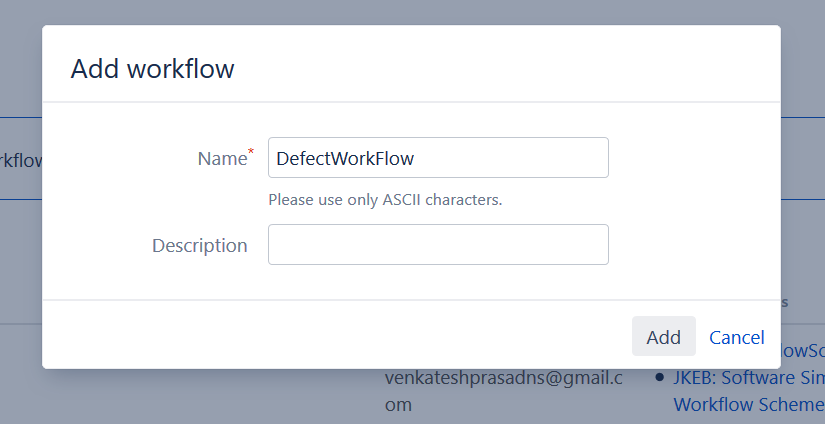


Please also note that you will need to set the “Default Issue Type” in the Issue Type Scheme orelse, we will get the below error while importing issues from excel to jira.

<https://confluence.atlassian.com/jirakb/external-system-import-fails-to-import-csv-file-779160696.html>

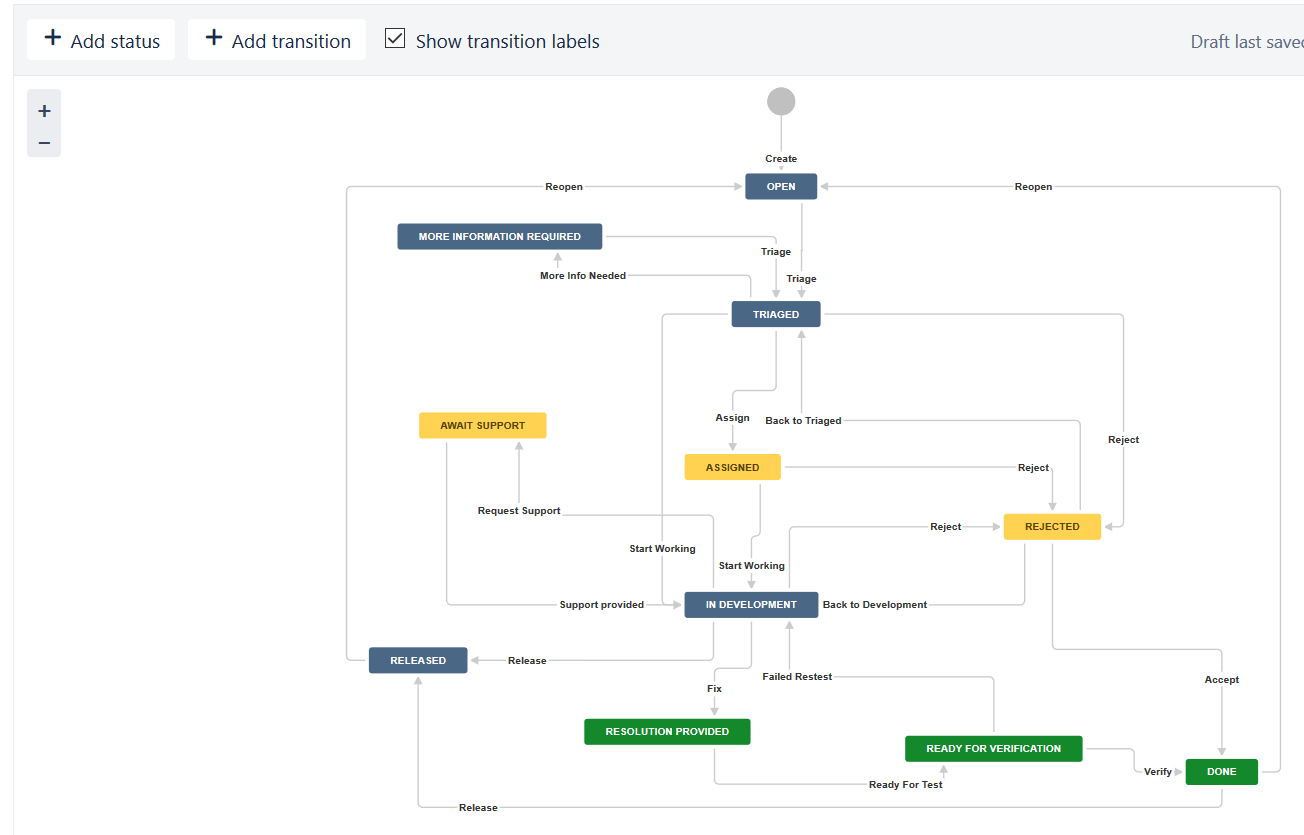
<https://confluence.atlassian.com/jirakb/incomplete-csv-import-when-issue-type-is-not-properly-defined-146407700.html?_ga=2.126538954.619746829.1542012280-705341794.1535370918>

6) Now, add the defect workflow. Click on Add Workflow and give it the name DefectWorkFlow

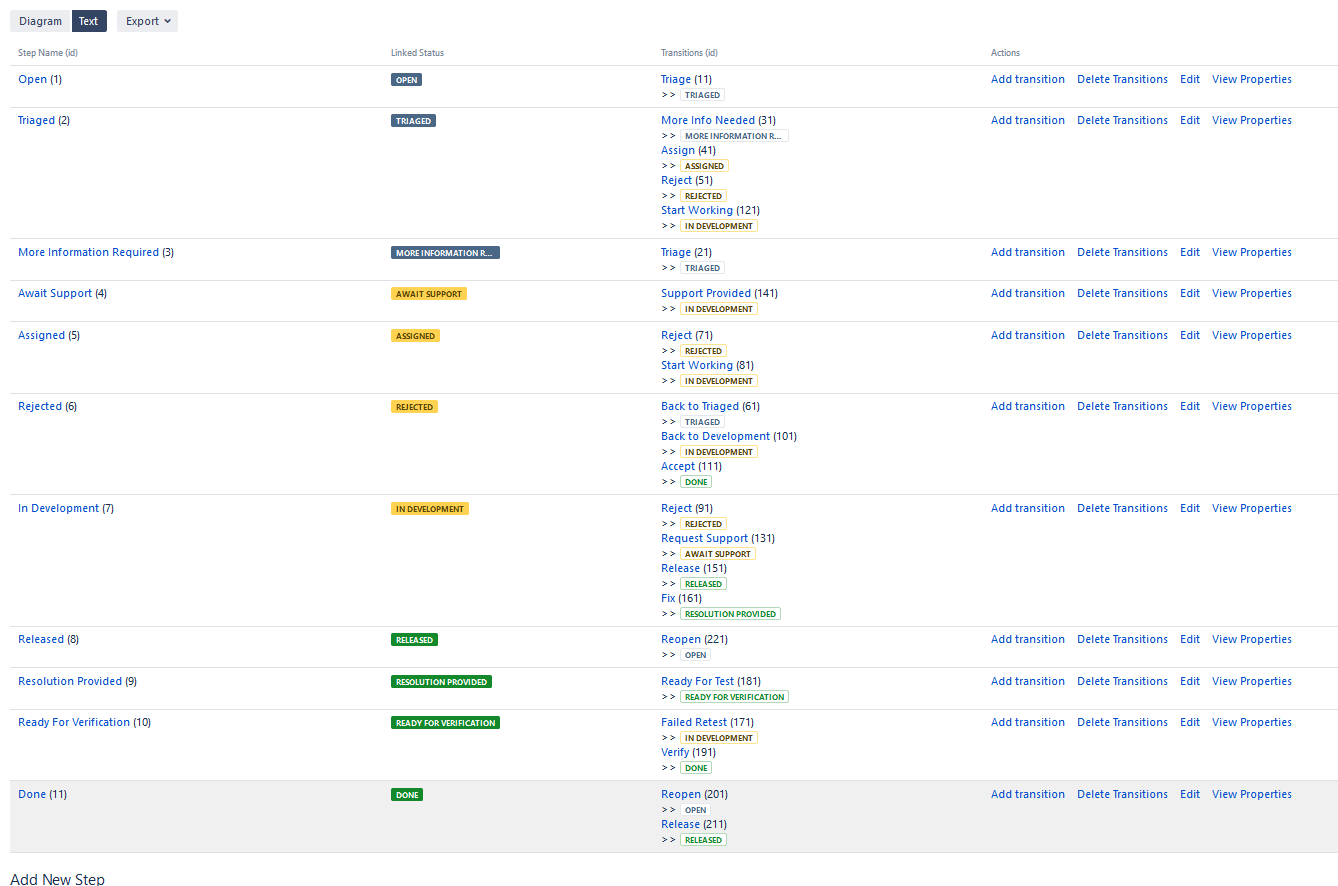


Once we click on Add, it opens up Workflow editor view, where you can create the custom Statuses and add custom Transition states. Once you are done with creating the required workflow, the workflow diagram should look something similar to this.

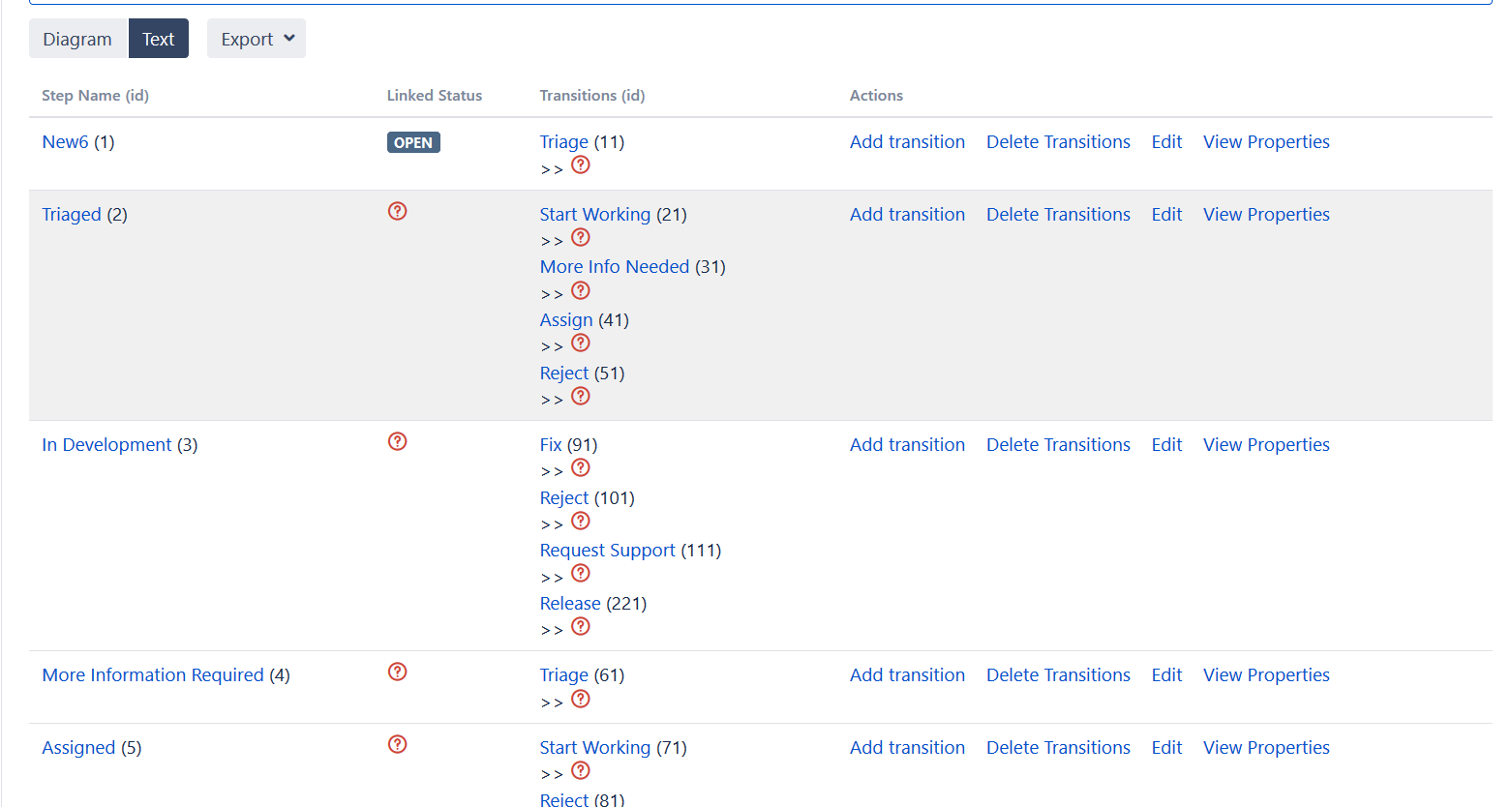
Diagram Mode of the workflow:



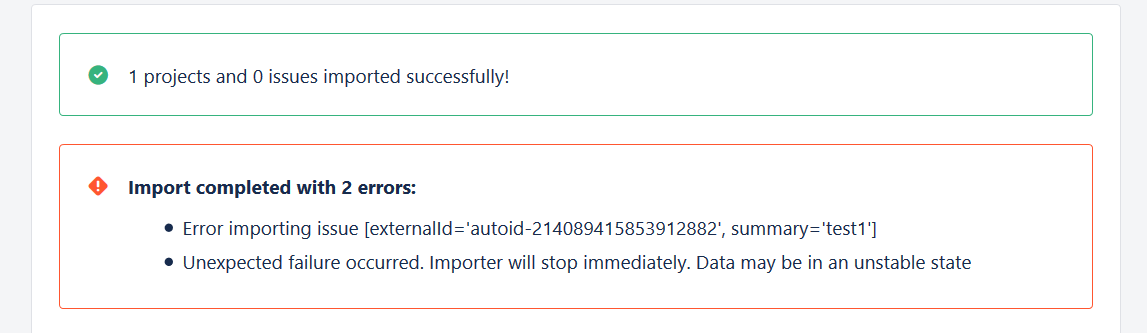
Text Mode of the workflow:

