**Test Parameters in Xray (Clear & Point-wise)**

**Test Parameters** in Xray are used to **run the same test with different data values** without creating multiple test cases.

**What are Test Parameters?**

* Variables used inside a **Test**
* Values are assigned during **Test Execution**
* Mainly used for **data-driven testing**

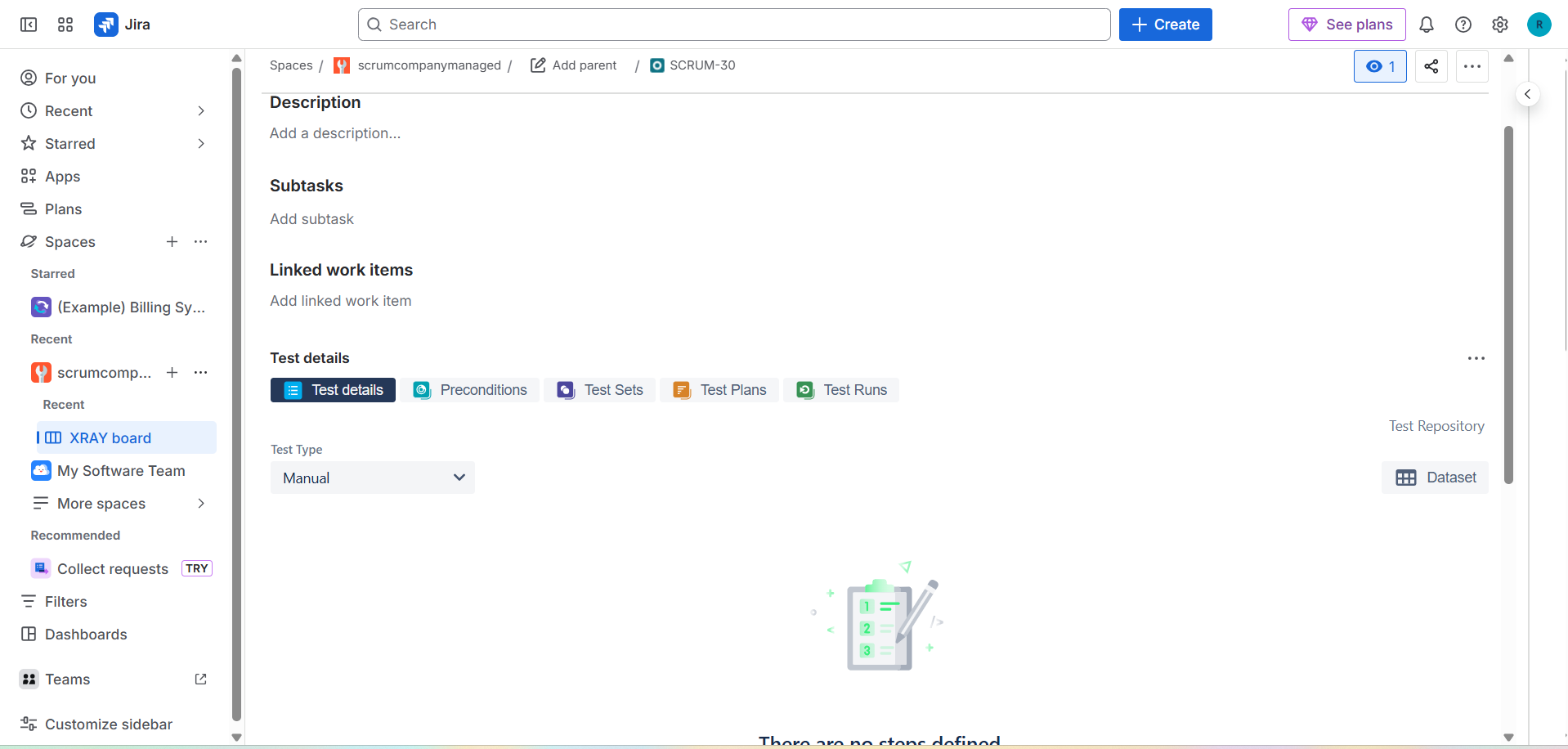
Example:

* Parameter: username
* Values: admin, user1, user2

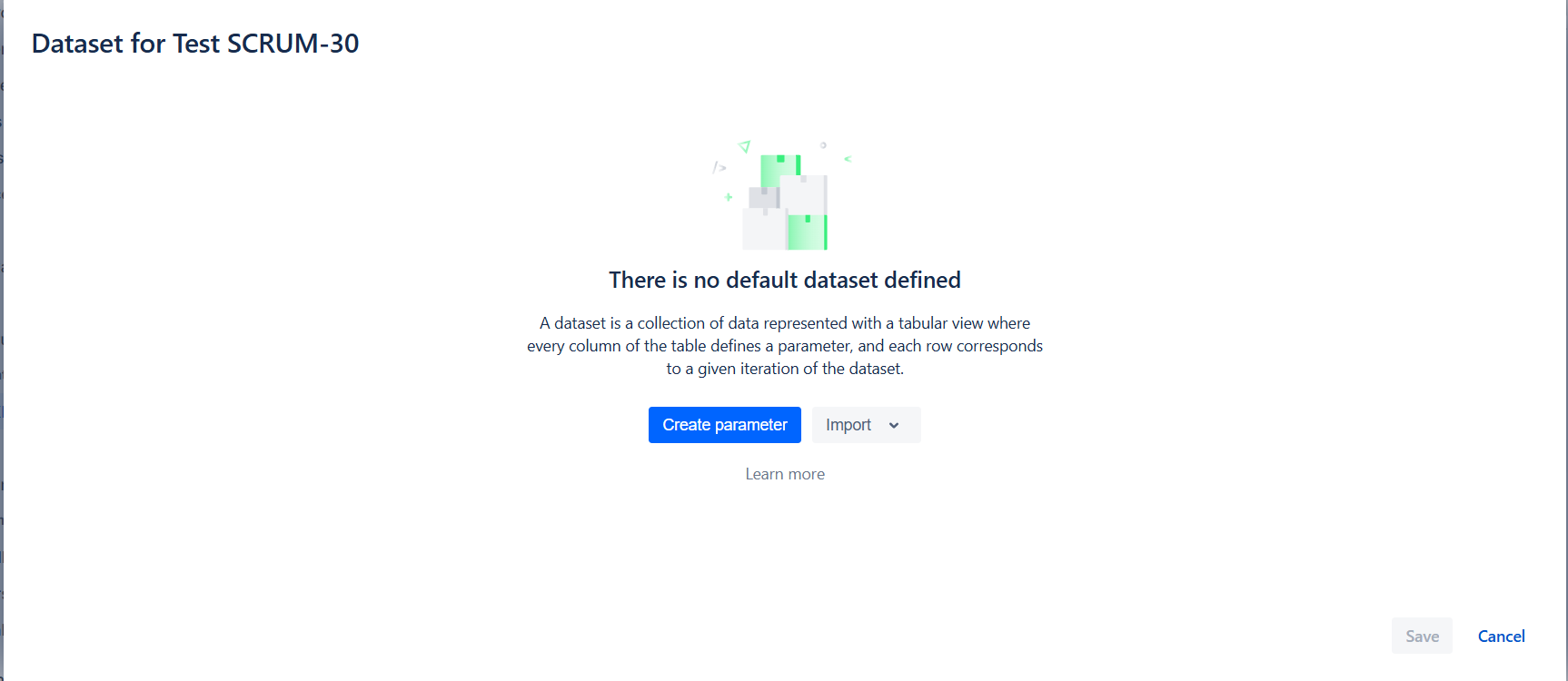
**Where Test Parameters Are Used**

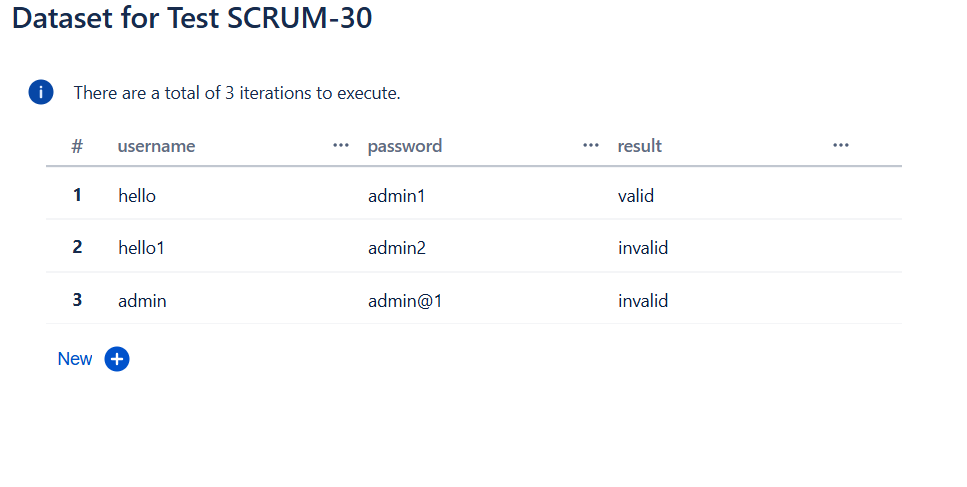
* **Manual Tests**
* **Cucumber Tests**
* **Automated Tests** (JUnit, TestNG, etc.)