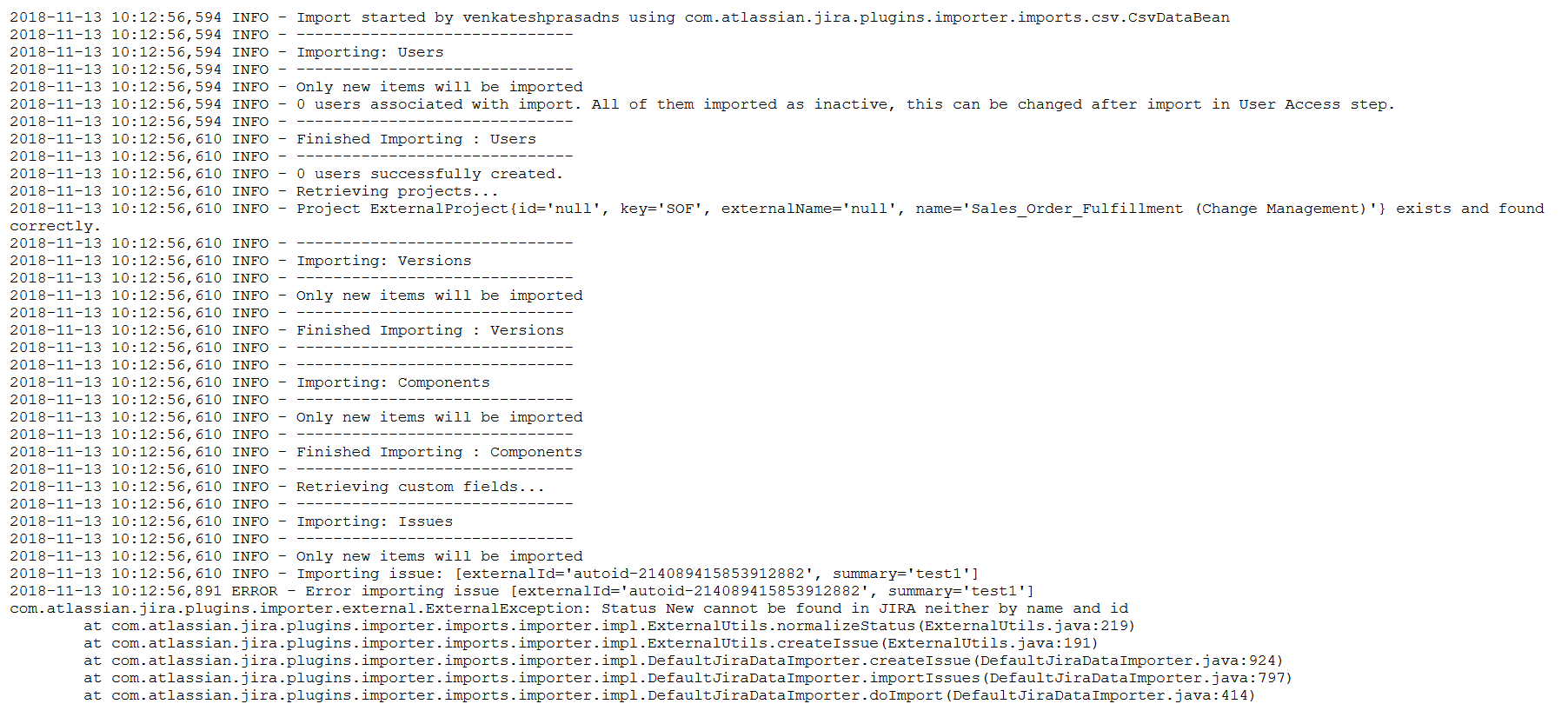


Please keep in mind that you need to click on the option ‘Text’ from ‘Diagram’ mode and see if there are any errors (If there are errors, you will see something similar to the below screenshot which has question marks )



If yes, we need to first fix all those. Else, we will get the below error while running CSV import

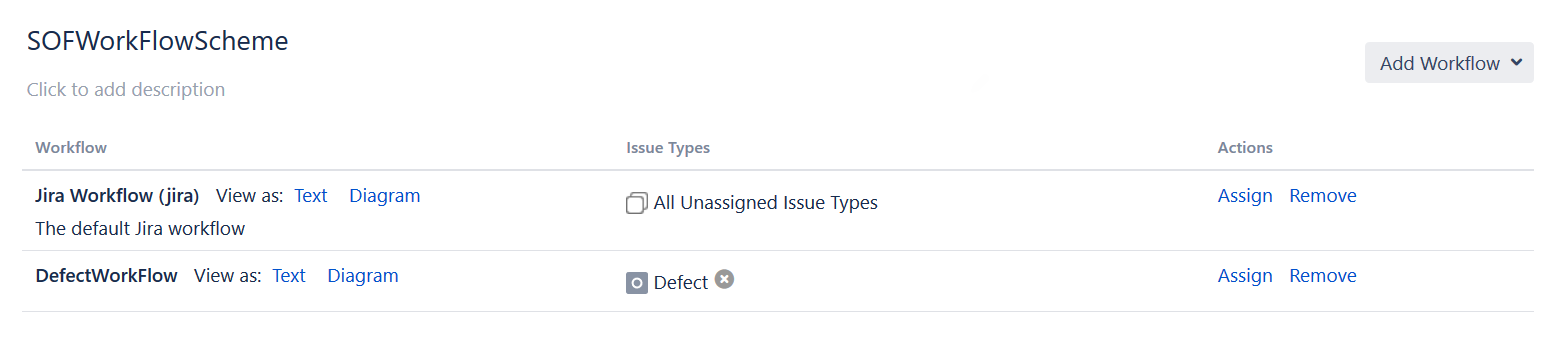




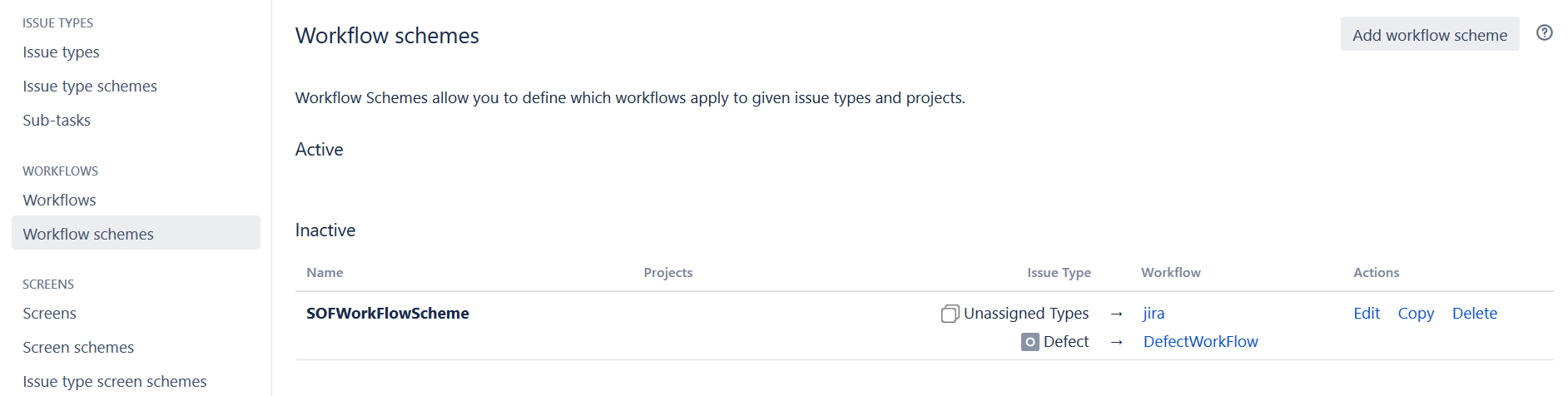
7) Now, we have to define the Workflow Schemes which allow you to define which workflows apply to given issue types and projects.

Click on ‘Add workflow scheme’ and give it a name ‘SOFWorkFlowScheme’ and click on Add

SOFWorkFlowScheme screen opens. Now, click on ‘Add Workflow’ and assign ‘DefectWorkFlow’ workflow to ‘Defect’ Issue Type and assign ‘Jira Workflow (jira)’ workflow to the rest of the Issue Types, as shown in the below screenshot.

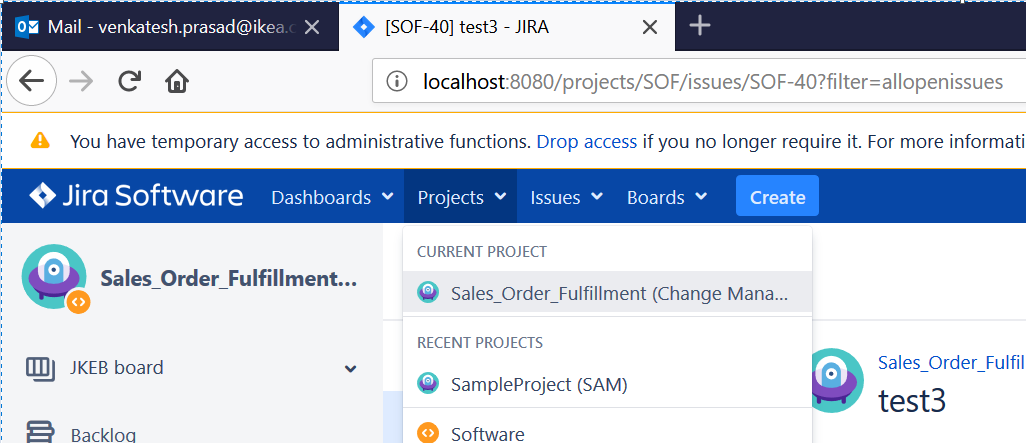


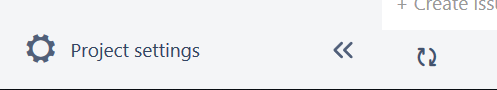
Now, this work flow scheme will stay in the Inactive workflow scheme section.



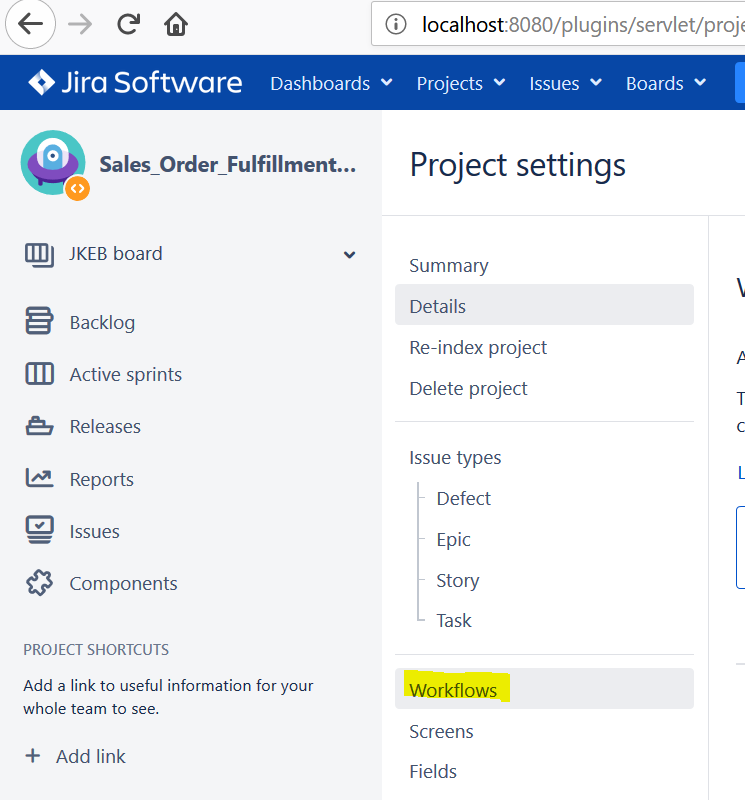
For it to become active, we will have to assign this workflow scheme to a project

To do this, we need to navigate to the Sales Order Fulfilment project and click on Project Settings which will be on the extreme bottom left of the screen.



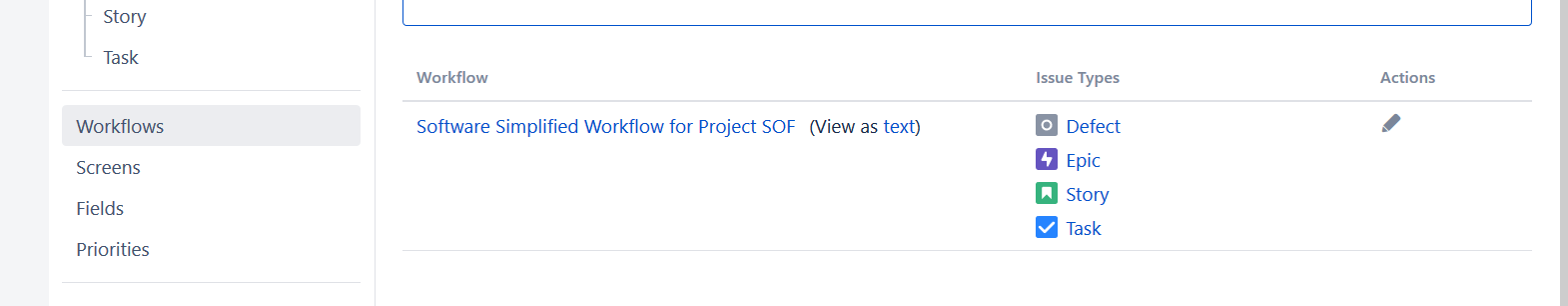


Now click on Workflows

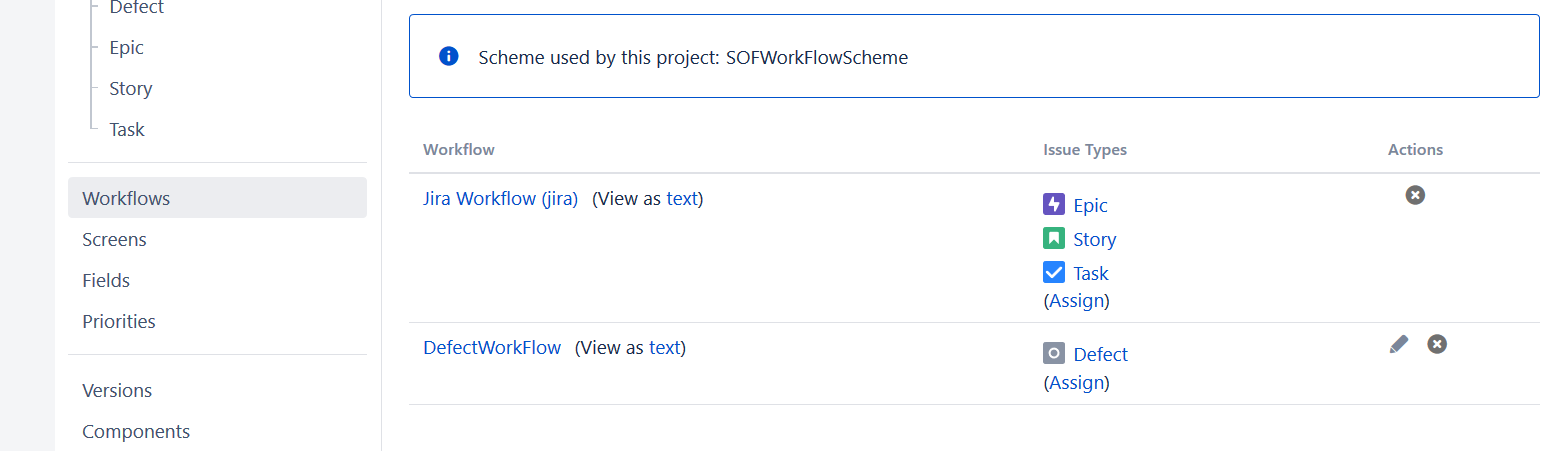


And assign the newly created scheme to this project. For doing this, click on ‘Switch Scheme’ and then Associate the SOFWorkFlowScheme to the Project (By default, Software Simplified Workflow for Project SOF scheme would have got assigned to this project)

Before Switch Schema



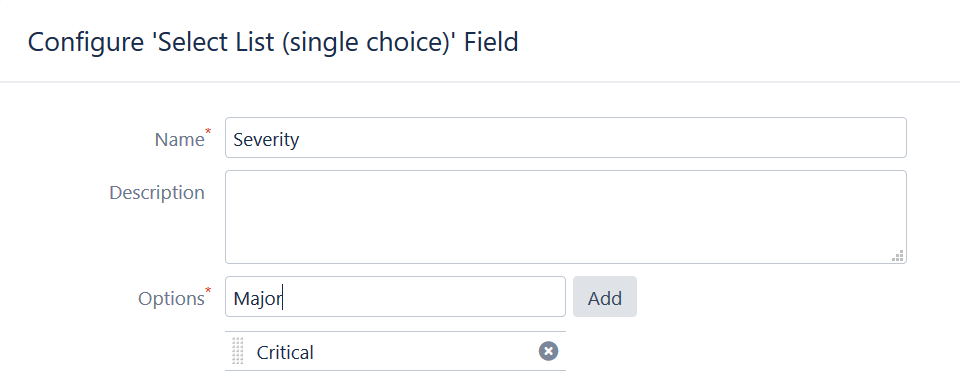
After Switch Schema



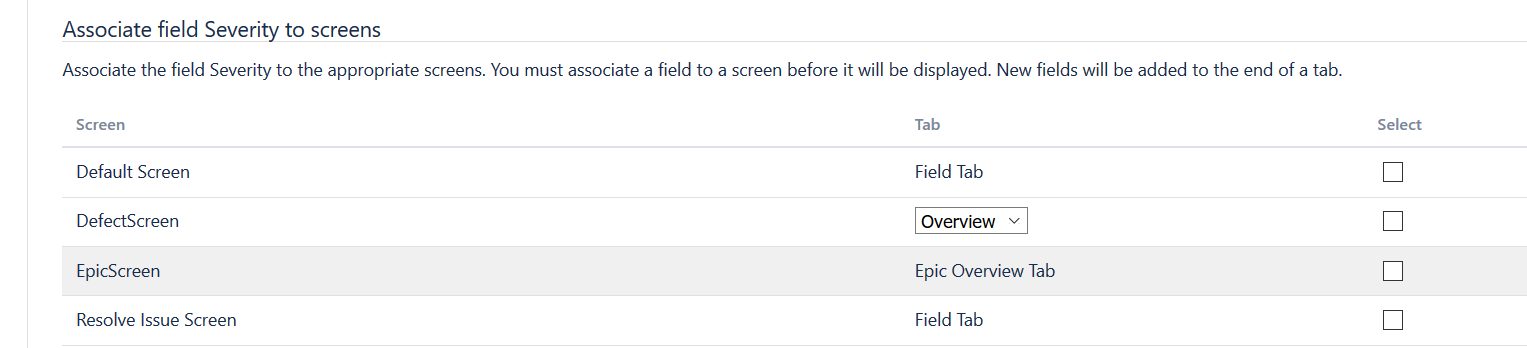
8) We have to also create Custom Fields in Jira, similar to fields existing in RTC (Rational Team Concert), and then use those Custom fields to configure Screens and then use those screens to the Issue Items in the project.

So, we can start by creating Custom Fields. Navigate to ‘Custom Fields’ option and click on ‘Add Custom Field’

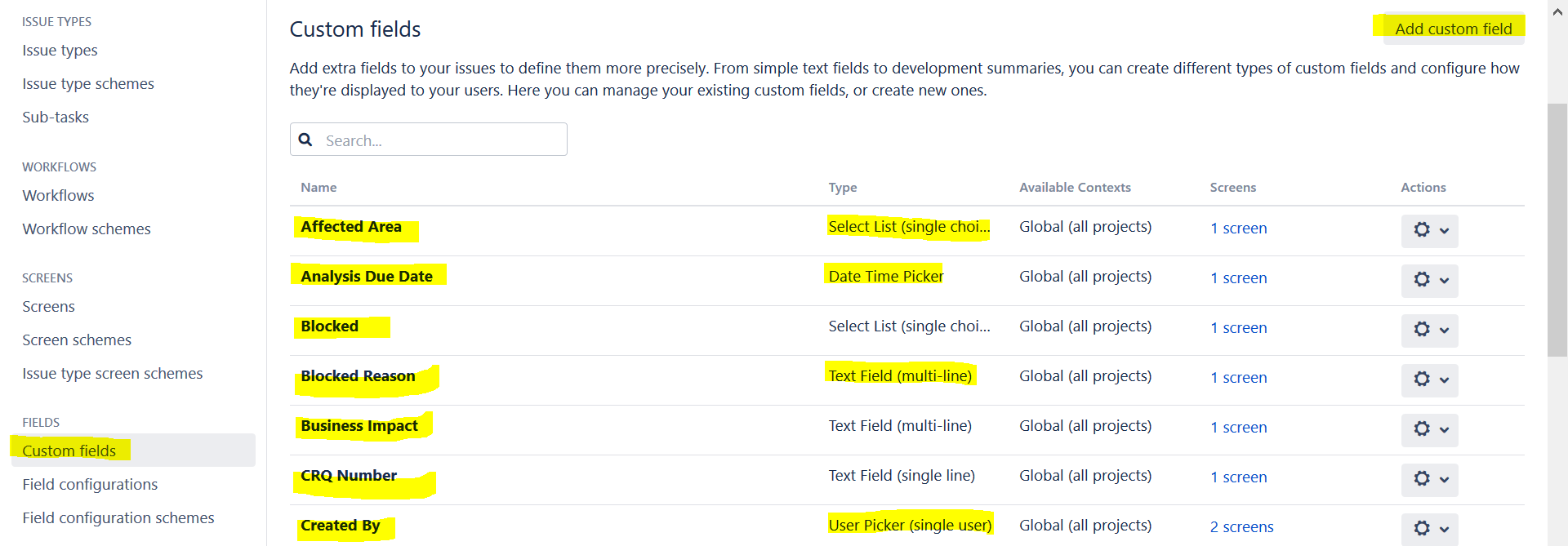
Choose the Field Type, and fill the Name of the Field and other details and click on Create, as shown below.



Now, you will be taken to the ‘Associate the Field to the Screen’ screen. Here you can associate the field with the screens. If you do not have the screens created at this stage, you can create them now. For that, follow point number 12) of this document.



After we create all the needed Custom Fields, we should be able to see a screen like this.



Below is the list of Custom fields seen in RTC which are to be created in Jira.   
Below Fields that are Not highlighted in yellow means that these fields already exists in Jira and hence we do not need to create them in Jira again.

|  |
| --- |
|  |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | |  | RTC Defect Tabs |  |  |  | |  | Overview Tab of RTC Defect : |  |  |  | |  | CLM Fields | Jira Fields | CLM Fields | Jira Fields | |  | Type: | Type | Priority: | Priority | |  | Filed Against: \* |  | Planned For: \* |  | |  | Severity: \* |  | Estimate: |  | |  | Found In: \* |  | Time Remaining: | cannot be pulled by query | |  | Affected Area: |  | Analysis Due Date: |  | |  | Environment: |  | Fix Due Date: |  | | Not Required | Solution Manifest: |  | Verification Due Date: |  | |  | Test Activity: \* |  | Resolution Date: |  | |  | Project Area: |  | Failed Retest Count: |  | | Cannot be pulled by  query, but same as Filed Against | Team Area: |  | Fixed By: |  | |  | Creation Date: | Created | Fixed In: |  | |  | Created By: | Reporter | External Issue No.: |  | |  | Tags: | Labels | CRQ Number: |  | |  | Owned By: \* | Assignee | Blocked: |  | |  |  |  | Blocked Reason: | cannot be  pulled by query | |  |  |  |  |  | |  | Description | Description |  |  | |  | Discussion | Comment/Activity |  |  | |

|  |  |
| --- | --- |
| Links Tab of RTC Defect : |  |
| CLM Fields | Jira Fields |
| Attachments | cannot be pulled by query |
| Links |  |
| Also, define each  relationships as one field | Please see below table |

Below is the list of links available in RTC’s Link Tab :

|  |
| --- |
| Affected By Defect |
| Affects Plan Item |
| Affects Requirement |
| Affects Test Case Result |
| Blocks |
| Blocks Test Execution |
| Children |
| Contributes To |
| Copied From |
| Copies |
| Depends On |
| Duplicate Of |
| Duplicated By |
| Implements Requirement |
| Mentions |
| Parent |
| Related |
| Related Artifacts |
| Related Change Request |
| Related Test Case |
| Related Test Case Result |
| Related Test Execution Record |
| Related Test Plan |
| Related Test Script |
| Resolved By (workitem) |
| Resolves |
| Successor |
| SVN Revisions |
| Tested By Test Case |
| Tracks |
| Tracks Requirement |

|  |  |  |  |
| --- | --- | --- | --- |
| Analysis Tab of RTC Defect : |  |  |  |
| CLM Fields | Jira Fields | CLM Fields | Jira Fields |
| Resolution Code: |  | Root Cause References: |  |
| Root Cause Type: |  | Workaround Possible: |  |
| Involved Systems: |  | Solution/Workaround: |  |
| Root Cause Short Description: |  | Business Impact: |  |

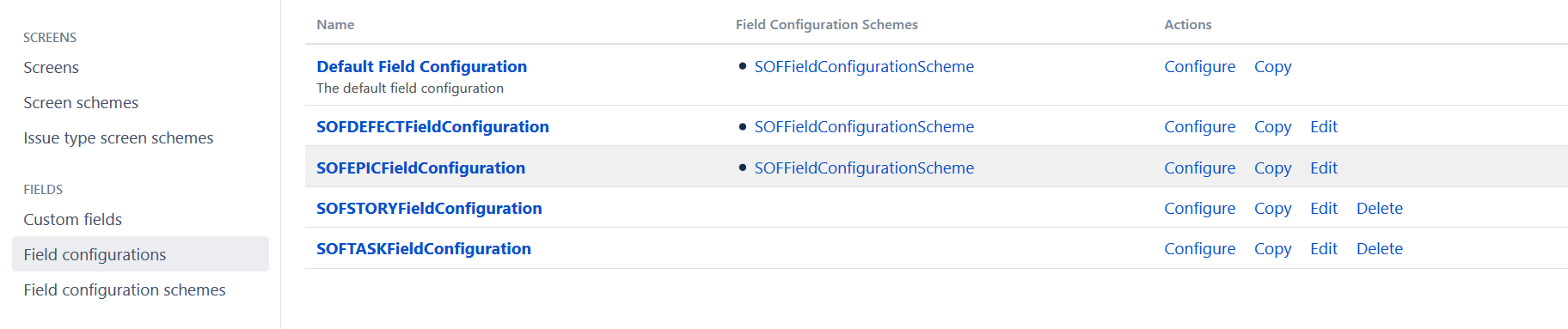
Below Trending tab is not a mandatory tab to be created in Jira. These are calculated fields in RTC

|  |  |  |  |
| --- | --- | --- | --- |
| Trending Tab of RTC Defect : |  |  |  |
| CLM Fields | Jira Fields | CLM Fields | Jira Fields |
| Triage Time (hours): |  | Verification Time (hours): |  |
| Fix Time (hours): |  | Turnaround Time(hours): |  |
| Deployment Time (hours): |  |  |  |

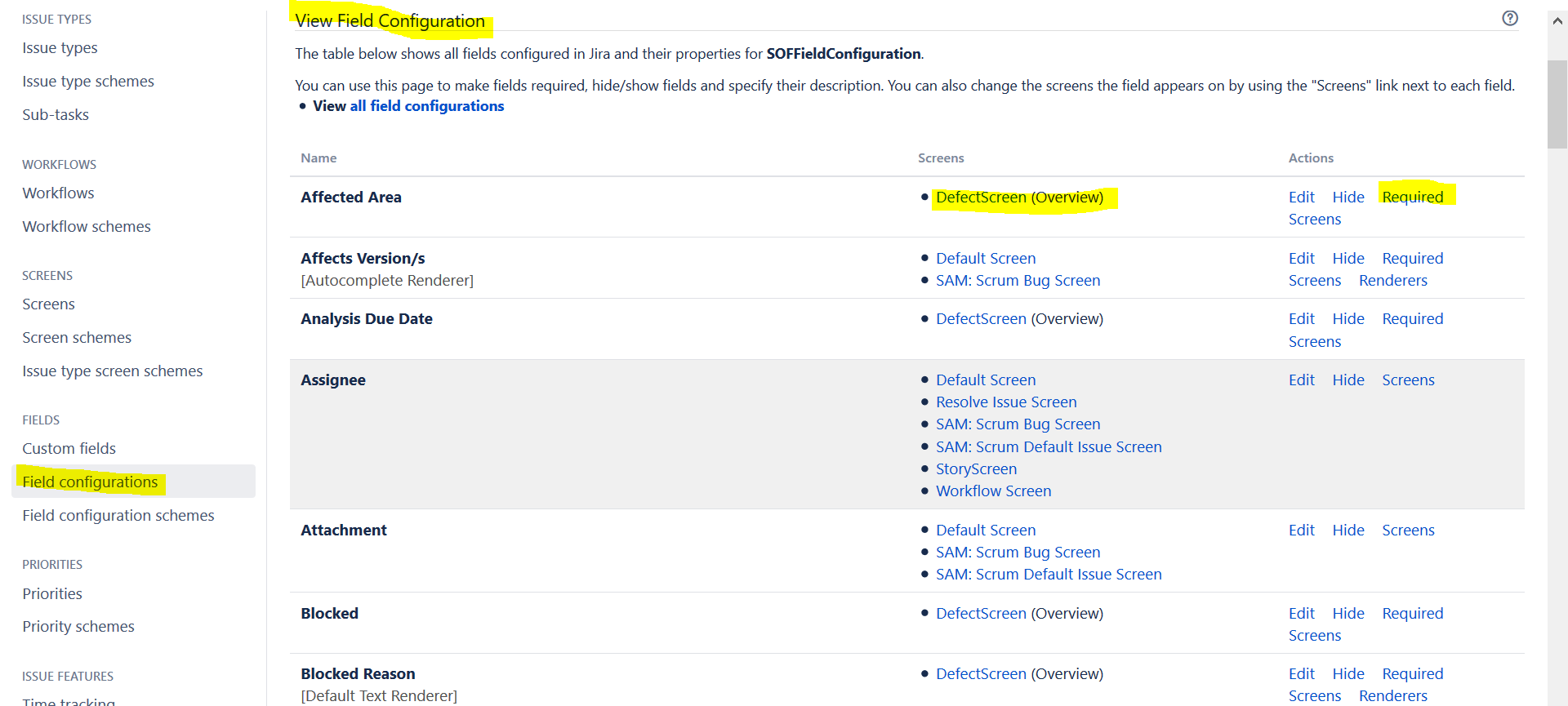
9) Now we have to create Field Configuration. Field Configurations are helpful to hide a field from editor view or to make a field mandator or optional while editing and saving the issue type.