Click on the dataset

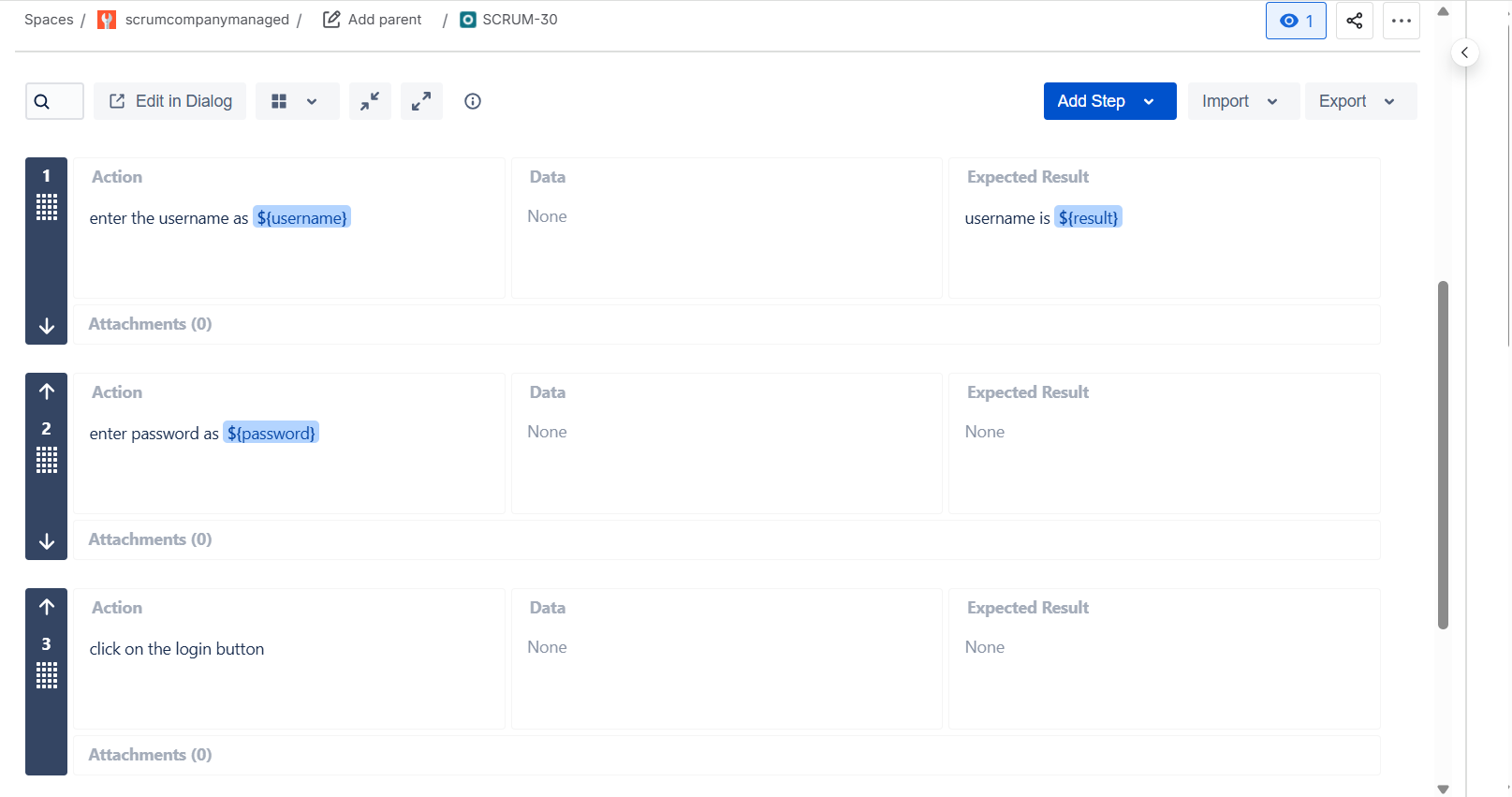


Click on create parameter

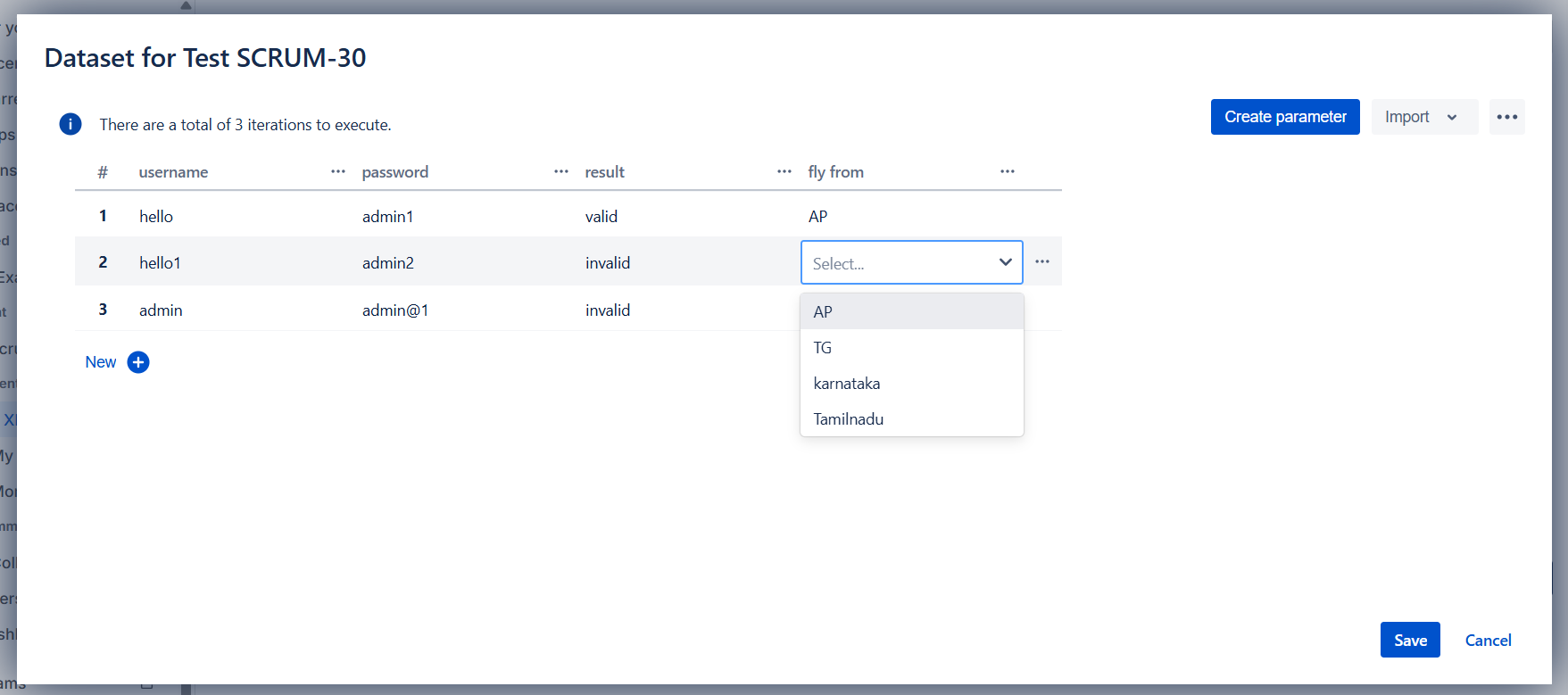




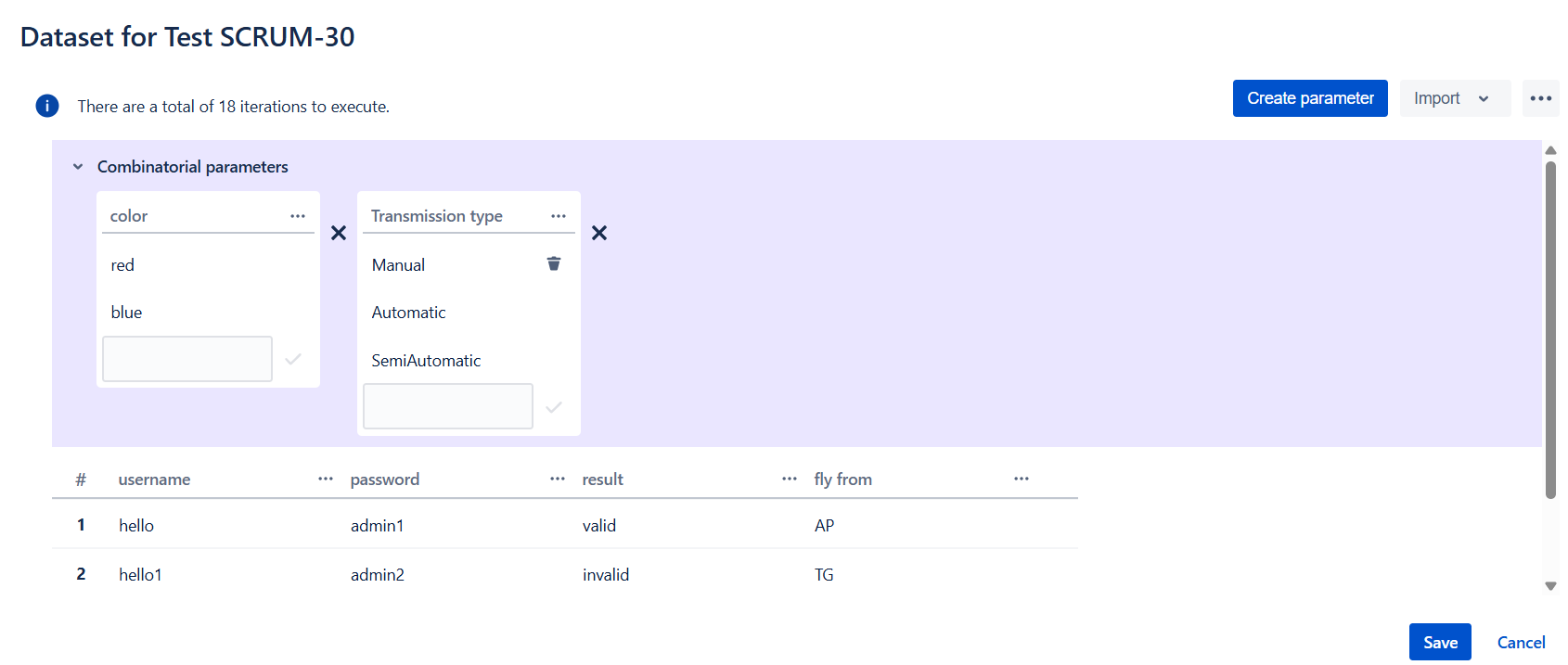
Adding tests



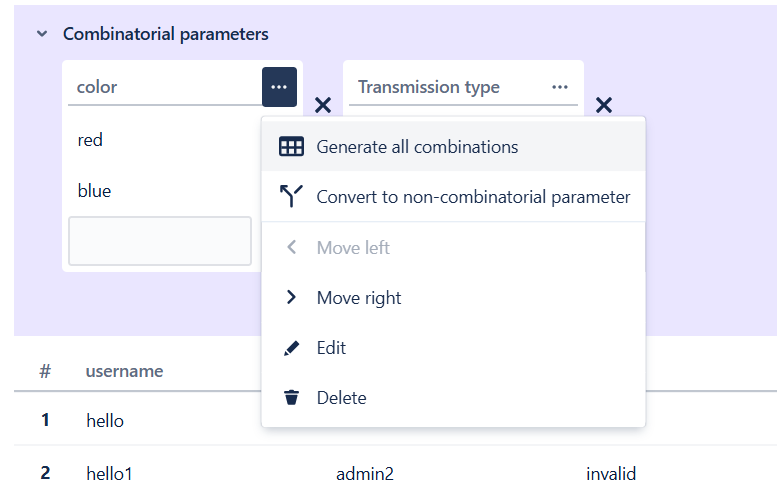
Adding list in the dataset

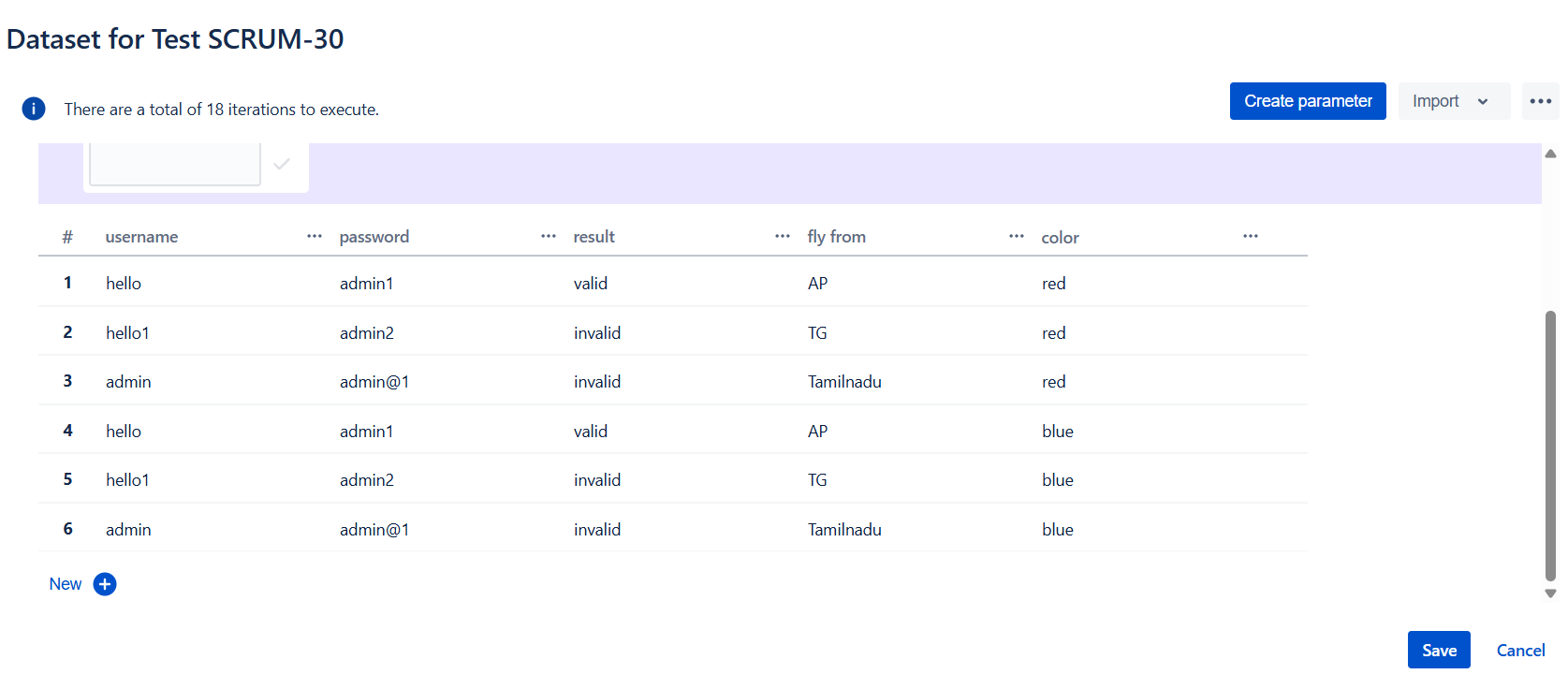


Creating combinatorial parameters



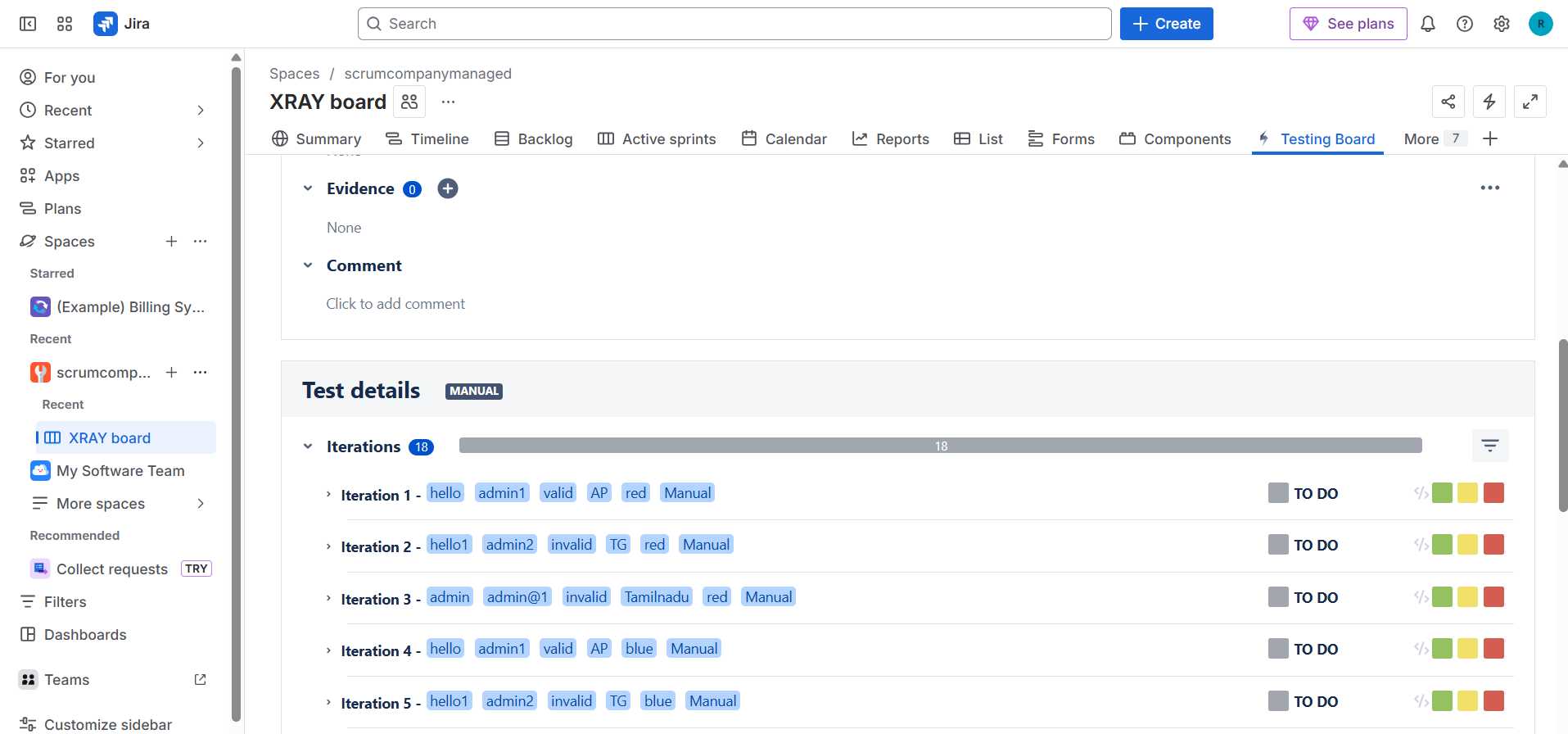
Generating combinations to all the parameters from combinatorial parameters

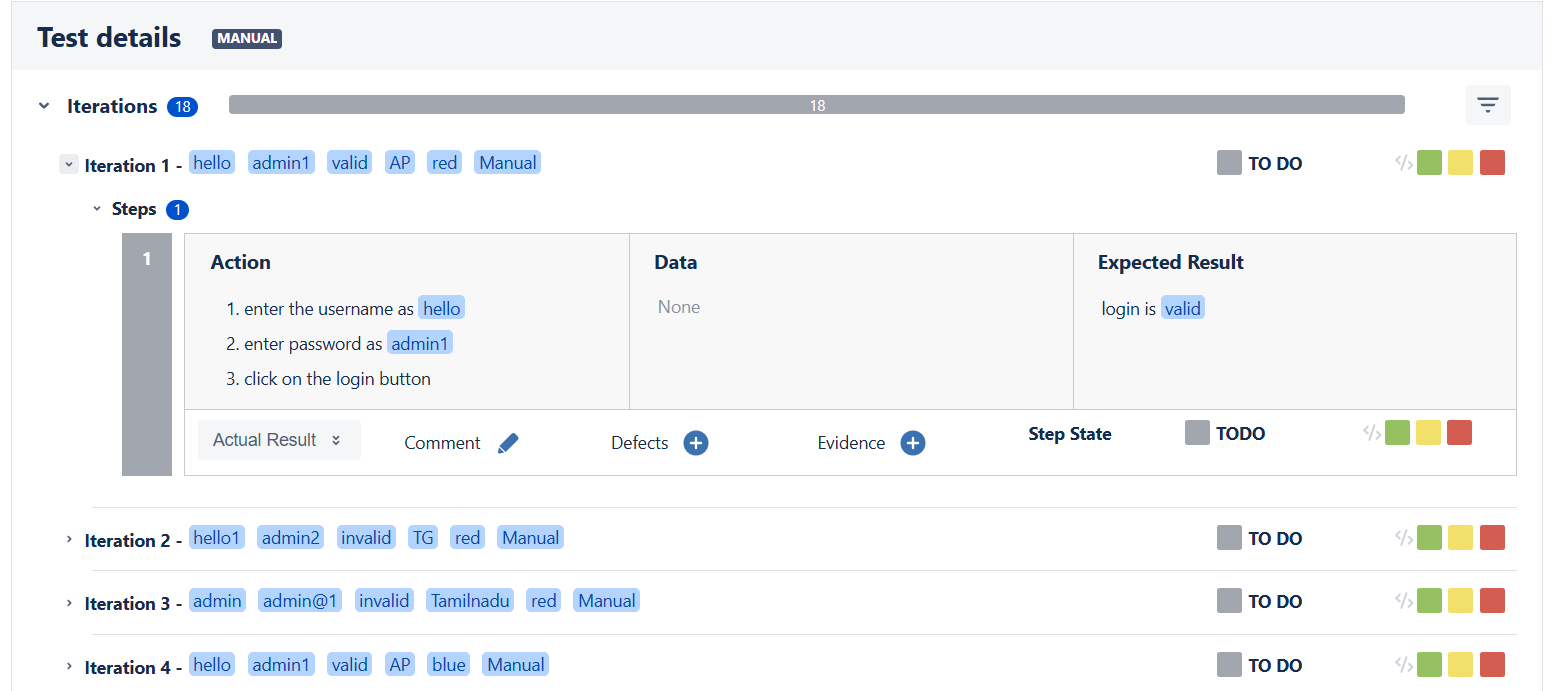




**Executing the parameterized test**

For executing the parameterized tests we have to create the test execution





**Test repository in xray**

**What is the Test Repository?**

The **Test Repository** in Xray is a **folder-based structure** used to **organize and manage test cases** within a Jira project.

Think of it as:

📁 **A file system for your test cases**

**Key Purpose of Test Repository**

* Organize tests **logically**
* Improve **test case management**
* Easy navigation for large projects
* Reuse tests across plans and executions

**Where You Can Find It**

* Go to **Jira Project**
* Open **Xray**
* Click **Test Repository**

**Structure of Test Repository**

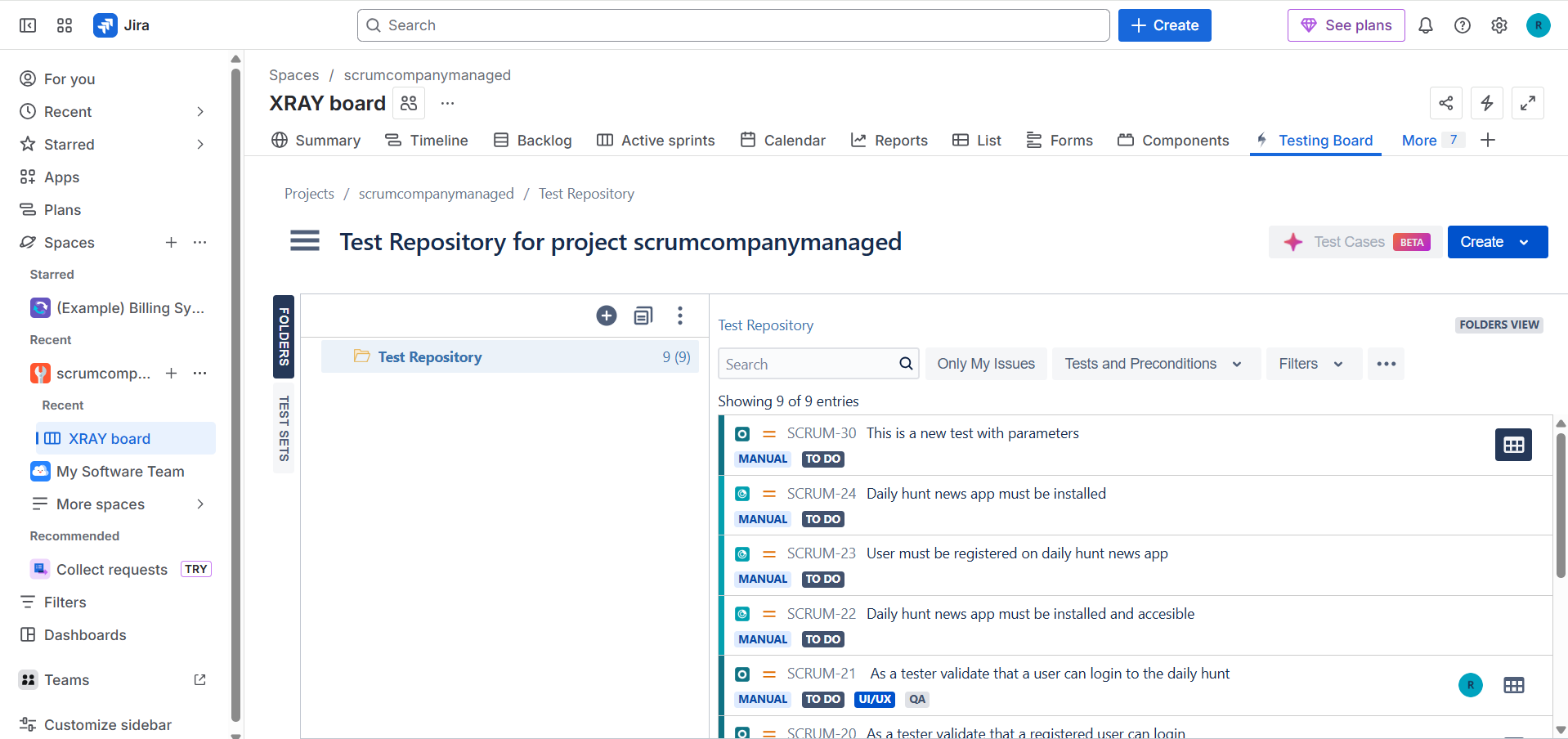
* **Root folder**
* **Subfolders**
  + Module-wise (Login, Payments, Sports)
  + Feature-wise
  + Sprint-wise
  + Release-wise

**What You Can Do in Test Repository**

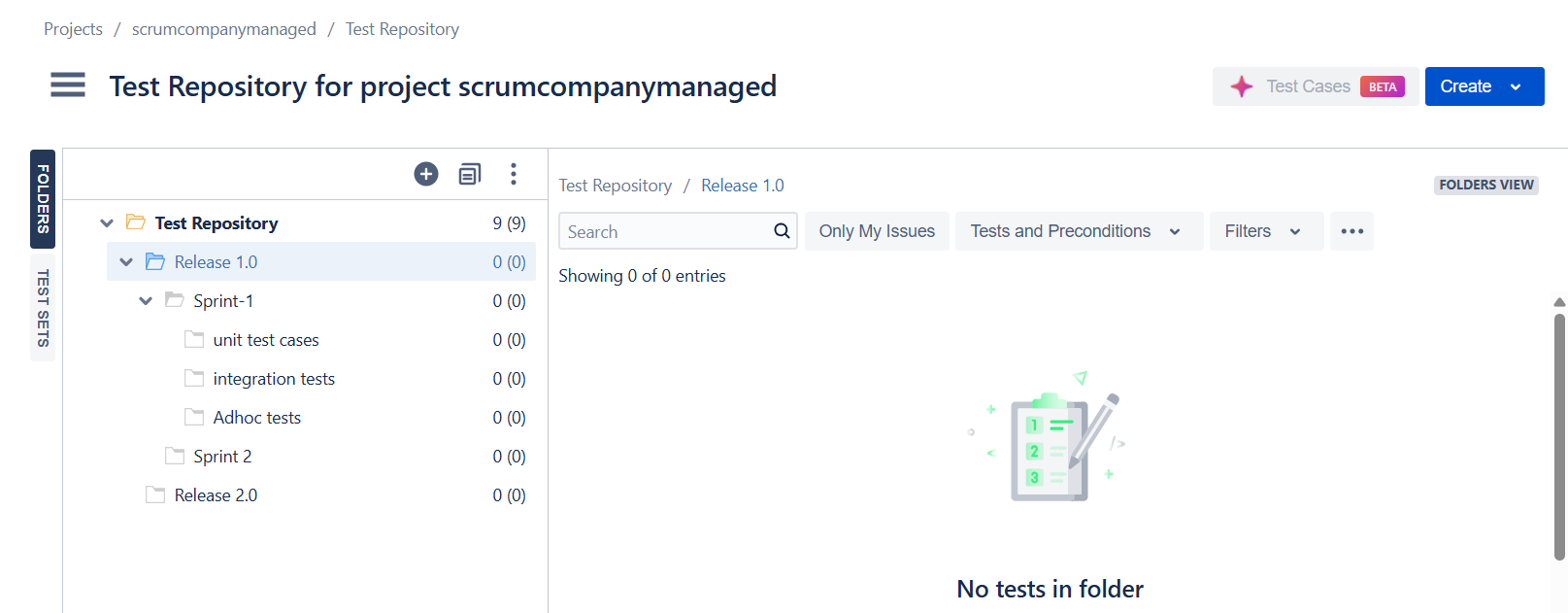
* Create folders and subfolders
* Create new **Tests** inside folders
* Move existing tests between folders
* Bulk manage tests

📌 Note:

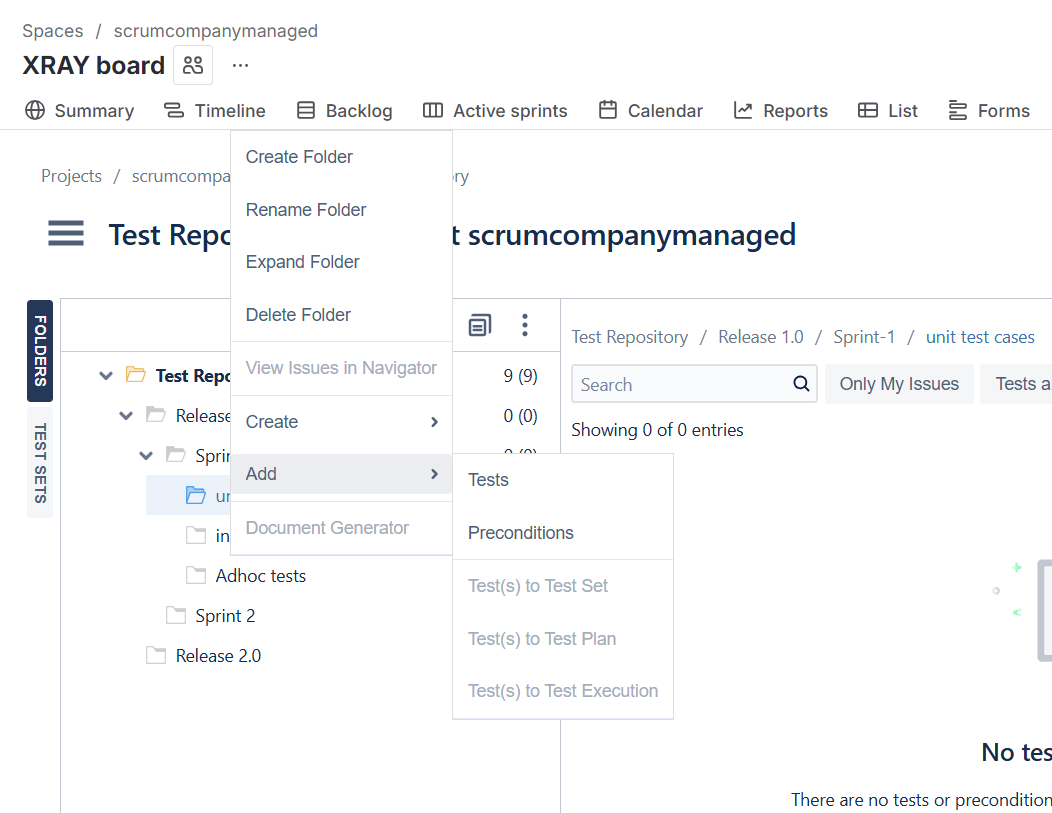
* Repository **does not execute tests**
* It is only for **organization**

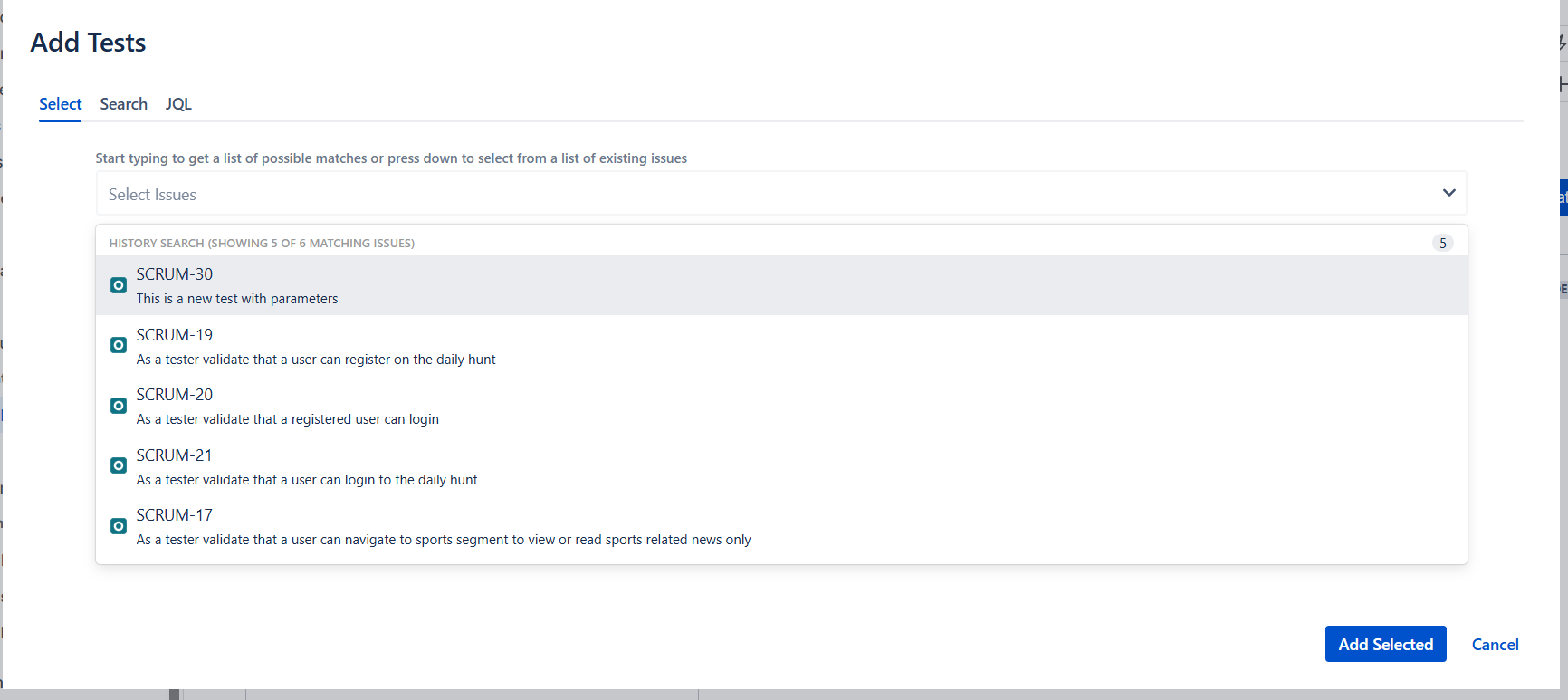


**Creating a repository**

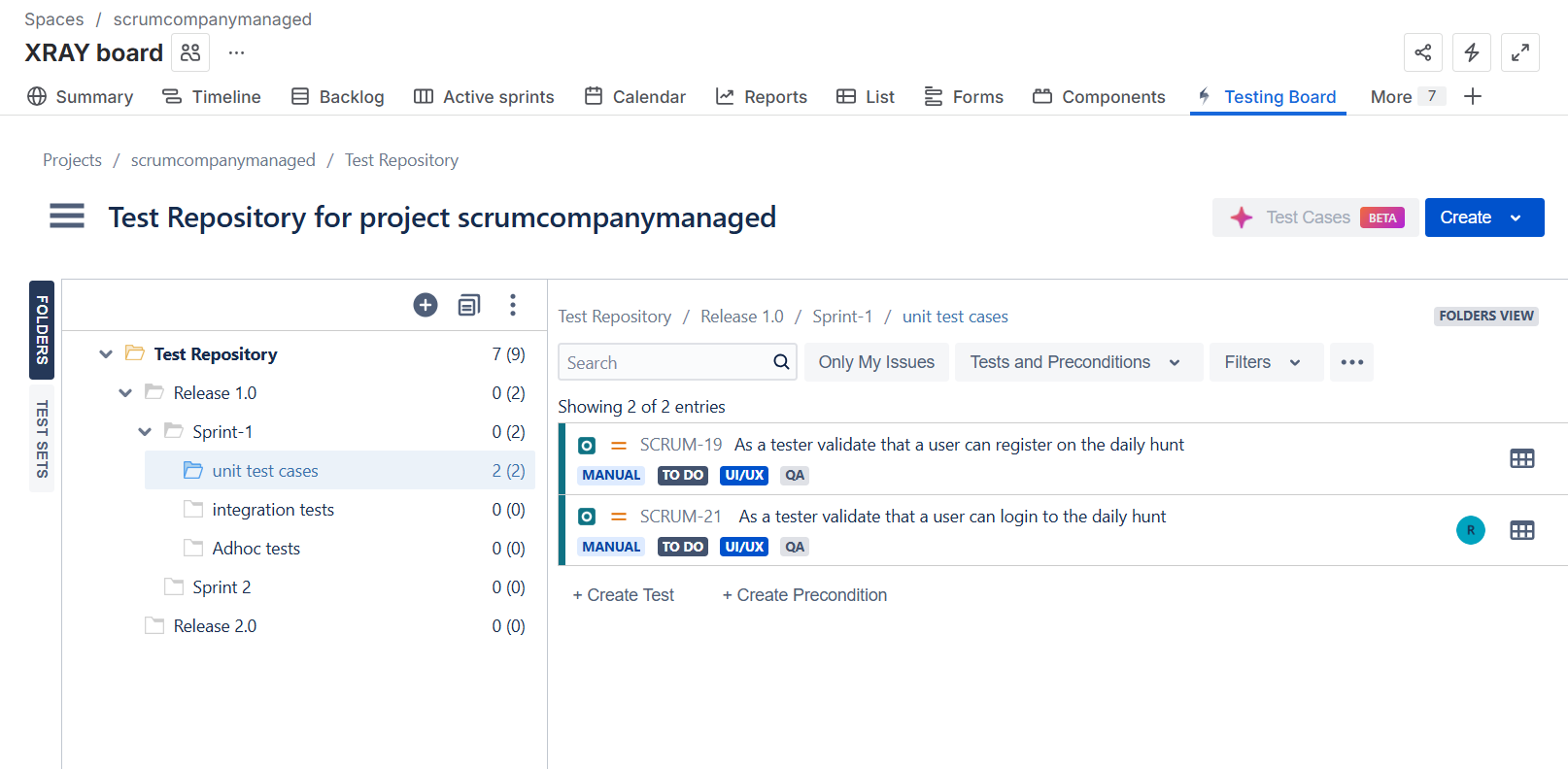
****

**Adding tests to the repository**

****

****

**Tests added from test repository to the unit tests repository**

****

**In the above we can also see the how many no of tests are present in respective folder!!!**

**Test coverage in xray**

**What is Test Coverage?**

**Test Coverage** in Xray shows **which requirements (work items)** are covered by **which tests**, and whether those tests have been **executed and passed**.