We need to have separate Field Configuration for every Issue type in a project.

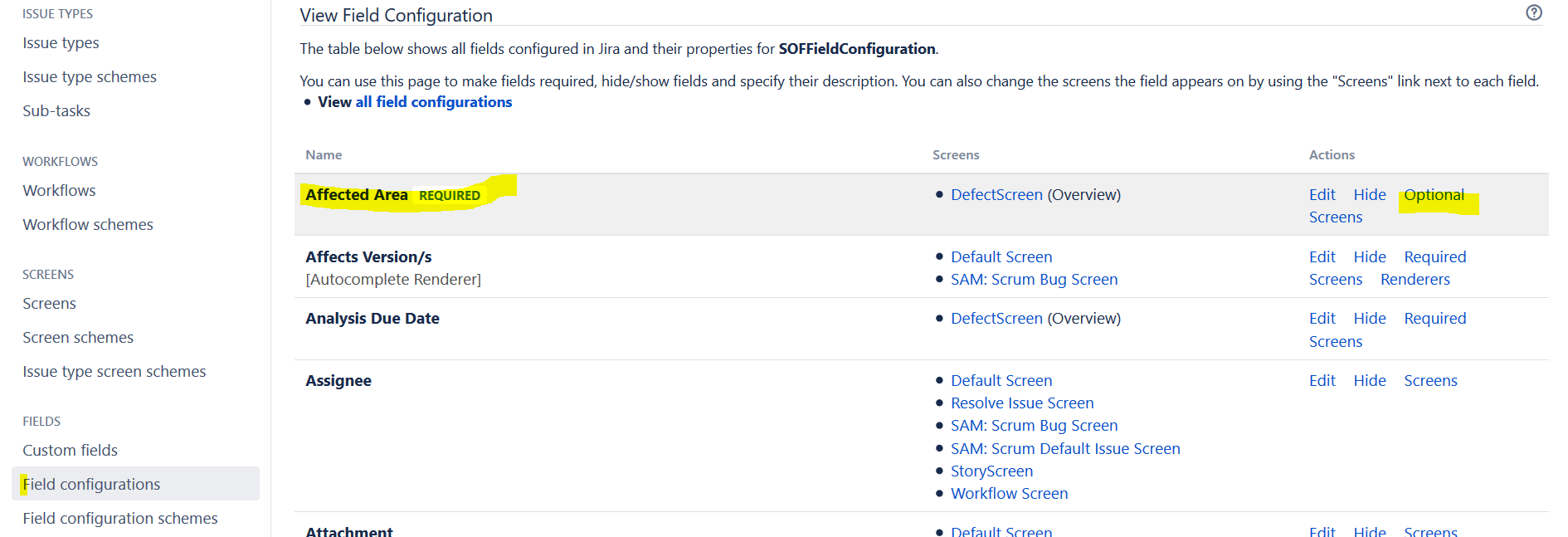
For this instance, we will give the name as SOFDEFECTFieldConfiguration, SOFEPICFieldConfiguration, SOFSTORYFieldConfiguration, SOFTASKFieldConfiguration



In each of the above field configuration creation, we will be taken to the next screen where we will be presented with the list of all fields along with the screens to which it has been tagged to and Actions (Optional or Mandatory field). Please see the below screenshot



You can make a field mandatory by clicking on the ‘Required’ button and the field will change to Required. As shown below.

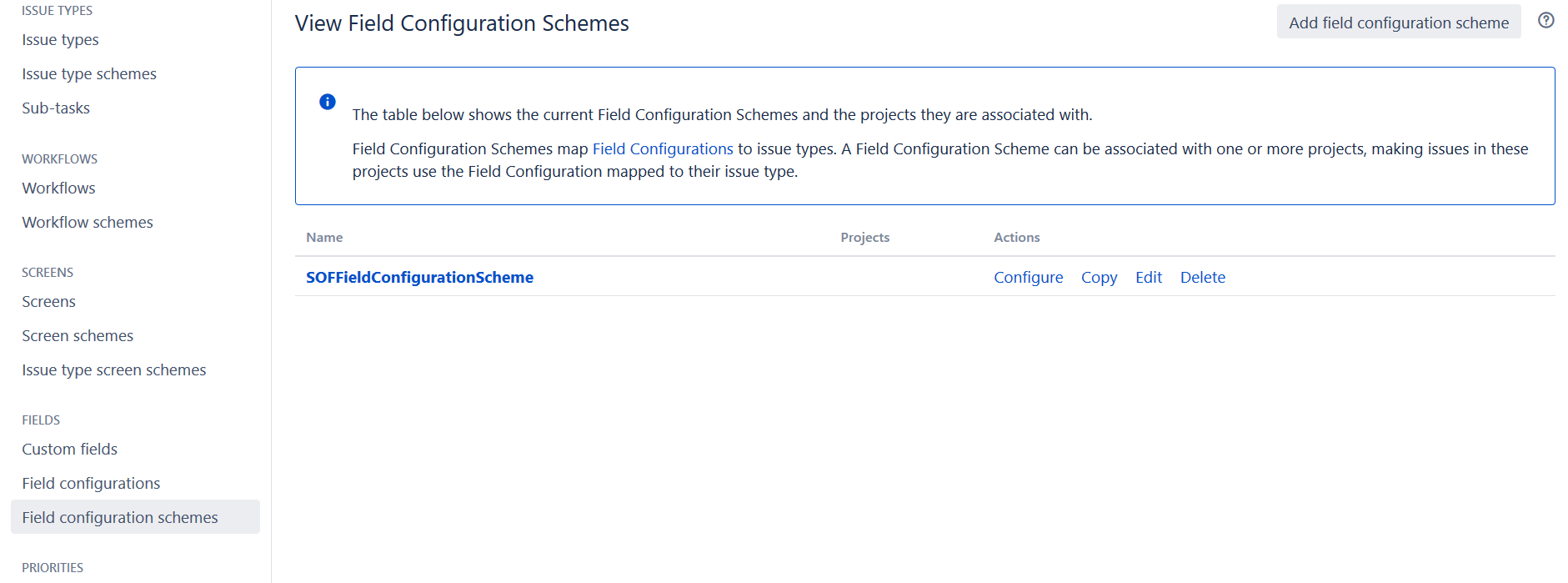


Another important configuration to be performed here is with the Renderers. By default, a Multiline Text Custom Field will be “Default Text Renderer”. This has to be changed to “Wiki Style Renderer” if you want the wiki rendering text to be applicable while performing the excel to jira imports. This is because the \\ will only be rendered as a line break when the Wiki Style Renderer is applied to the field (as opposed to the Default Text Renderer that is associated by default with custom text fields). Without the Wiki Style Renderer, you'll just see \\ in the JIRA field after the import, if the CSV had \\ in it.

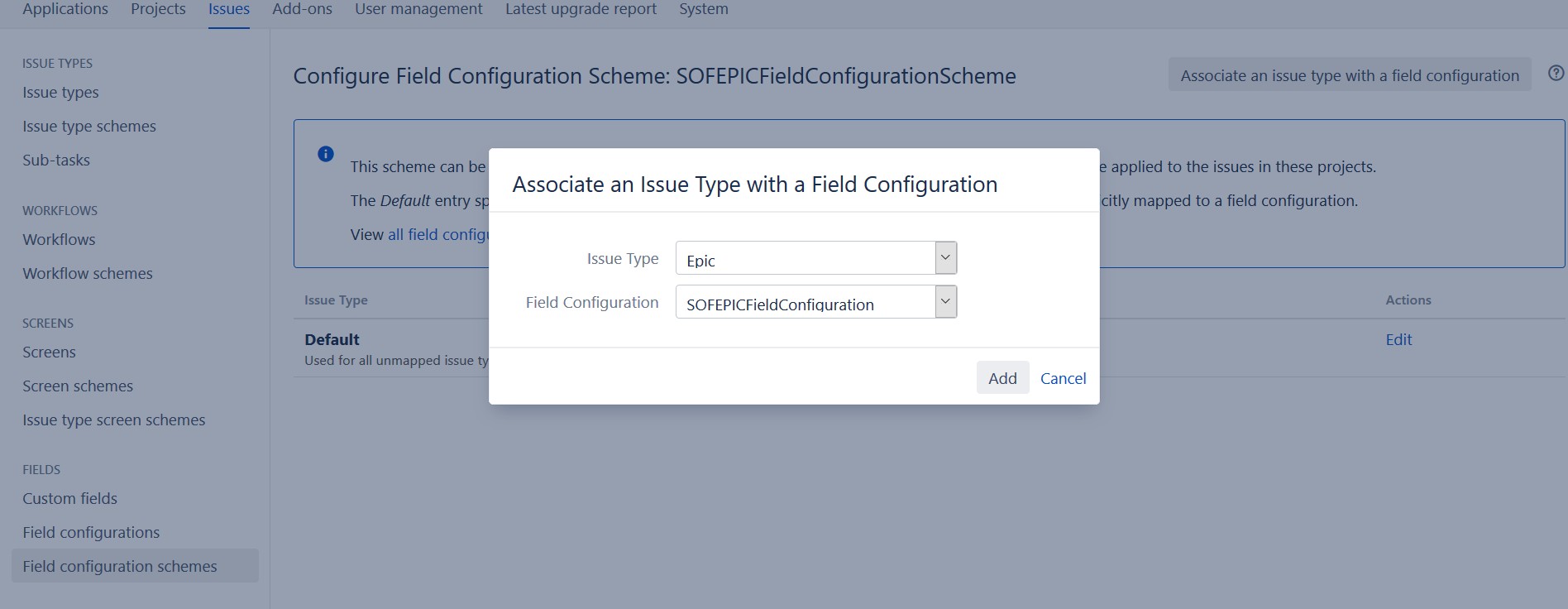
See this link for more details.

<https://community.atlassian.com/t5/Answers-Developer-Questions/import-CSV-data-as-Jira-issues/qaq-p/541576?_ga=2.202259154.2118889702.1544013960-705341794.1535370918>

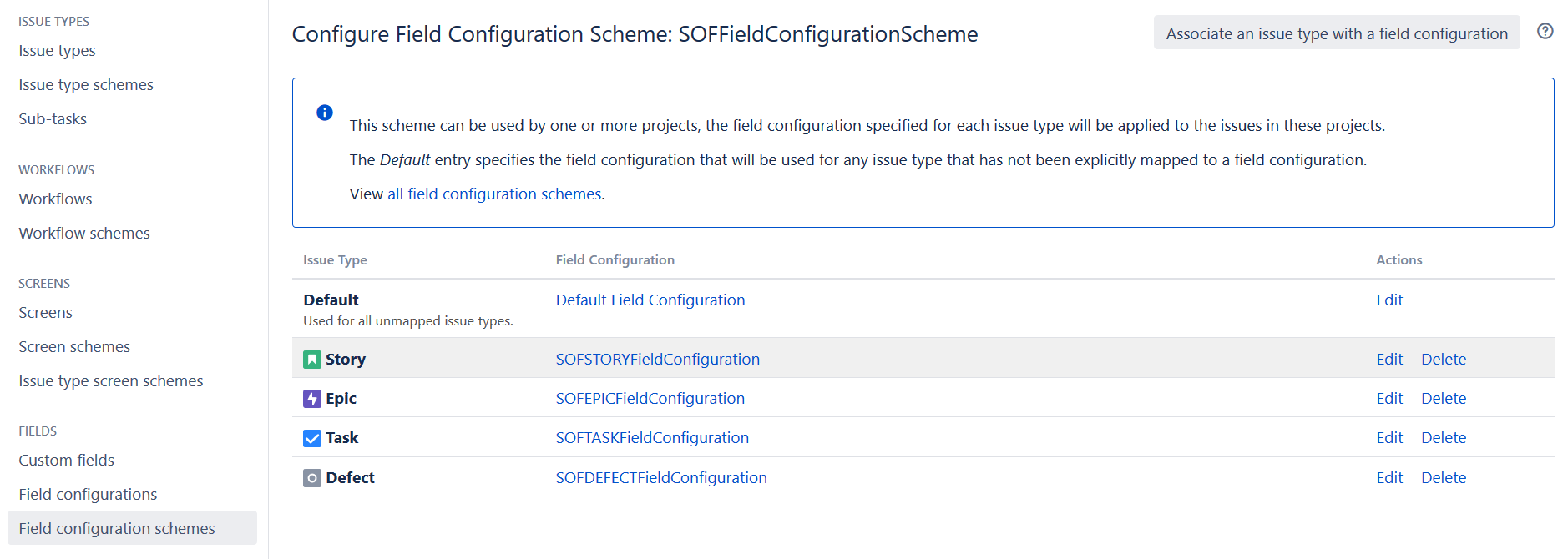
10) Now, this field configuration is to be applied to a project. So we create a new Field Configuration Scheme called SOFFieldConfigurationScheme and tag the SOFFieldConfiguration with the newly created scheme. And then use this scheme in the Sales Order Fulfilment project.



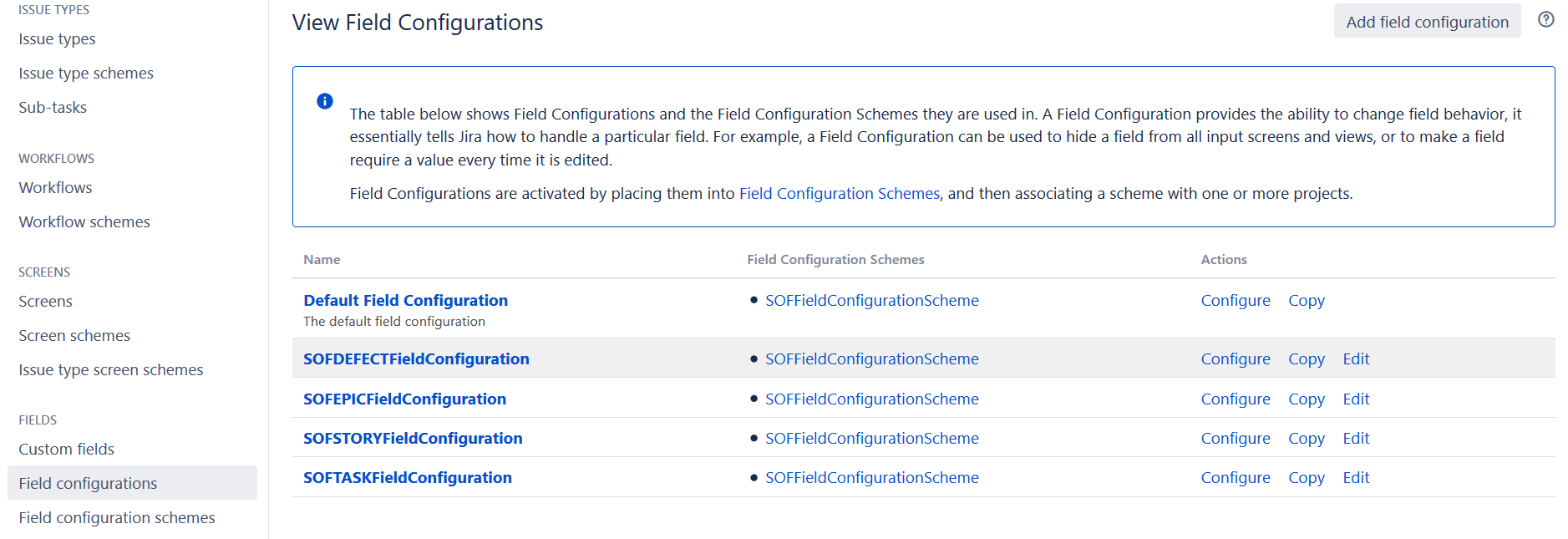
Give the name as SOFFieldConfigurationScheme and click on Add, it will take you to the next screen where you can Associate an Issue Type with the Field Configuration



Once we associate all the required work items to the scheme, then the screen should look like below.

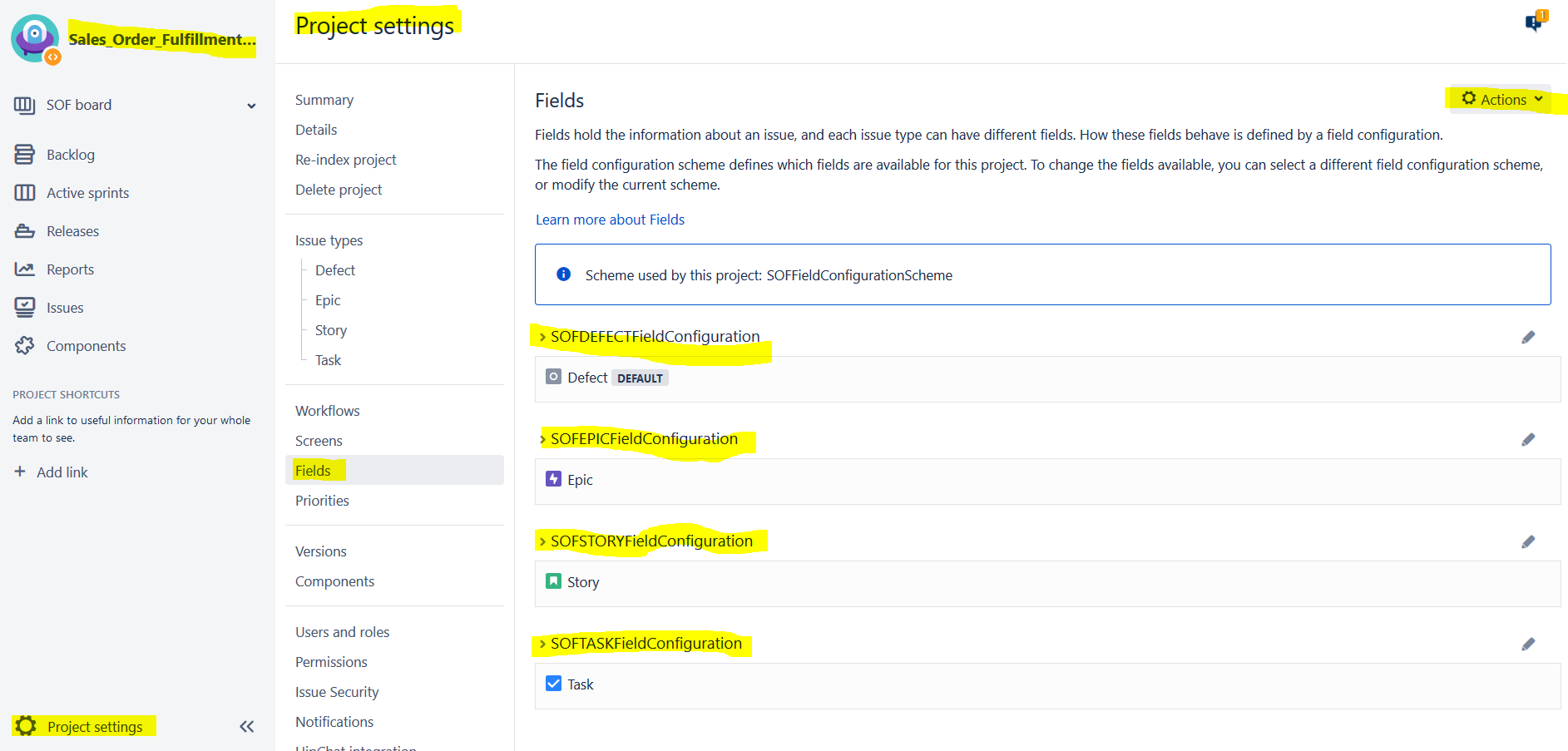


If we now navigate to the field configuration screen, we will be able to see all the Field Configurations tagged to the Scheme.



11) Now go to Project Settings and apply this scheme to the project.

Click on ‘Fields’ section and Click on ‘Actions’ on the top right and select ‘Use a different Scheme’ from the dropdown. Now select the scheme you created in step number 10) above which is SOFFieldConfigurationScheme and click Associate. Now the project screen should look like this.

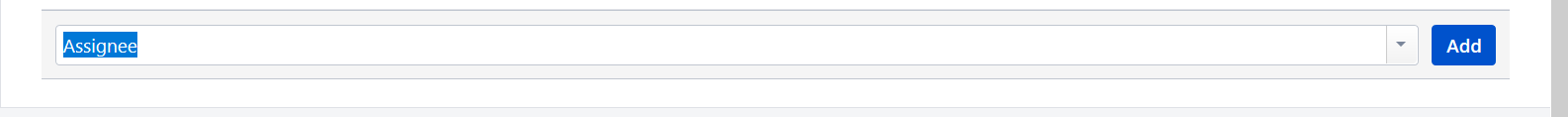


12) We will need to now work on the screen configurations.

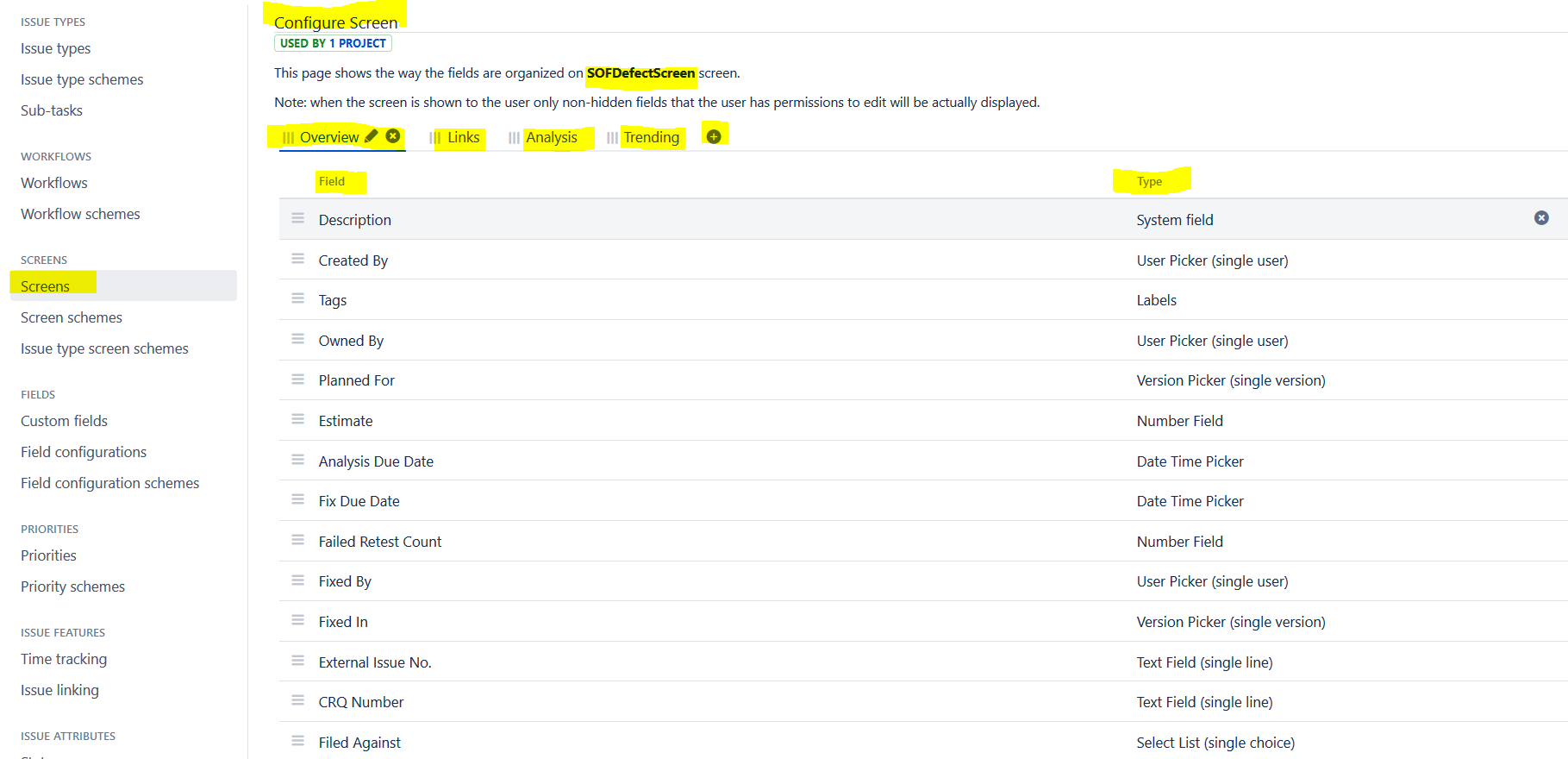
First click on Screens. ‘View Screens’ will display all the existing screens. Click on ‘Add Screen’ and give the name ‘SOFDefectScreen’. Similarly, create for all issue types’’, ‘SOFEpicScreen’, ‘SOFStoryScreen’, ‘SOFTaskScreen’

When you create new SOFDefectScreen, it will then take you to the ‘SOFDefectScreen’ configure screen. There we can create tabs called ‘Overview’, ‘Links’, ‘Analysis’, ‘Trending’ by clicking the + button

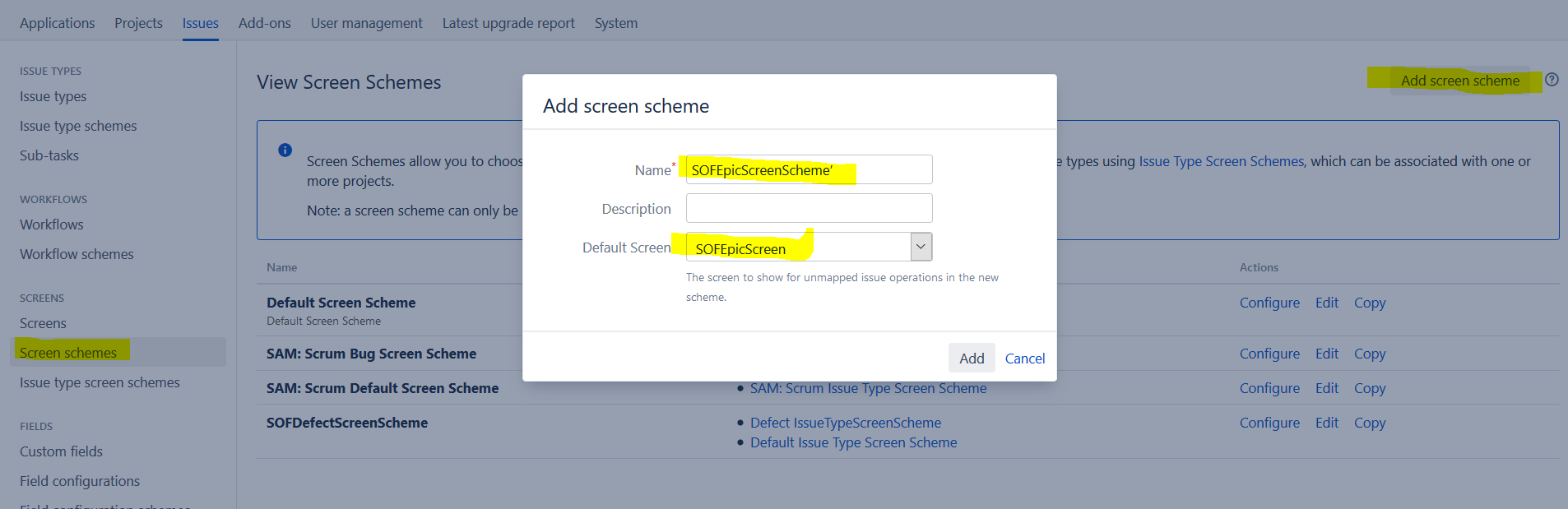
Select the Custom Fields that you created in the previous section in point number 8) and add them one by one and keep adding them to this screen’s different tabs.



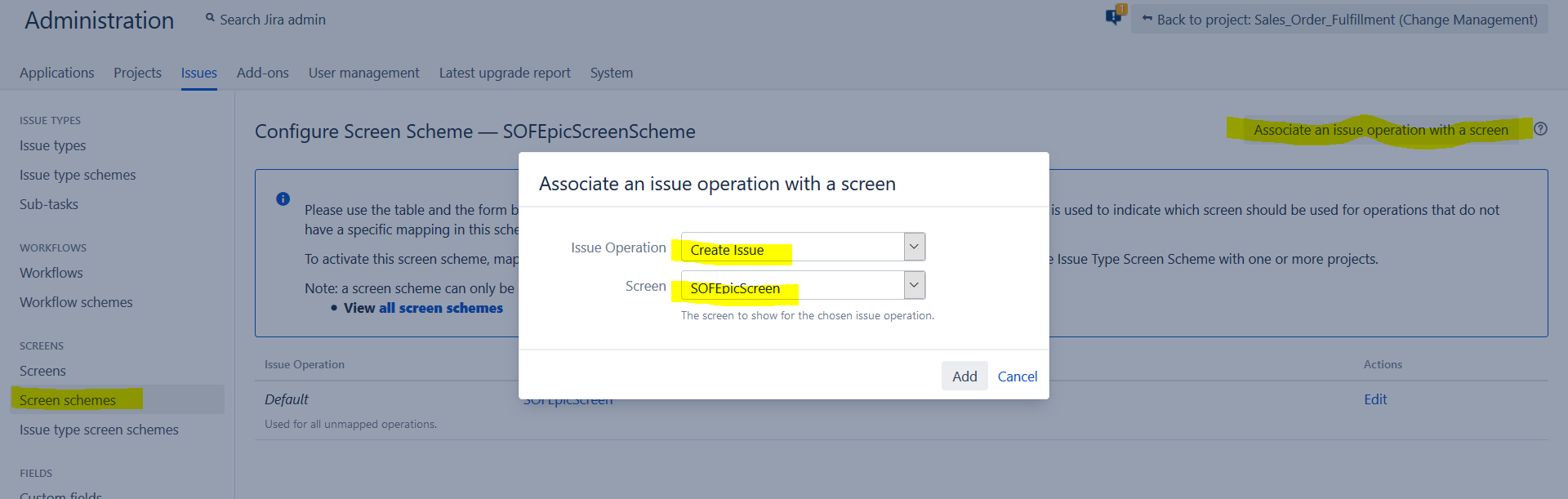
Once done with adding all the required custom fields, ordering it in required sequence, creating all the required tabs, the screen should look similar to this.