In short:

**It answers: “Is this requirement tested?”**

**Why Test Coverage Is Important**

* Ensures all requirements are tested
* Avoids missing test cases
* Improves quality and traceability
* Helps in release decisions

**How Test Coverage Works in Xray**

**1. Link Tests to Work Items**

* Link **Test** → **Story / Task / Bug**
* This creates **requirement coverage**

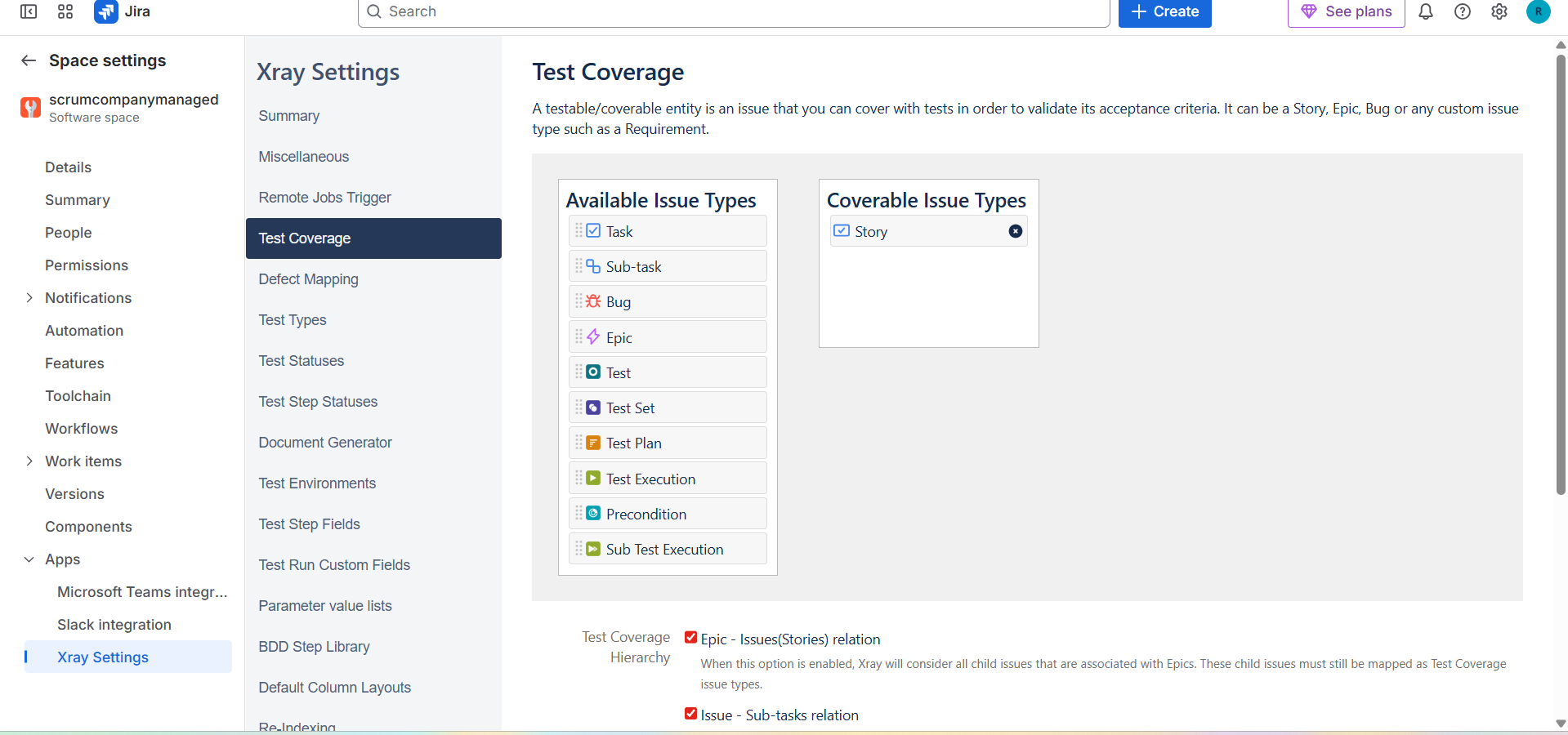
**2. Execute the Tests**

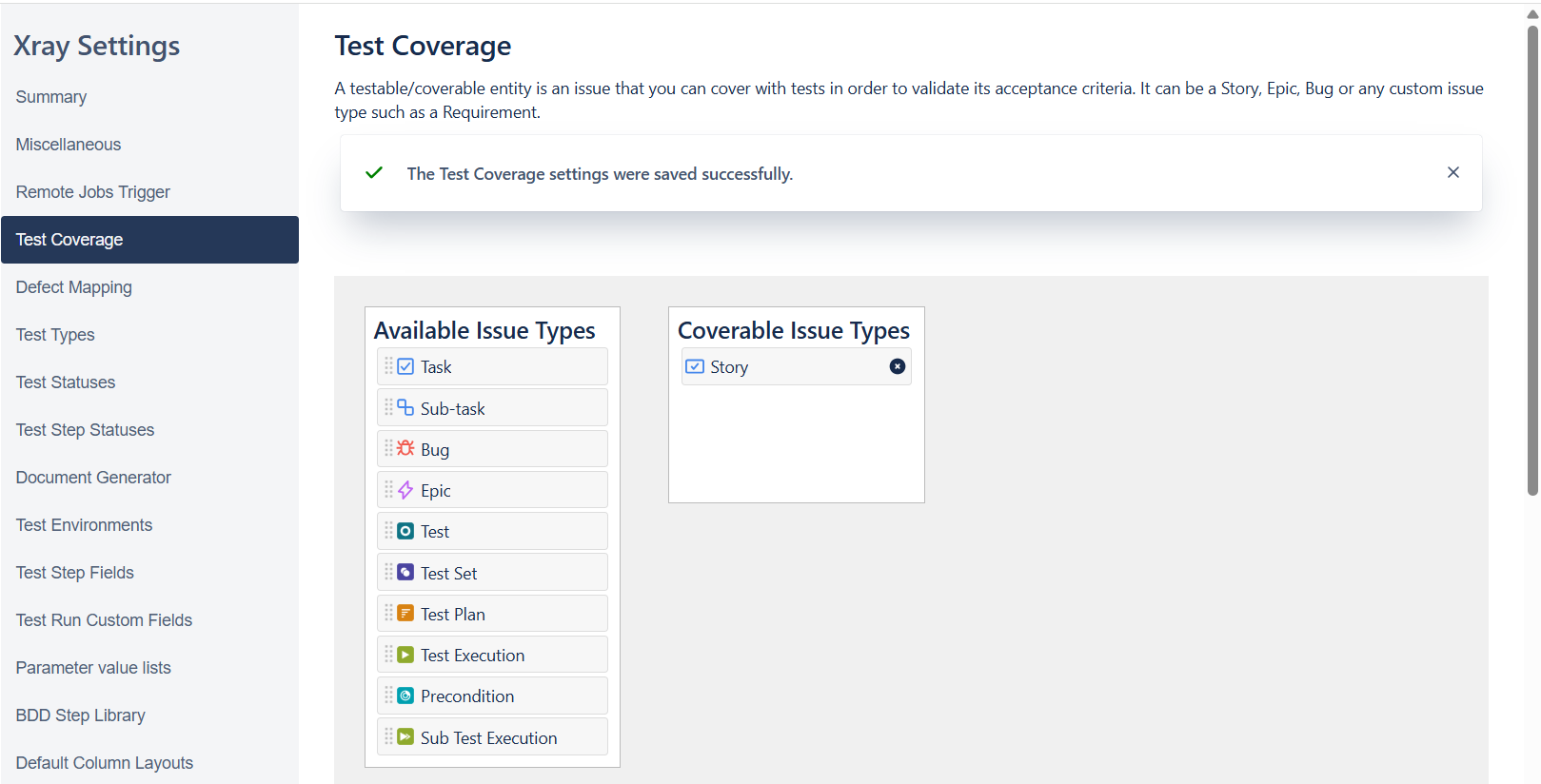
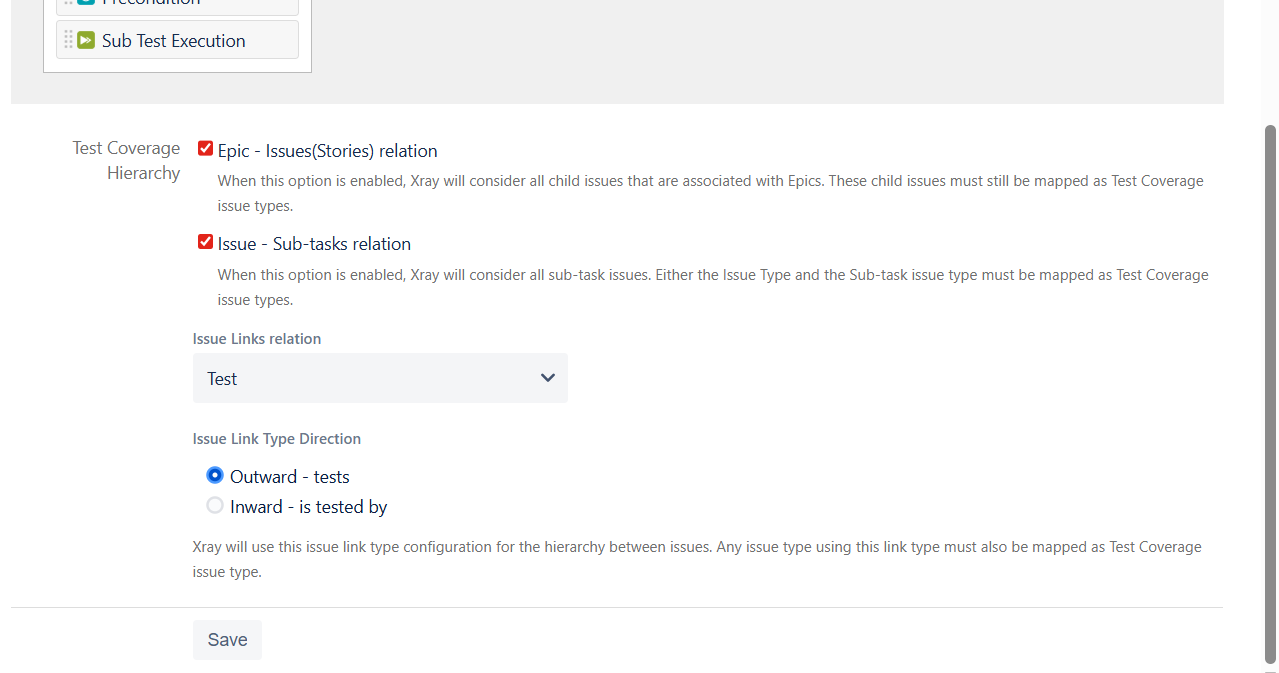
* Add tests to a **Test Execution**
* Run them (Pass / Fail / Abort)

**3. View Coverage Status**

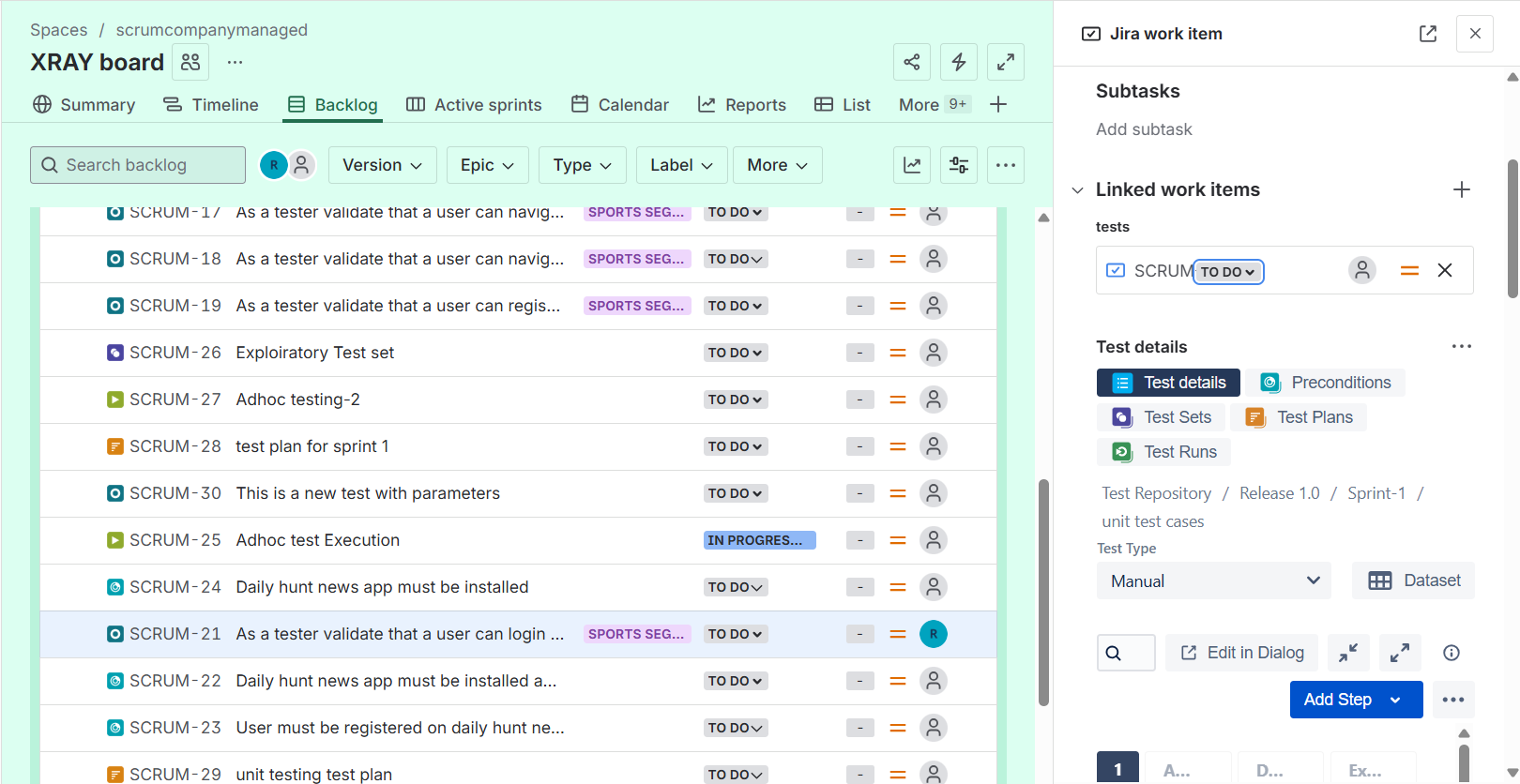
* Xray calculates coverage based on:
  + Linked tests
  + Latest execution status