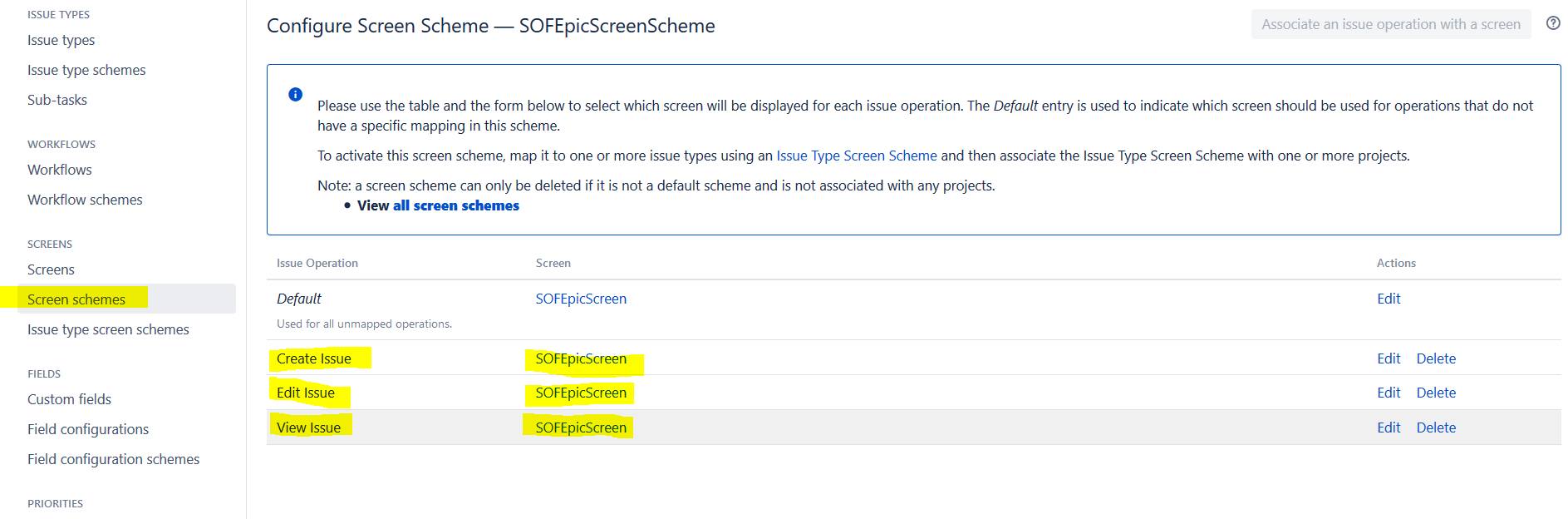
13) Now create the Screen Schemes. Create the Schemes one for each issue type. So, we need to create ‘SOFDefectScreenScheme’, ‘SOFEpicScreenScheme’, ‘SOFStoryScreenScheme’, ‘SOFTaskScreenScheme’



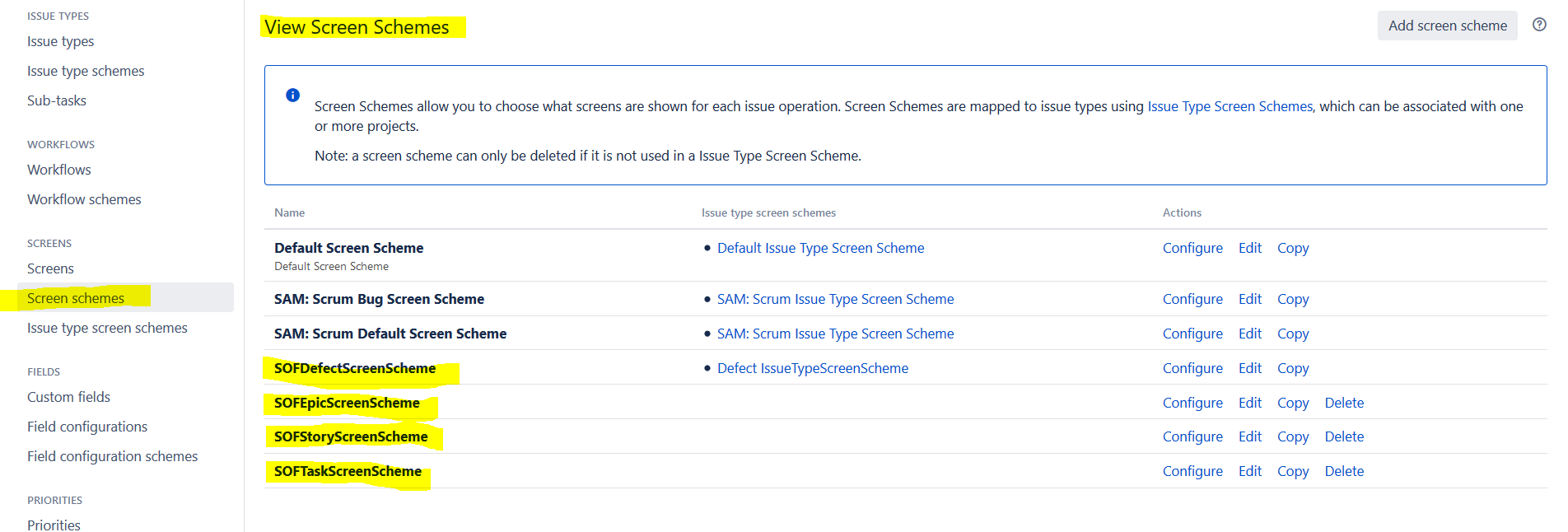
It will then take us to the ‘Configure Screen Scheme’ page. There we can ‘Associate an issue operation with a screen’. Please see below



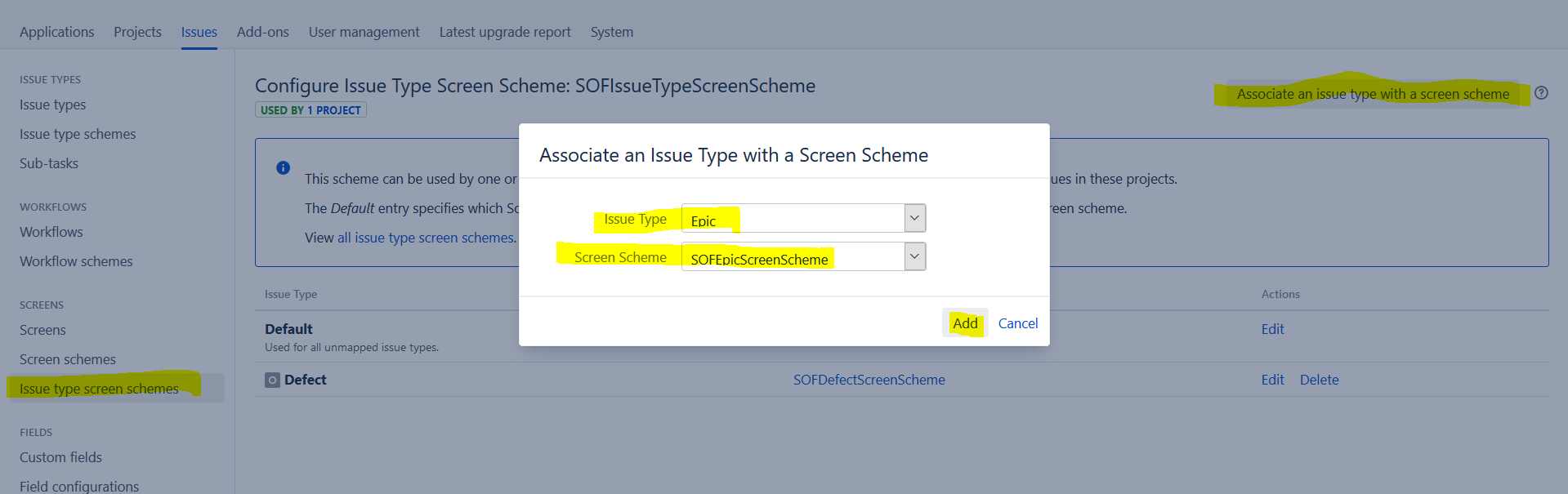
Similarly, do it for Edit and View operations. Once this is done the screen should look like below.



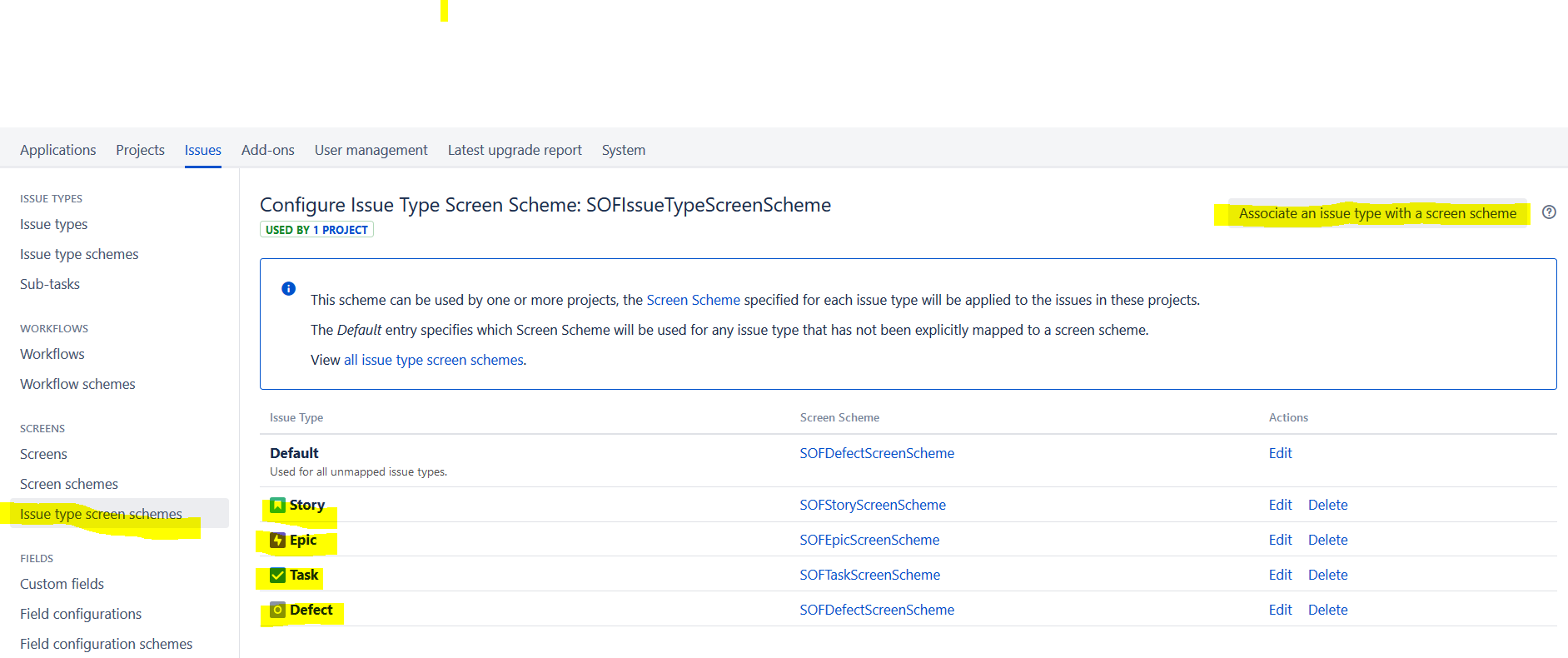
Similarly, do the same for the rest of the screen schemes, i.e for SOFStoryScreenScheme and SOFTaskScreenScheme. Once this is done, the screen should look like this.



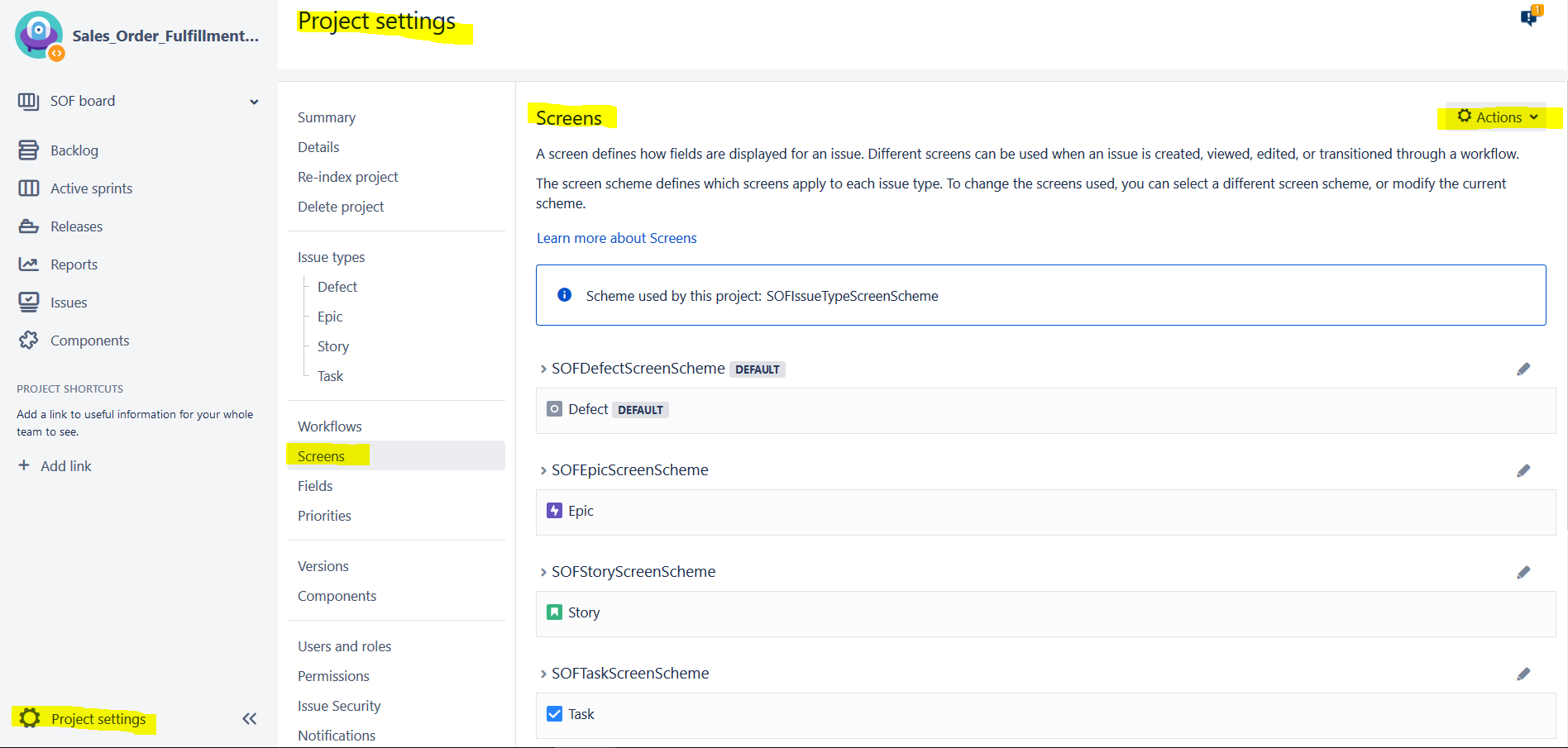
14) Now, create a Issue Type Screen Scheme called ‘SOFIssueTypeScreenScheme’. It will take you to the configure issue type screen scheme page. Here you can associate an issue type with a screen scheme



Once you are done with associating the issue types with the scheme, then the screen should look like this.



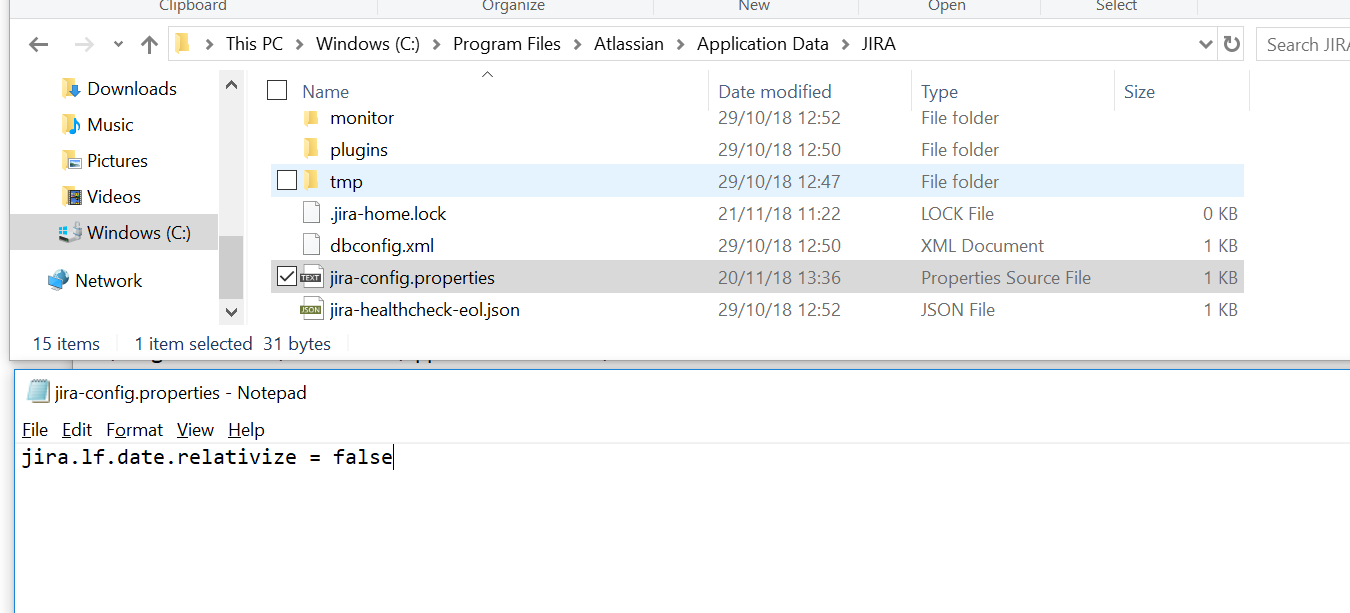
15) Now, associate this Issue Type Screen Scheme with the Sales Order Fulfilment project area. Once done, the screen should look like below.



**Other than this, there are a few more miscellaneous configurations which may be done on the Jira server**

When we browse through the issues in Jira, we can see relative dates in the date fields like Modified date, Creation date, etc. These dates will display as "Today" or "Yesterday" for the next 7 days and then they switch back to actual date format (If you hover the mouse curser over these relative dates, you can still see the actual date). These may not look convenient or be practical for teams to use it. In these cases, we can disable the JIRA application's relative date formatting and simply show standard date formats.

1. To do this, Stop the Jira Application
2. Navigate to Jira Home Directory which will be in the below folder by default  
   C:\Program Files\Atlassian\Application Data\JIRA
3. Edit the file: jira-config.properties using notepad to add the below line  
   jira.lf.date.relativize = false
4. If this file does not exist, we should create it in the above mentioned location.



Configure Character Encoding to UTF-8.

Character encoding should be UTF-8 in CSV File, Destination Jira Server, Application server, db server  
Below Jira Knowledge base has more details and instructions on how to configure this.   
<https://confluence.atlassian.com/jirakb/jira-application-internationalisation-and-encoding-troubleshooting-203394762.html?_ga=2.266811213.2118889702.1544013960-705341794.1535370918>

Pending tests :

Check if the existing Discussion field in Jira can be used for mapping Comments field from CLM

Create CLM Users on Jira before importing, as we are getting errors with user names while running import

**Preparation of CSV Data Sheet**

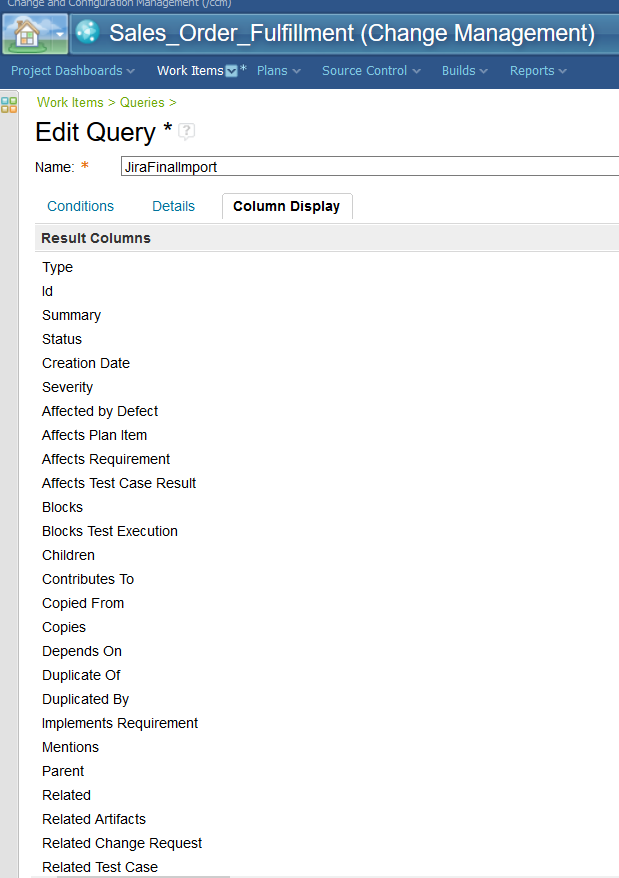
We prepare the queries in RTC and configure all the needed fields in RTC Query and run it. If the output is satisfactory, then we export that into excel. But this will be in the form of .xls or .xlsx . To perform the imports in Jira, jira needs the file to be in the .csv format.

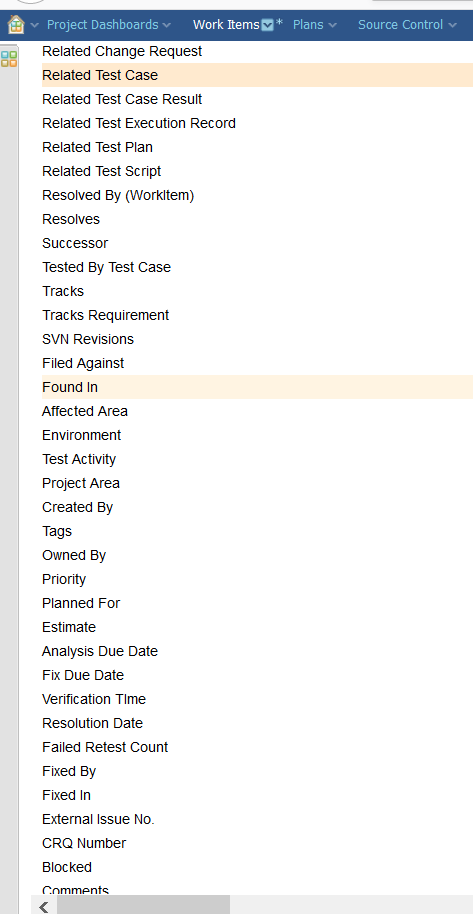
There are several ways to create the .csv data file. The simplest way is to google for several free online tools readily available in the internet where it converts a xls to csv.

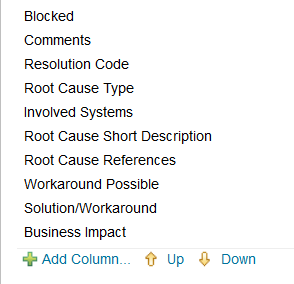
First try to see if the downloaded .csv file can be used as it is with a \t as delimiter while importing into Jira. If there is too much data in the excel, try to split the excel into multiple sheets and then import the data one sheet after the other so trouble shooting is easy.

1) First prepare a query in CLM by navigating to My Query section of RTC and provide the conditions to fetch Defects and then add the below columns in the “Column Display”

For defects, the report should have the following fields in it.







2) Once we have all the required data extracted from RTC into CSV, we can see that the first line of the excel will be the Heading field. This should be left as it is after extracting the data from RTC to CSV. This is the heading line which will be used to map the fields in Jira

3) The data in the CSV file after exporting from RTC will be in a different format which cannot be straight away used for importing into Jira. We need to make several changes to the CSV File. Below are the changes to be made.

4) CSV file will have the date format as “Dec 3, 2018 12:28 PM” when it is extracted from RTC. So, it should be changed to “12/03/2018 ” as we have configured Jira with this date format

Change the date format from Dec 4, 2018 11:17 AM To 12/04/2018 11:17 AM

Do it for all Date Columns. The date columns are Creation Date, Analysis Due Date, Fix Due Date, Resolution Date

5) In the Status column of the CSV file, Search for the Status called New and Replace it with Open. This is because Jira does not have the status called New configured. Jira’s default status of a new work item will be Open. Similarly, if your Jira instance does not have any specific state which is existing in Excel, then that has to be replaced with the Jira’s closest matching state.

Now, copy all these formula columns and paste them into a new excel file. While pasting, do a paste special and select Paste Value or else it will paste along with the formulas which we don’t want.

6) Line break characters are another complication. If this is not addressed correctly, the data in one column gets merged with the neighbouring columns and it will mess up the entire migration.

So, where ever there are MultiLine Text fields, do the following.

Use this formula  
=SUBSTITUTE(B2; CHAR(10); " \\ ")

This will remove all the line break characters and replaces it with \\

You must configure Jira’s Multi LineText field to be a Wiki Renderer Field instead of Plain Text Field.

After importing this field to Jira, Jira converts this \\ into new line inside the Jira field due to the Wiki Style Renderers field that we configured. I have tested this and it works fine, however, a few Jira forums have complained that this is not working fine for them. Please go through the below link  
Also remove links from the csv file before importing or the \\ will get merged with the hyperlink and will not be considered by Jira as line break character.

<https://community.atlassian.com/t5/Answers-Developer-Questions/import-CSV-data-as-Jira-issues/qaq-p/541576?_ga=2.202259154.2118889702.1544013960-705341794.1535370918>

<https://community.atlassian.com/t5/Jira-questions/How-can-I-import-from-CSV-with-carriage-return-and-line-feed-in/qaq-p/322723>

7) There are a few text columns in csv file which may needs special considerations. It is discussed below. However, these are purely requirements based scenarios and there isn’t any thumb rule to follow this strictly. I followed these steps while preparing a POC for migration, but you may ignore these if you feel these are irrelevant.

a) You can just use a | or ~ delimiter instead of a comma delimiter. In page 22 of this document, I have mentioned about using online converters available in the internet. You can use other delimiters instead of comma and then get the data sheet in the required csv format.

But, keep in mind that there could be | or ~ already available in these text fields which users may have typed while putting comments to the defects or stories. So, you may have to first do a Find and Replace all | or ~ with , or ; and then use the online tool to convert excel to csv with | or ~ as delimiter.

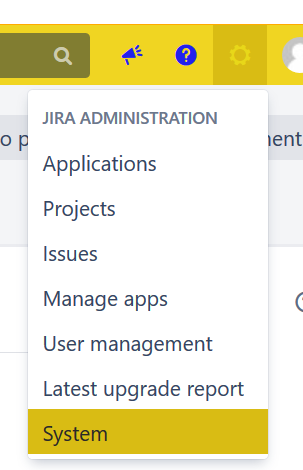
You may also search for “ (Double inverted commas) and replace it with “” in all the text columns in the csv file. This will help if you have the data imported into the Wiki Renderer field as the two double quotes gets converted to one double quote. This is not to be done if you are importing this to a standard text field.

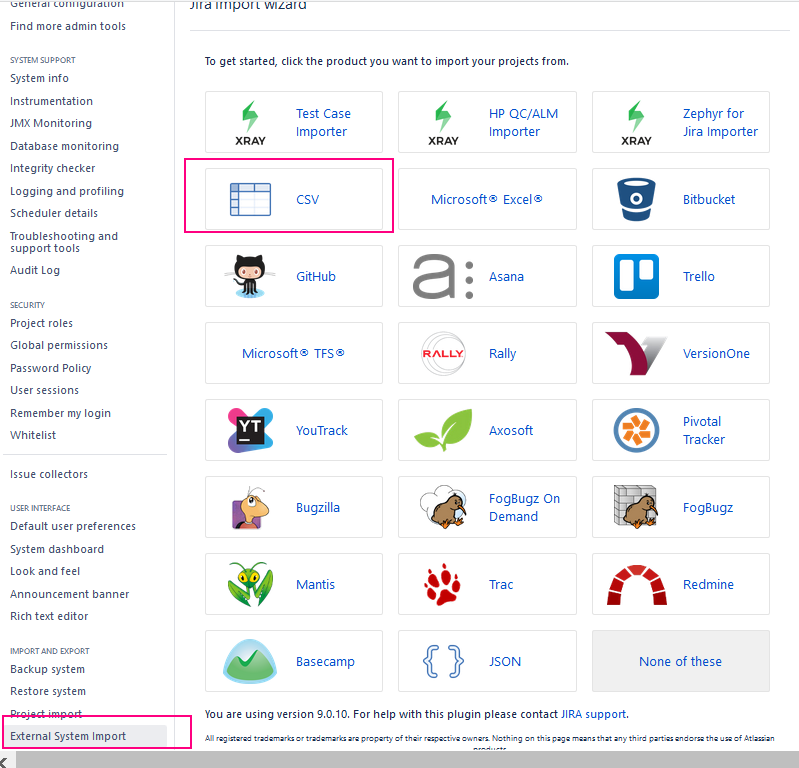
8) Now, for all the text fields, we need to encapsulate the whole text cell in double quotes and then insert the delimiter at the end. This is only if you are importing these text fields into a Wiki Renderer field. To do this, use the formula =""""&A2&""""&","  
Repeat the same for all the text fields like comments and any custom text fields available in the excel data sheet.