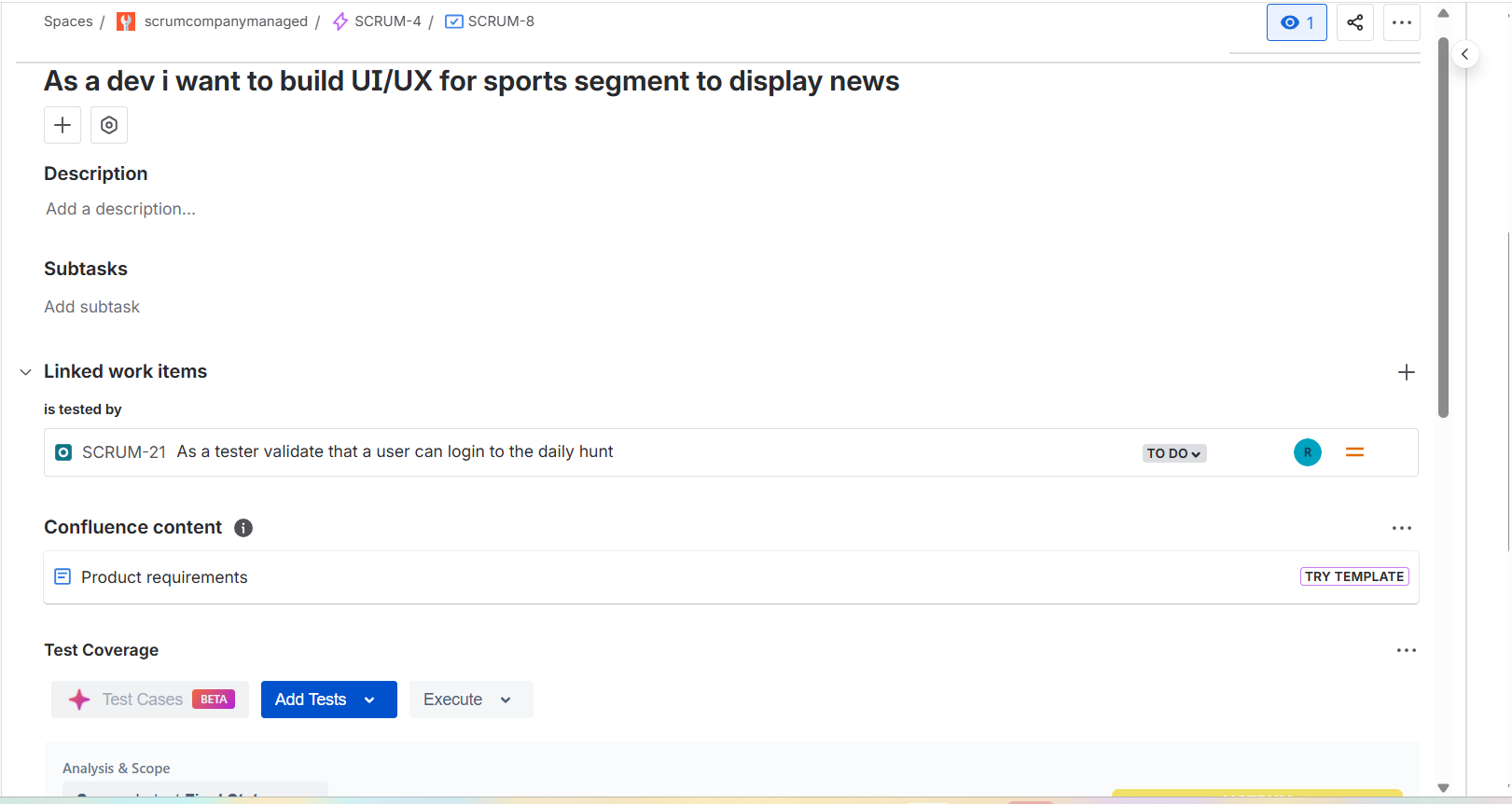
**Enabling the test coverage through space settings->apps->xray->test coverage**

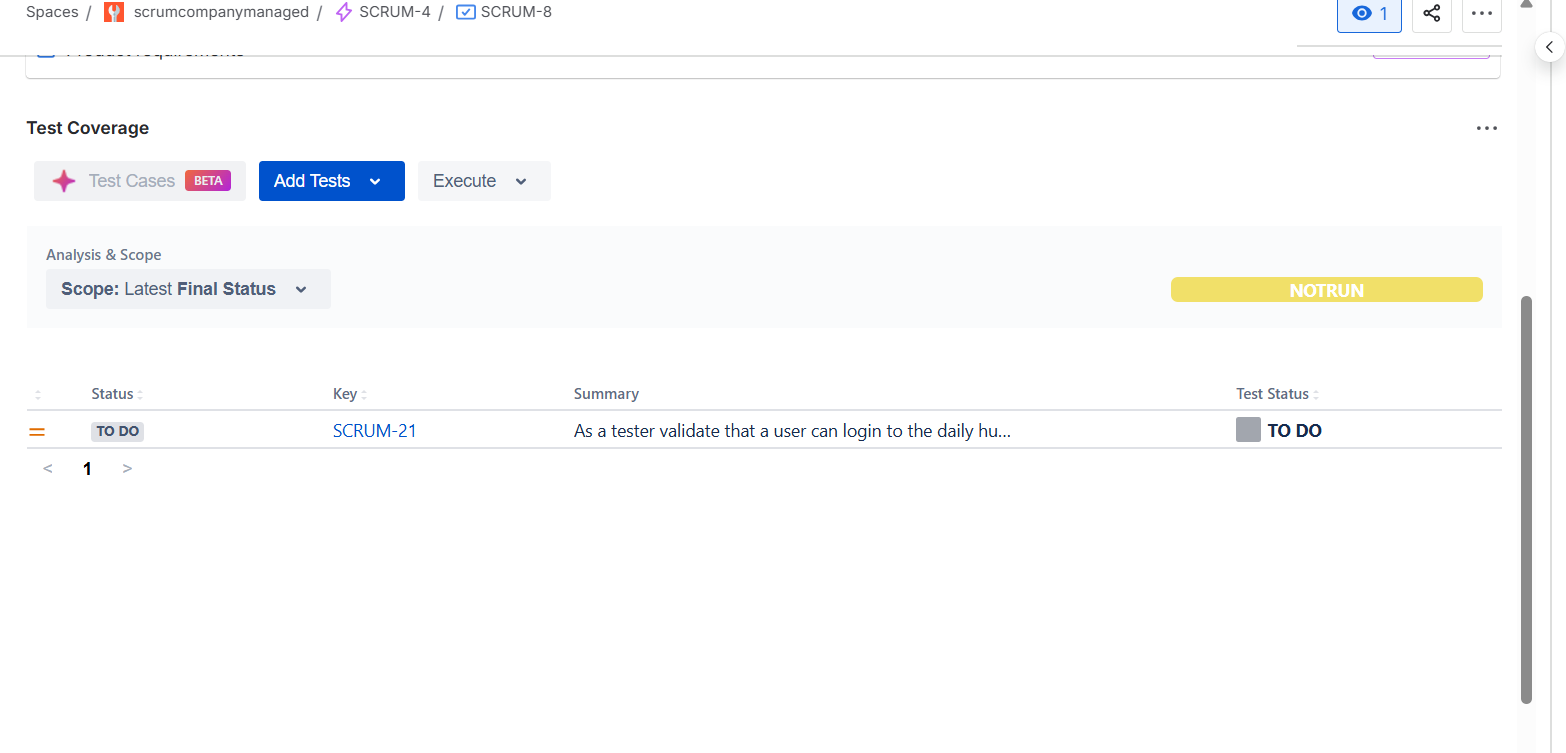
****

****

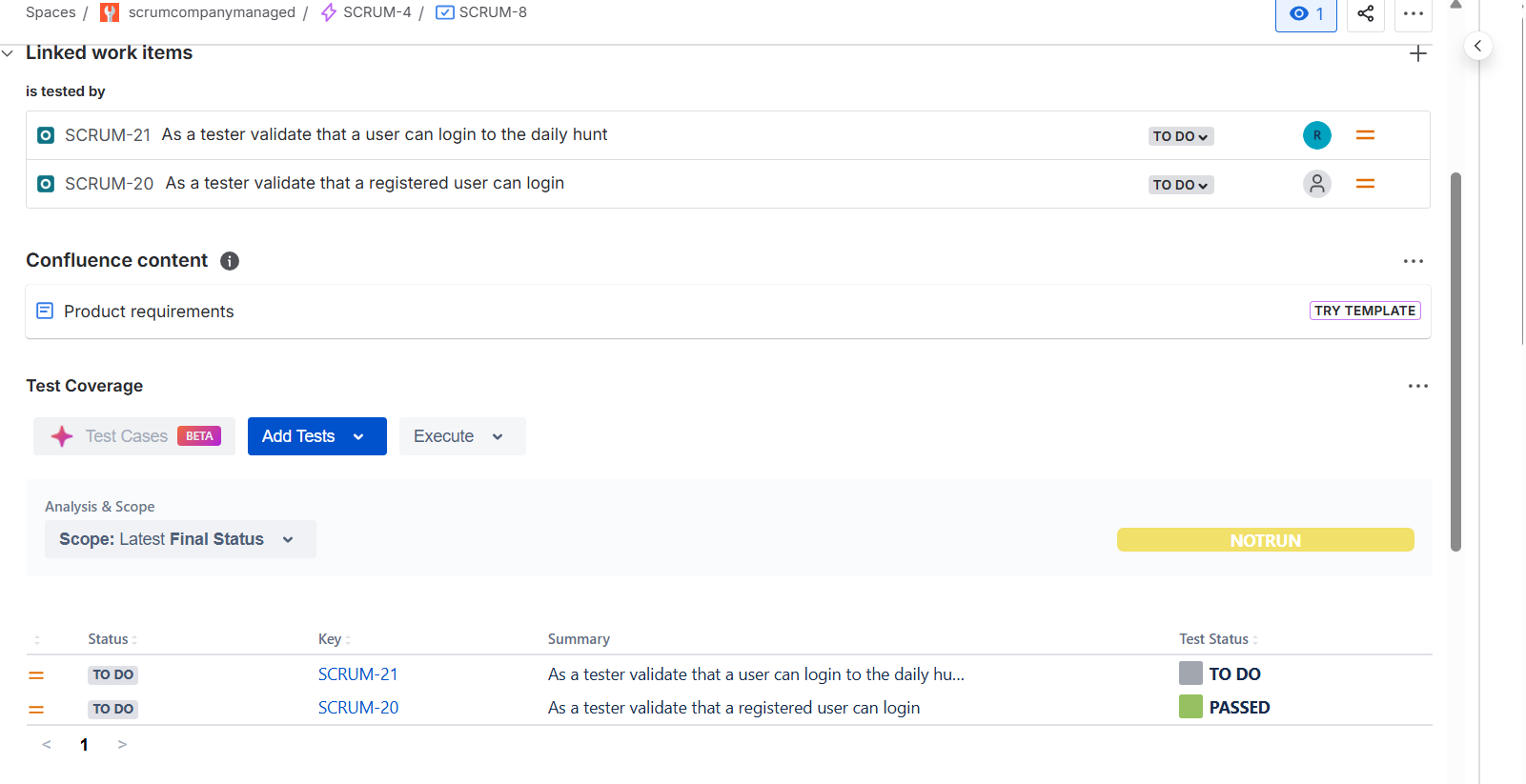
**Link a story to a test**

****

****

****

**Linked other work item**

****

**Defect Mapping**

**What is Defect Mapping?**

**Defect mapping** in Xray is the process of **linking defects (bugs)** to **failed tests or test executions** to track issues found during testing.

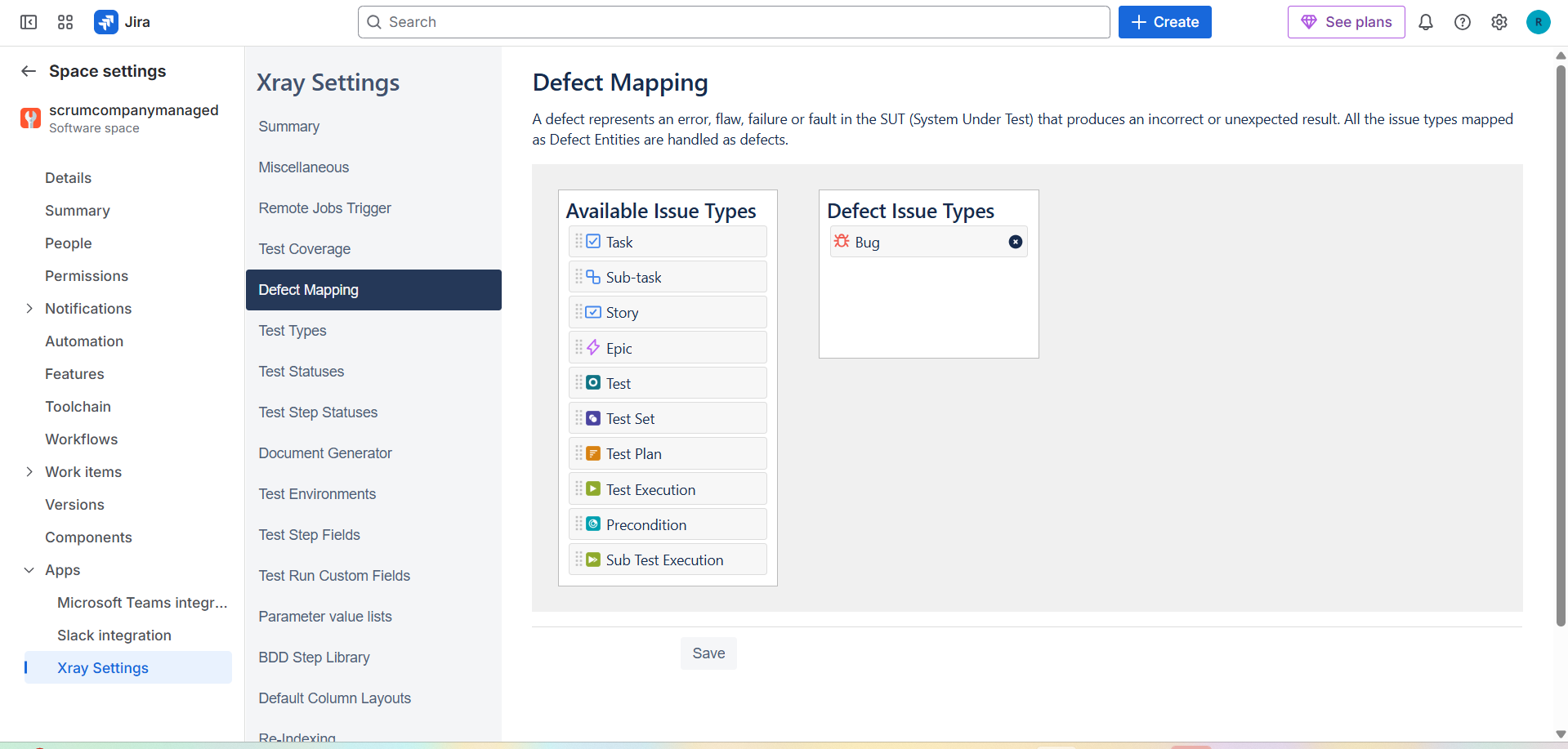
In short:

**It connects test failures to bugs in Jira.**

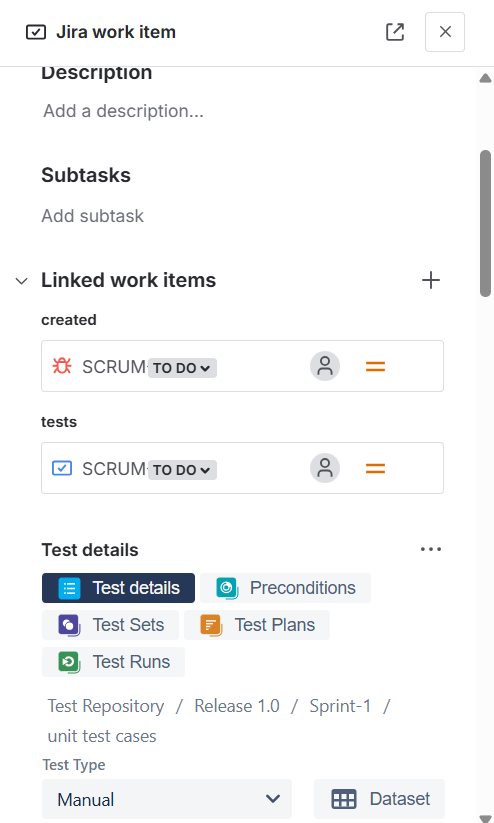
**Why Defect Mapping Is Important**

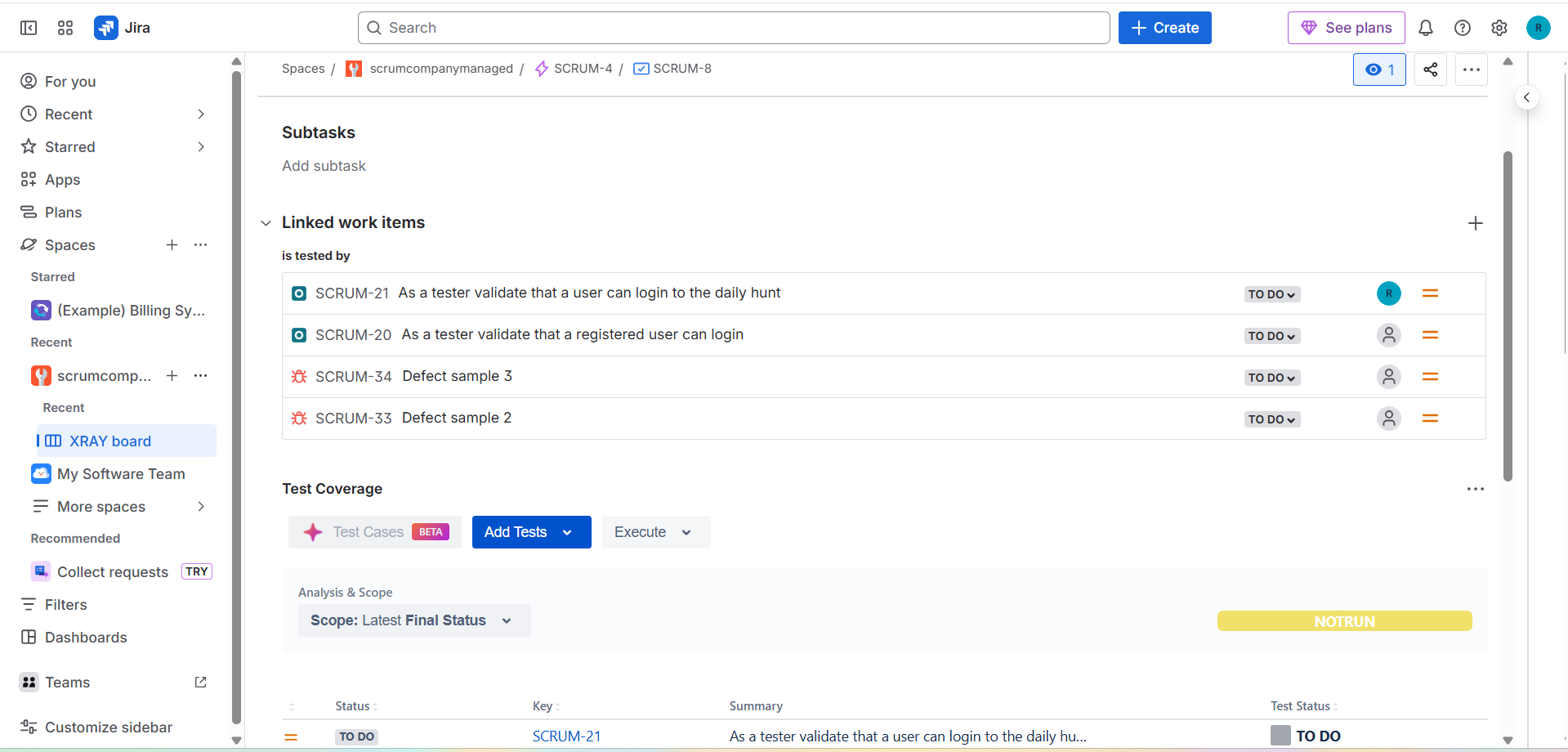
* Tracks which tests found which defects
* Improves traceability
* Helps analyze defect impact
* Provides better quality reports

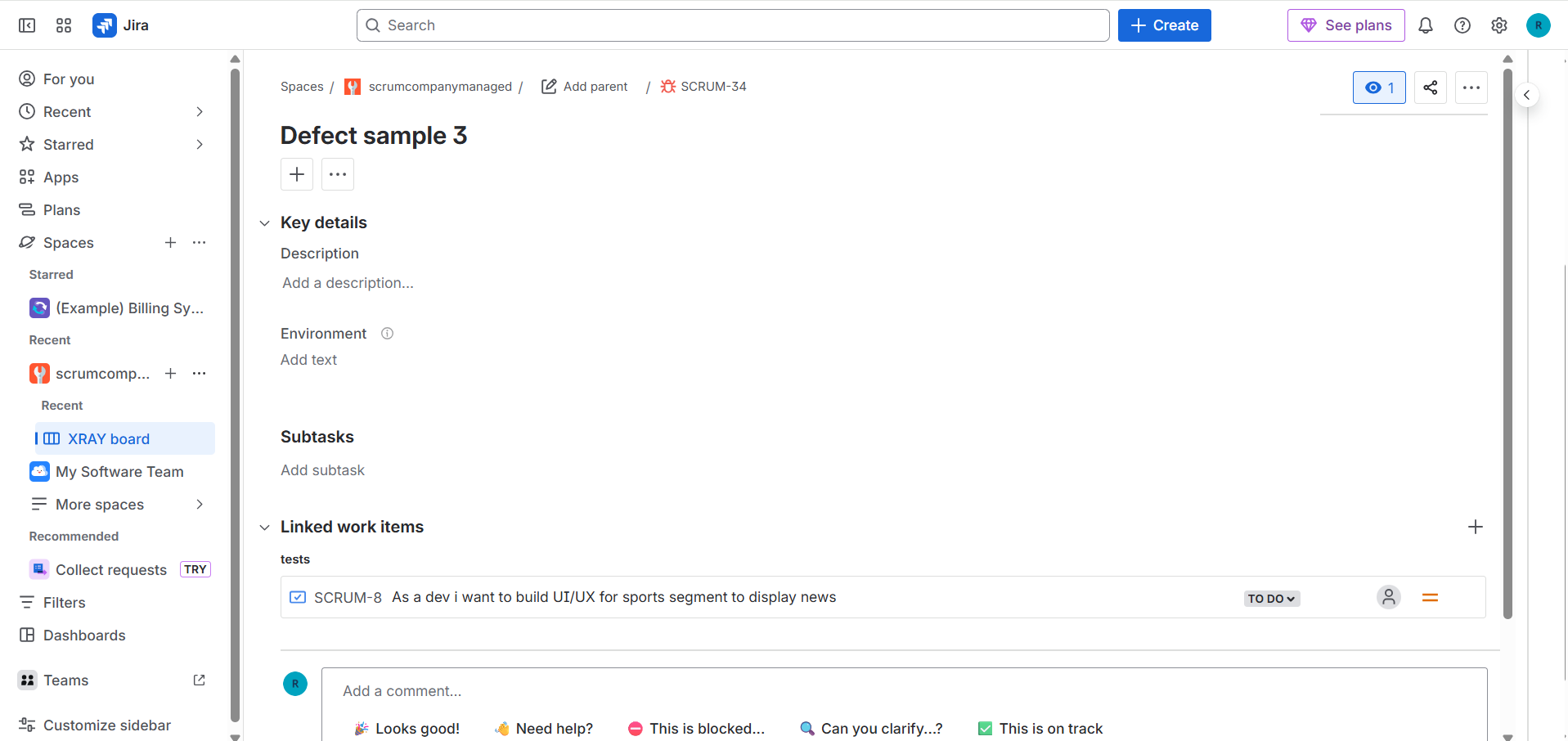
Inorder to do the defect mapping we have to enable some settings same as test coverage

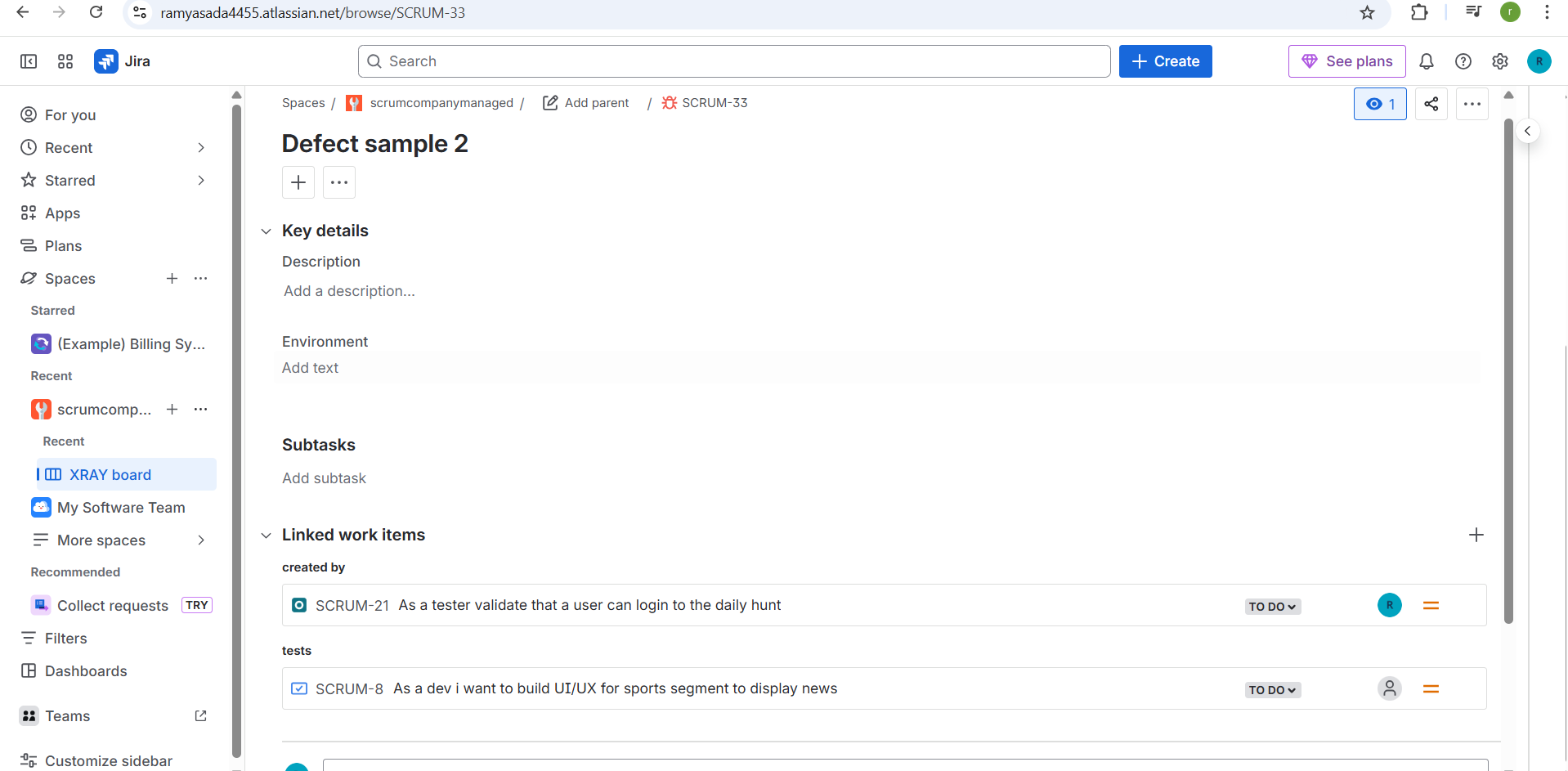


Linked defect to the work item









**Test Environment**

**What is a Test Environment?**

A **Test Environment** in Xray represents the **system setup** under which a test is executed, such as OS, browser, device, or deployment setup.

In simple words:

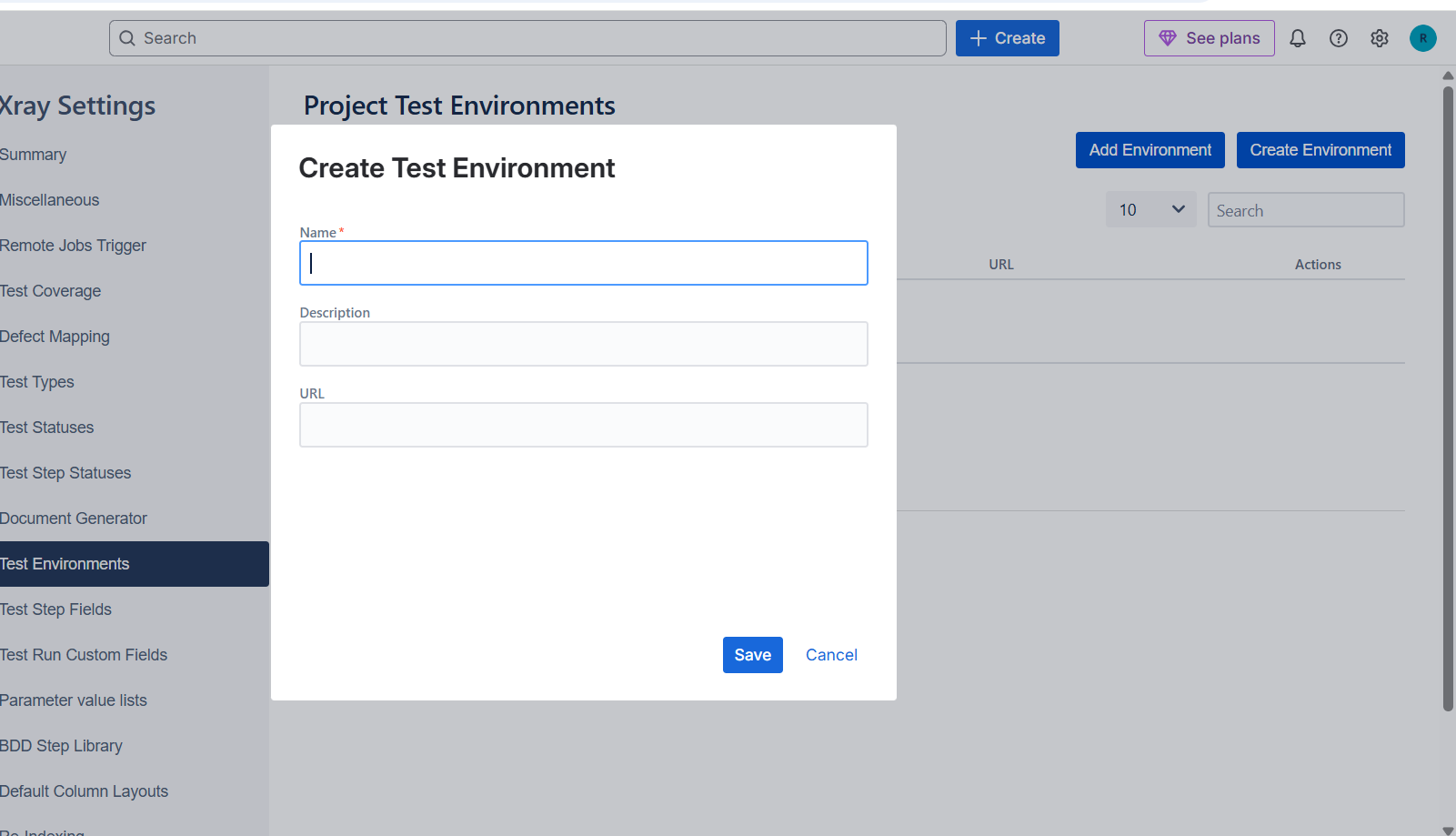
**It tells *where* and *under what conditions* the test was executed.**