Optionally, you may search for ‘ (single inverted comma) and remove it from all the text columns in the csv file.

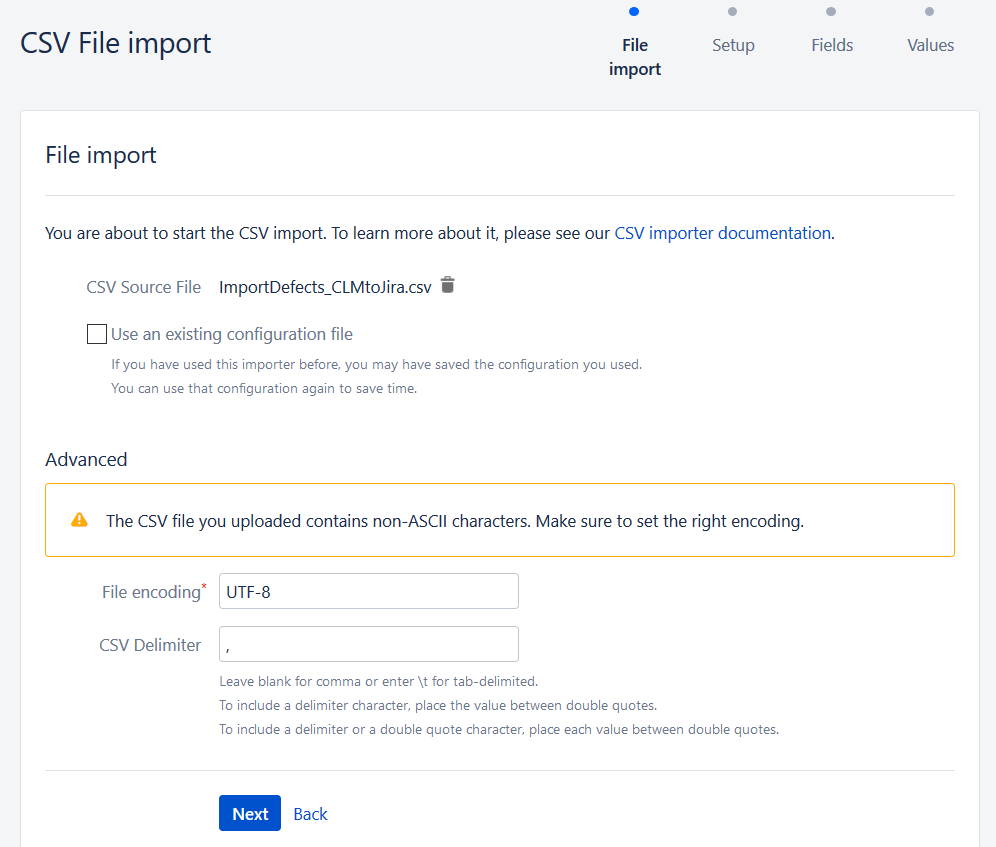
**Import the excel data into Jira**

Navigate to Jira Administration 🡪 System 🡪 Under Import and Export, select External System Import, select CSV

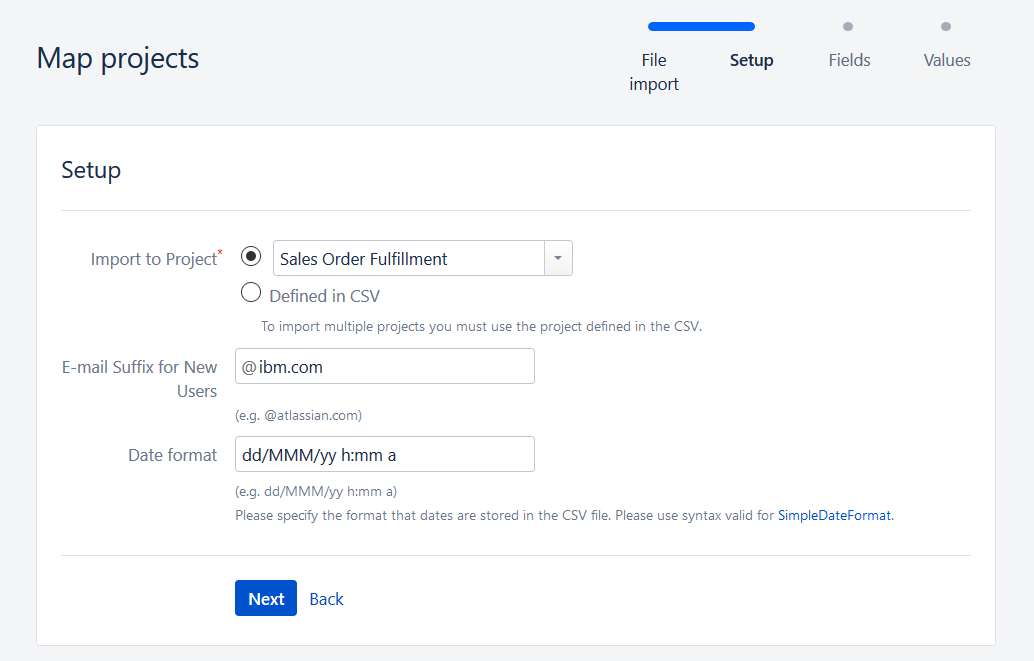




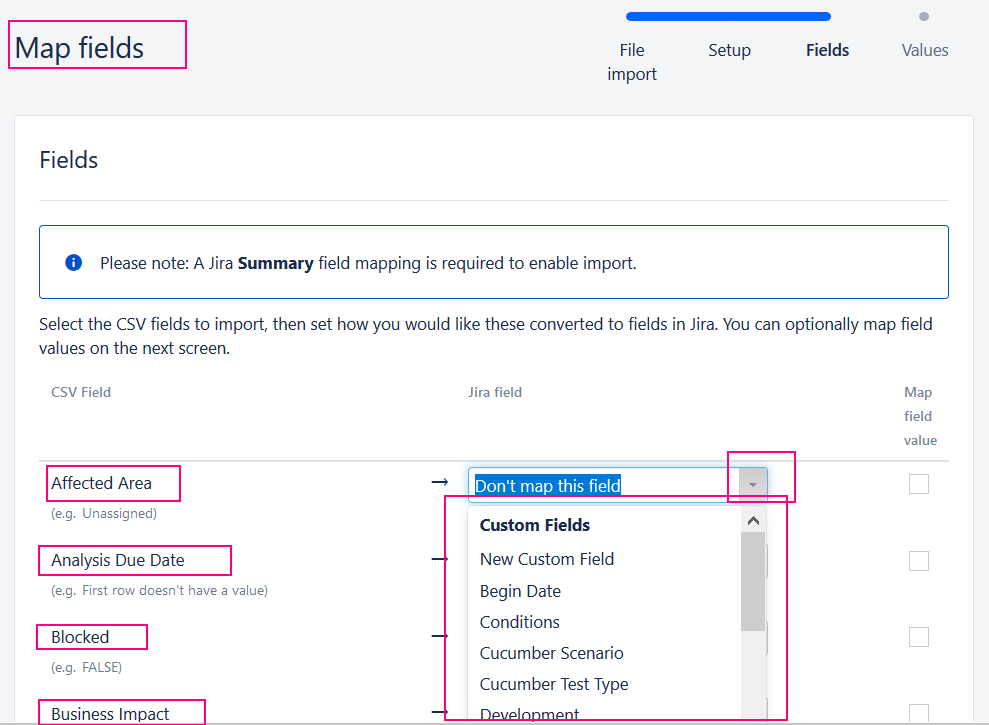
Browse the excel data file in the below screen, select the file encoding as UTF-8, delimiter as , if you have used commas or mention any other delimiter that you have used in excel, and then click next.



Now, select the project, email suffix and date format as shown below and click next

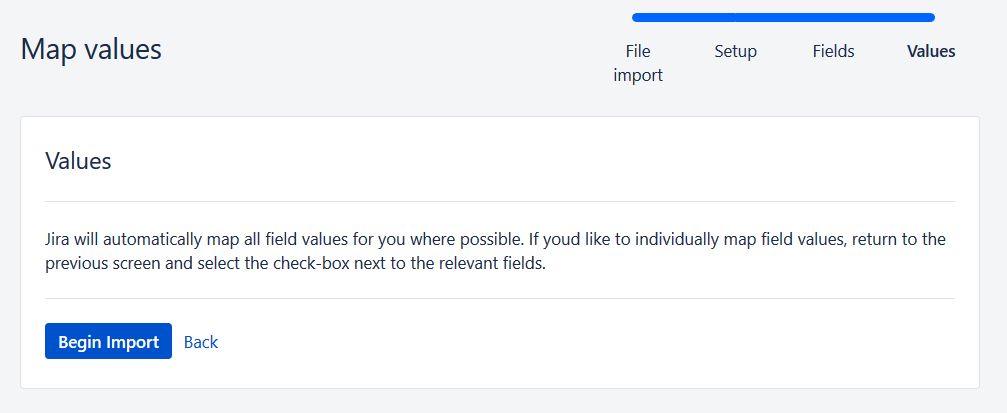


This is an important step. We need to map the fields between excel and Jira.  
Carefully review the first row of excel which is the heading row. Map the corresponding field in Jira be selecting the value from the dropdown as shown below.

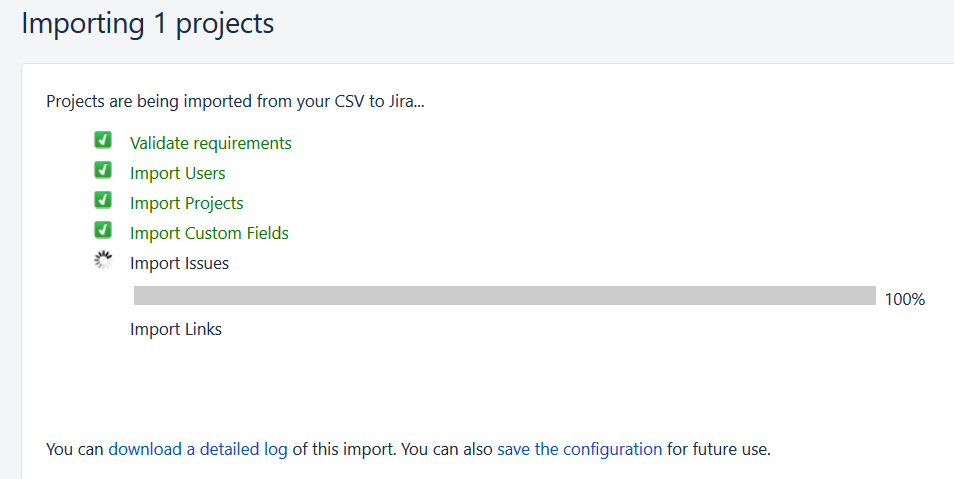


Once all the fields are mapped, click on Next.

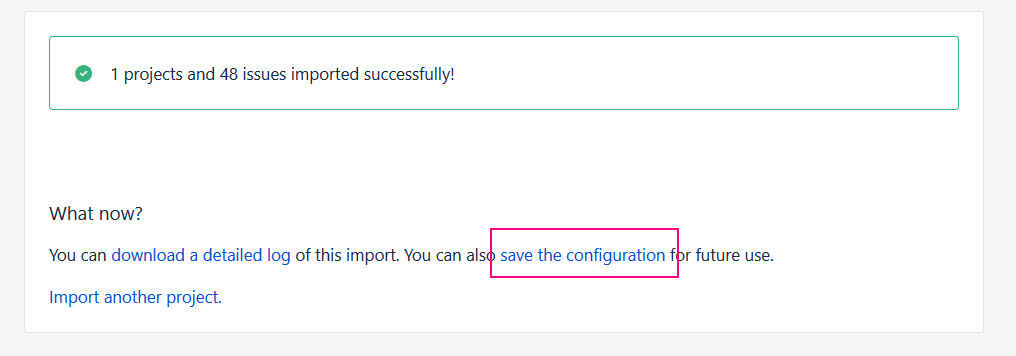
In the below screen, click on Begin Import as we do not wish to touch the field values and we can let Jira to map it automatically



We will get the below screen

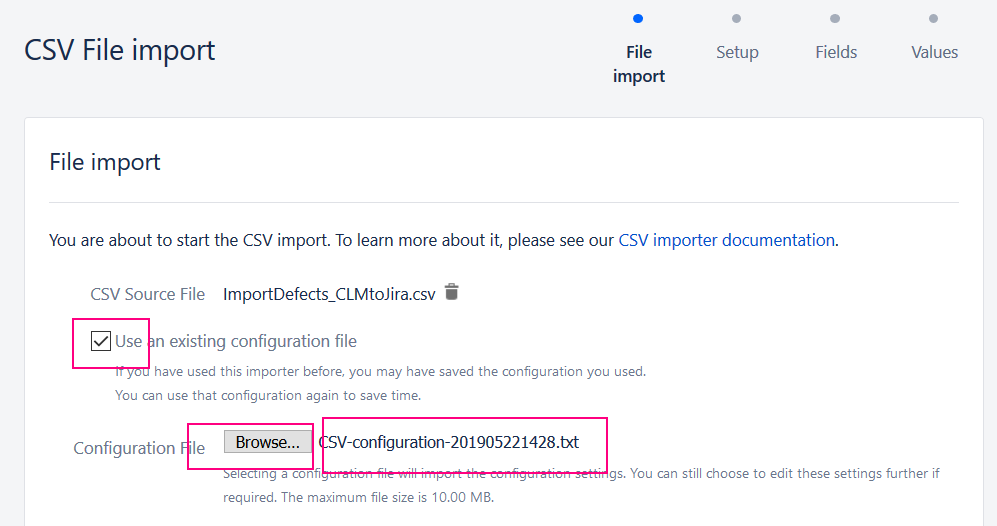


And finally this screen.



We can now download the config file so that next time if we are running the import, we do not have to do the field mapping again as all these are stored in this file.

During the next import, just point to the config file in the first screen and all the options in the following screens will automatically get filled up. See the below screenshot



We can now verify if the issues are imported successfully into the project by navigating into Issues menu and search for recently created issues