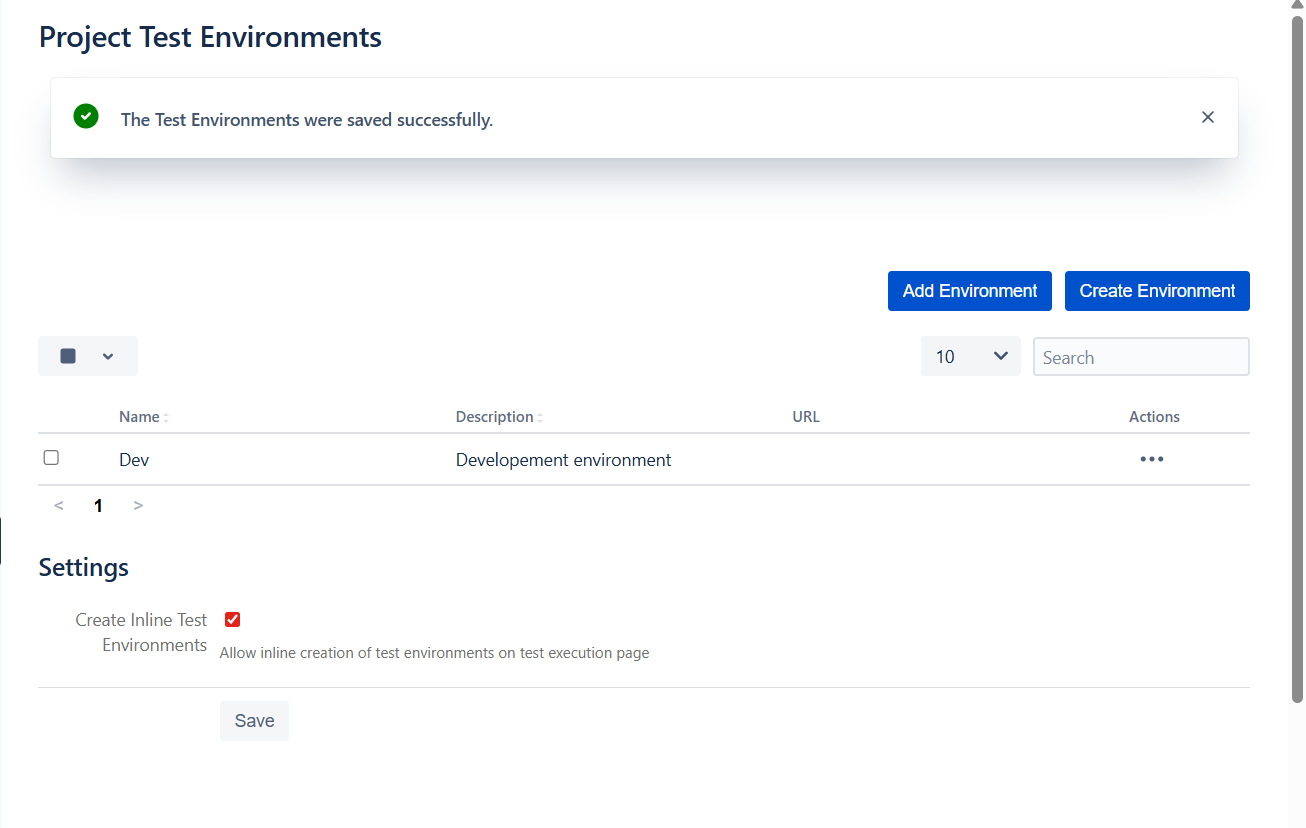
**Why Test Environments Are Important**

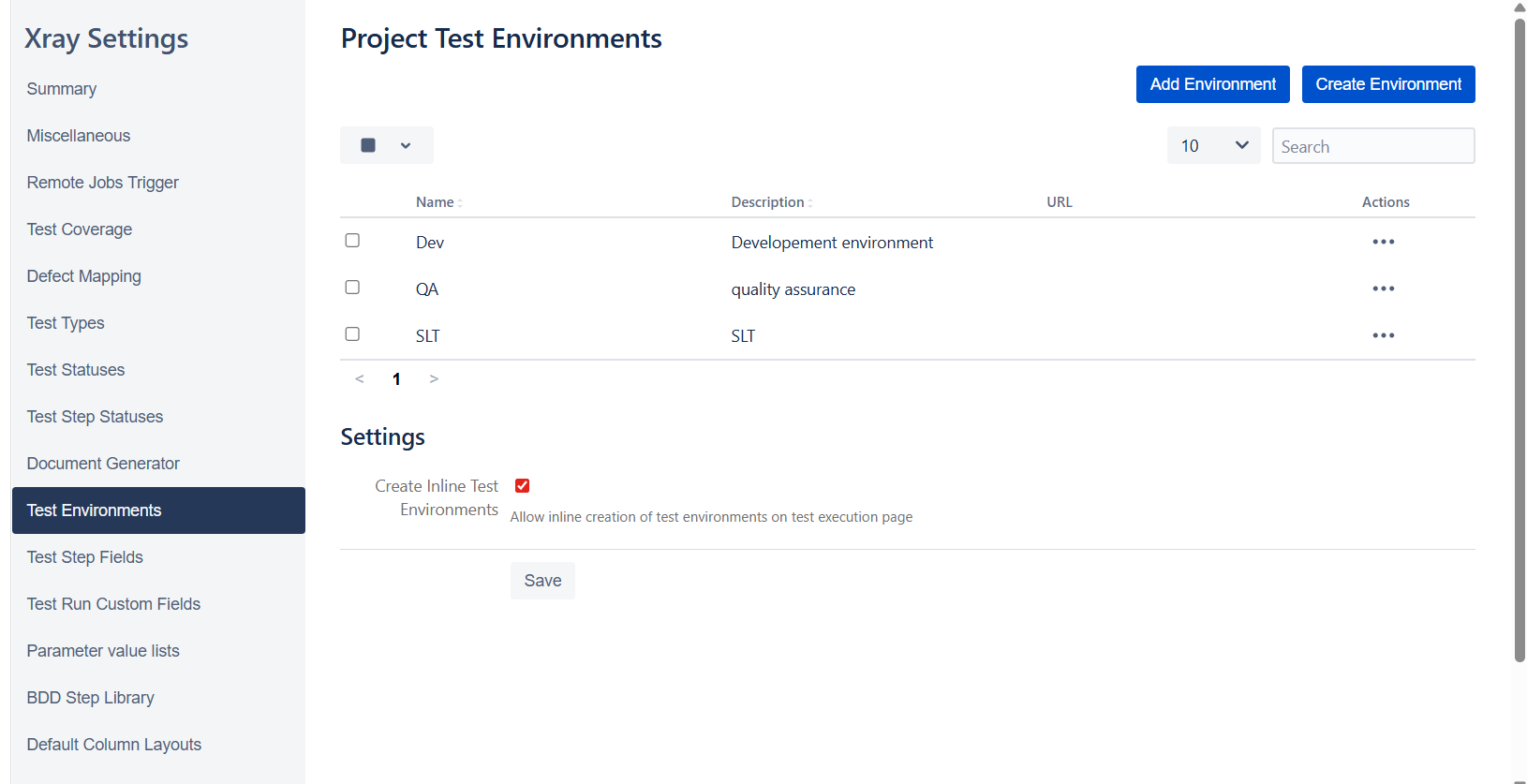
* Same test behaves differently in different environments
* Helps identify environment-specific issues
* Improves defect analysis and reporting
* Essential for cross-browser / cross-platform testing

Same as defect mapping we have enable some settings for test environment

Create environment

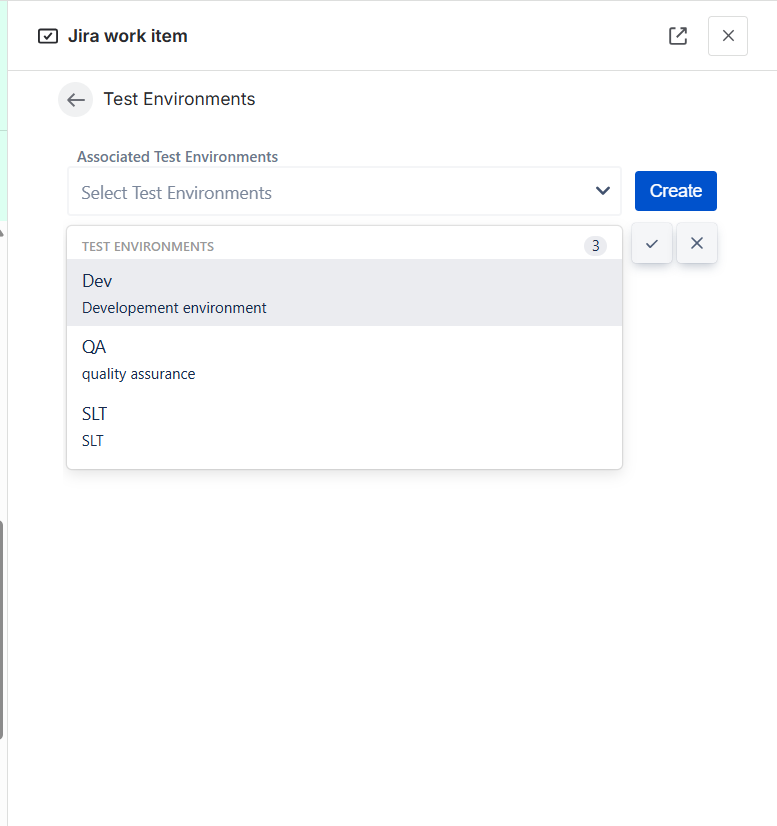


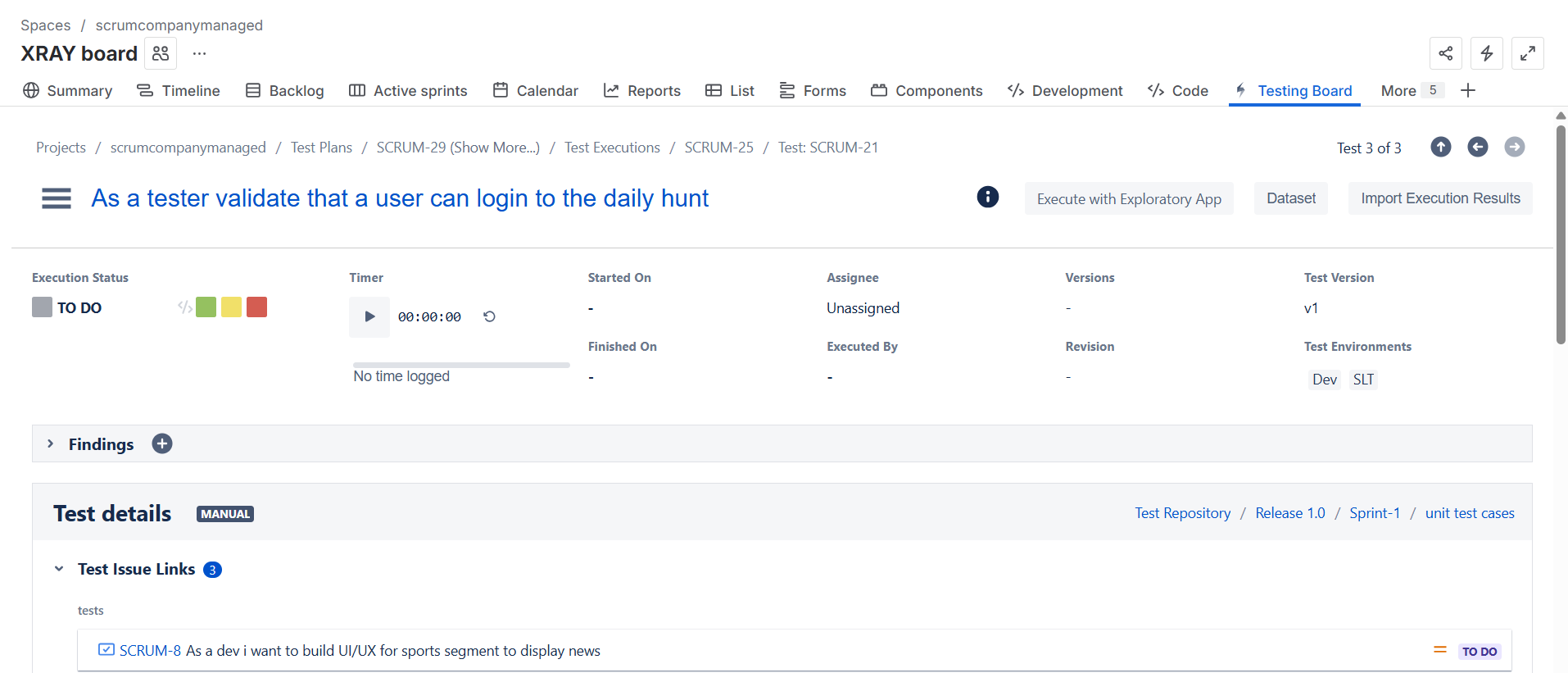




If we see the test execution we can find the test environment

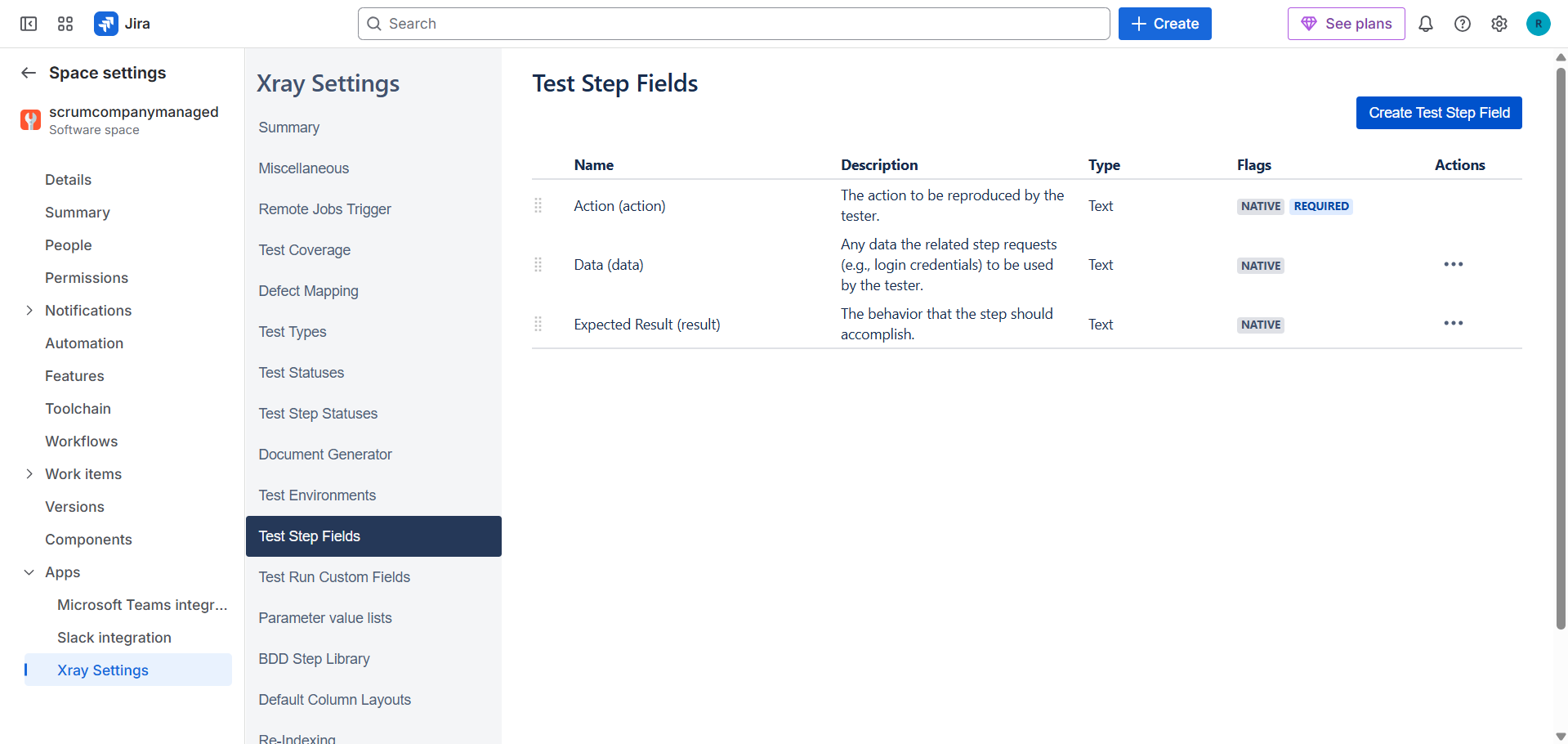


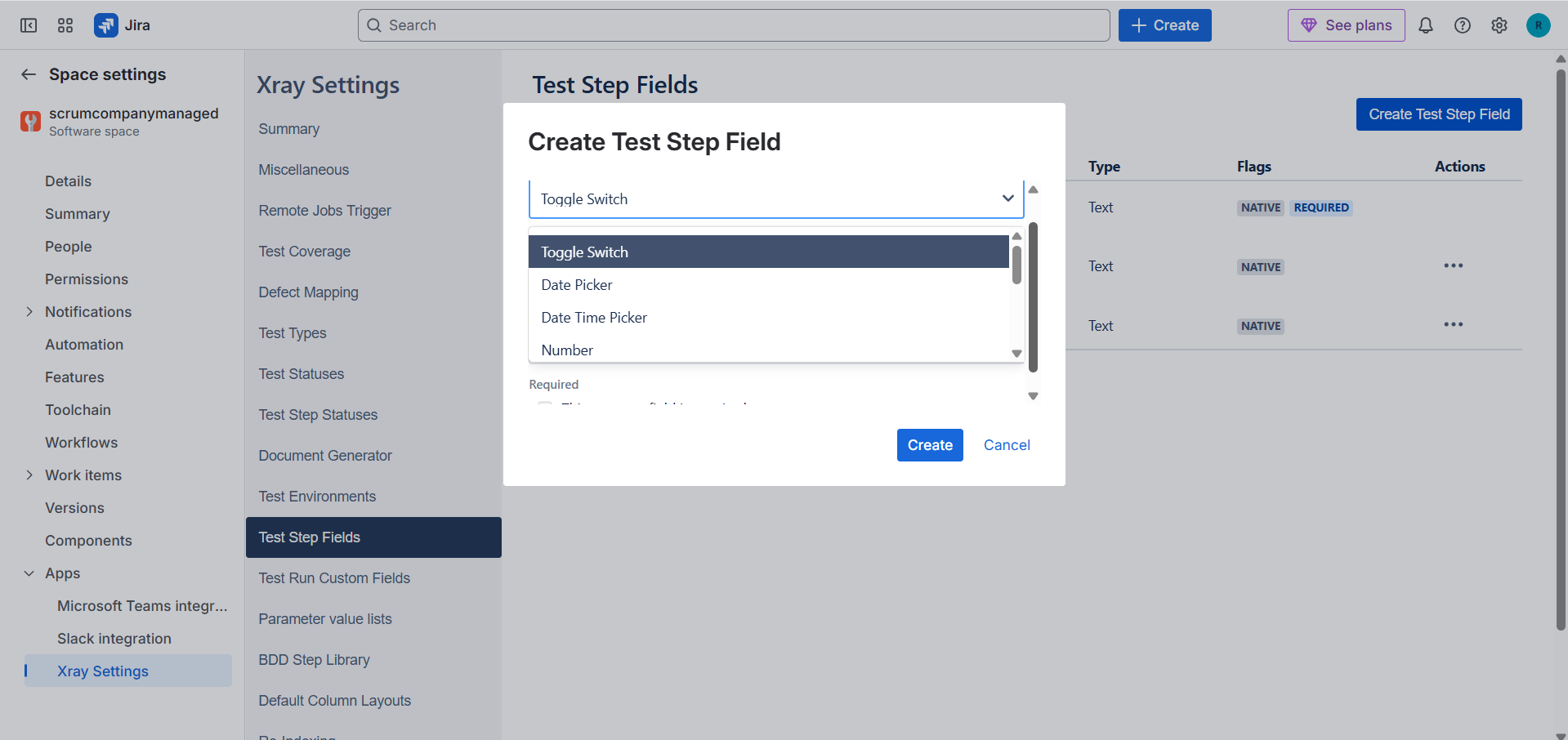


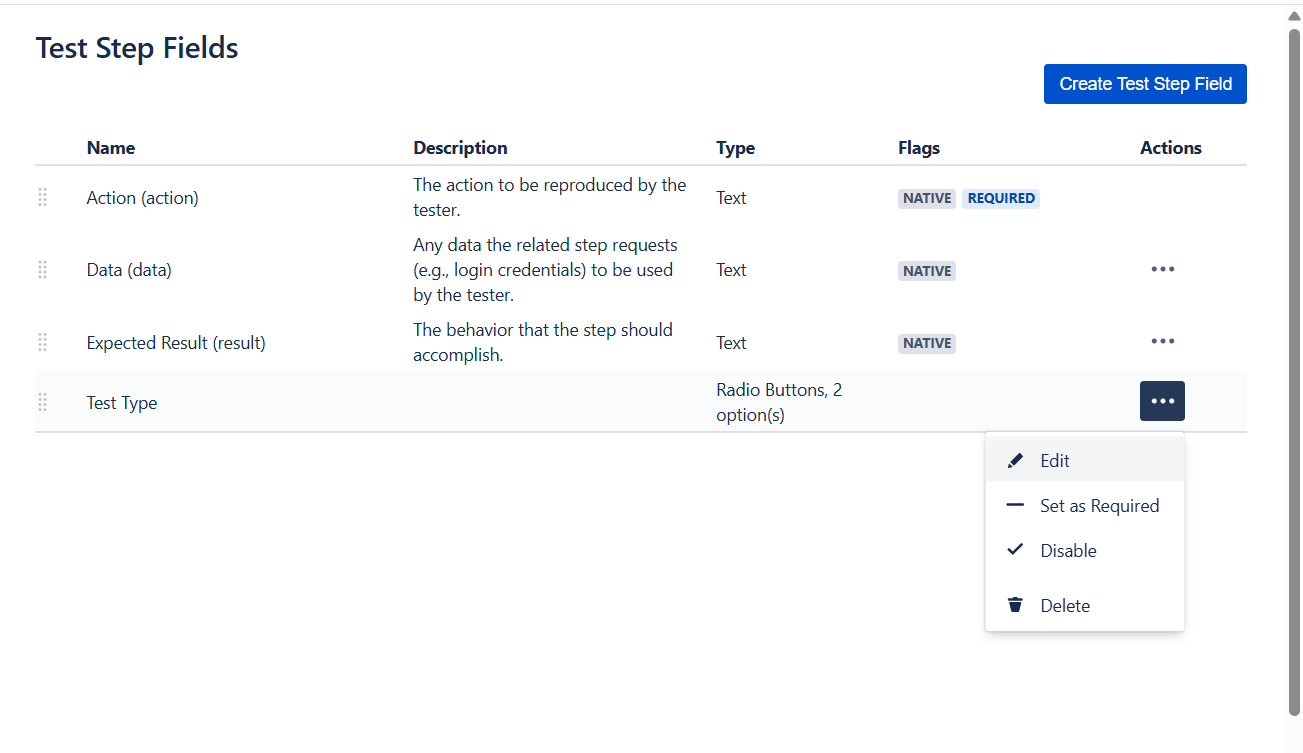


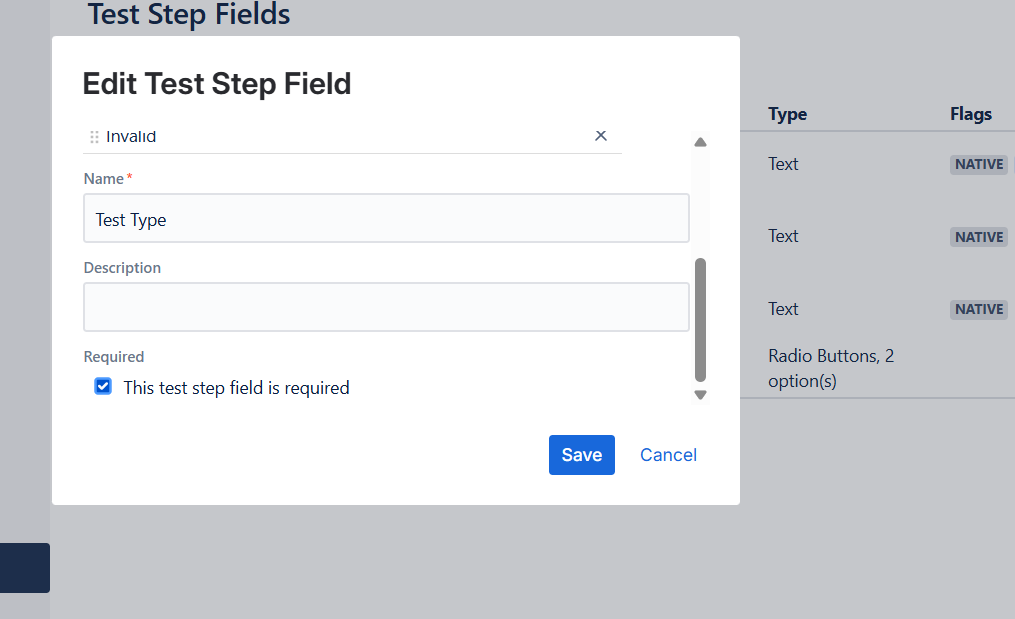
**Custom field in xray**

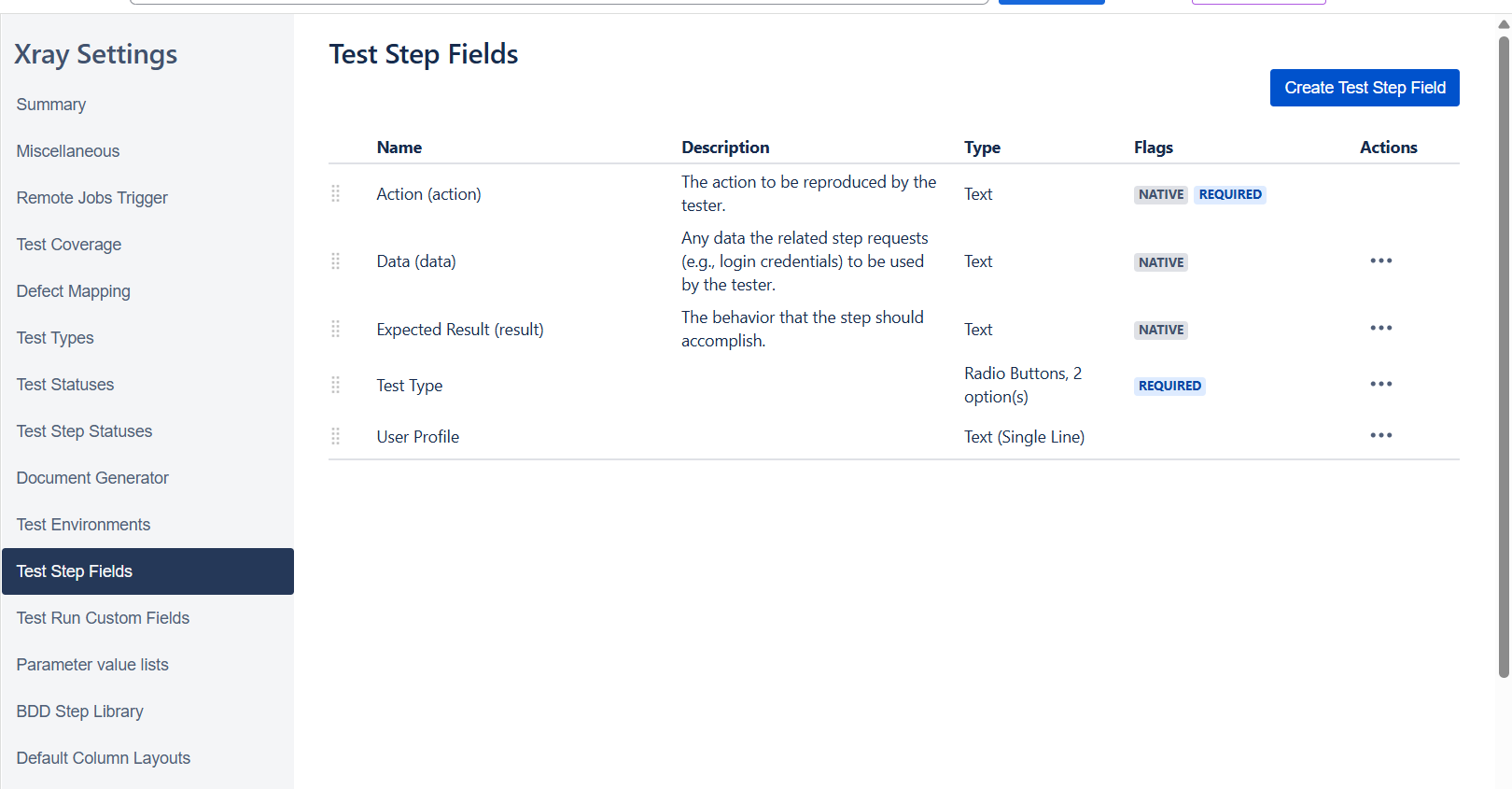
Inorder to create the custom fields we have to navigate to the xray settings through the space settings and then to the test step fields

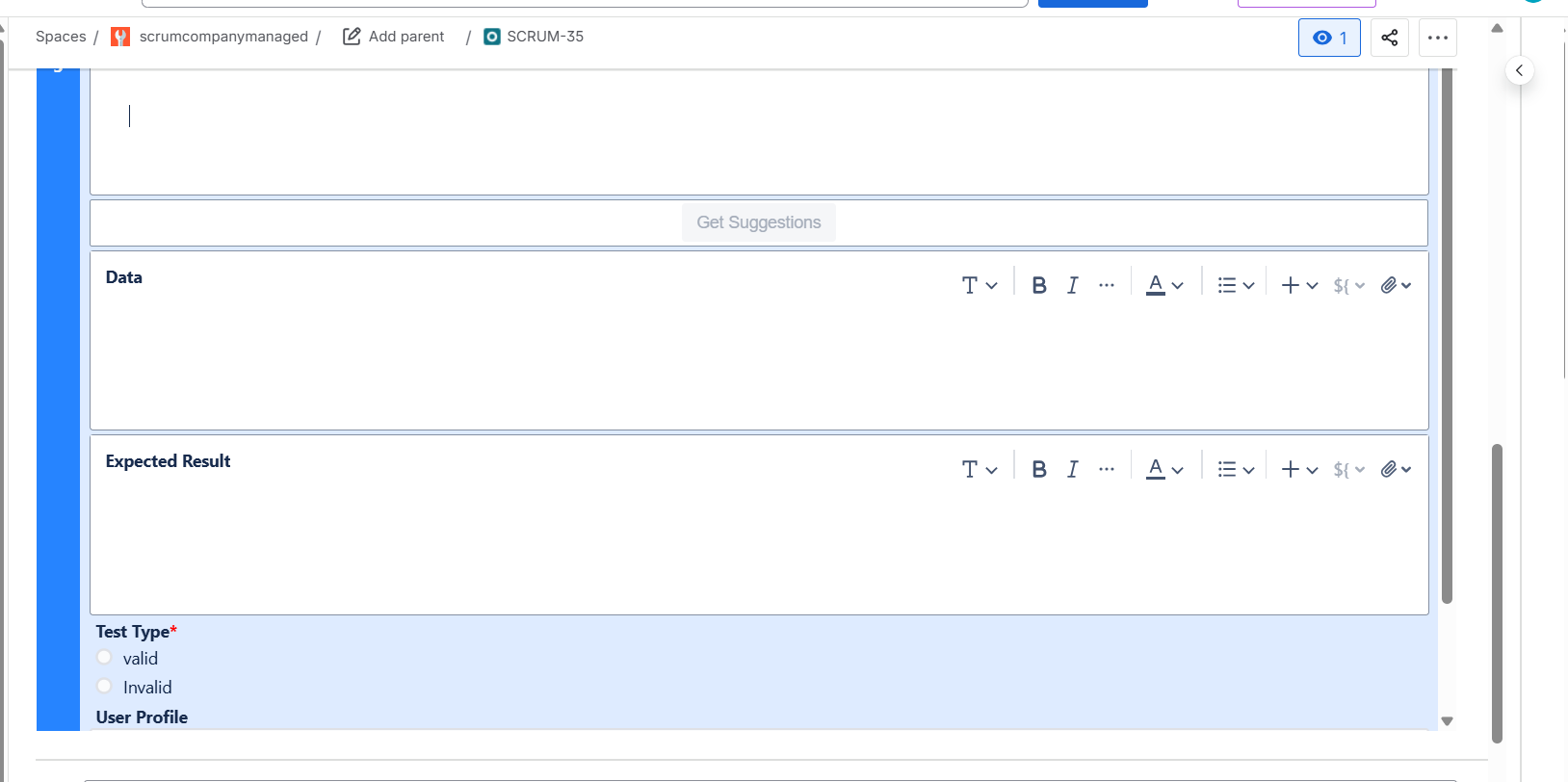


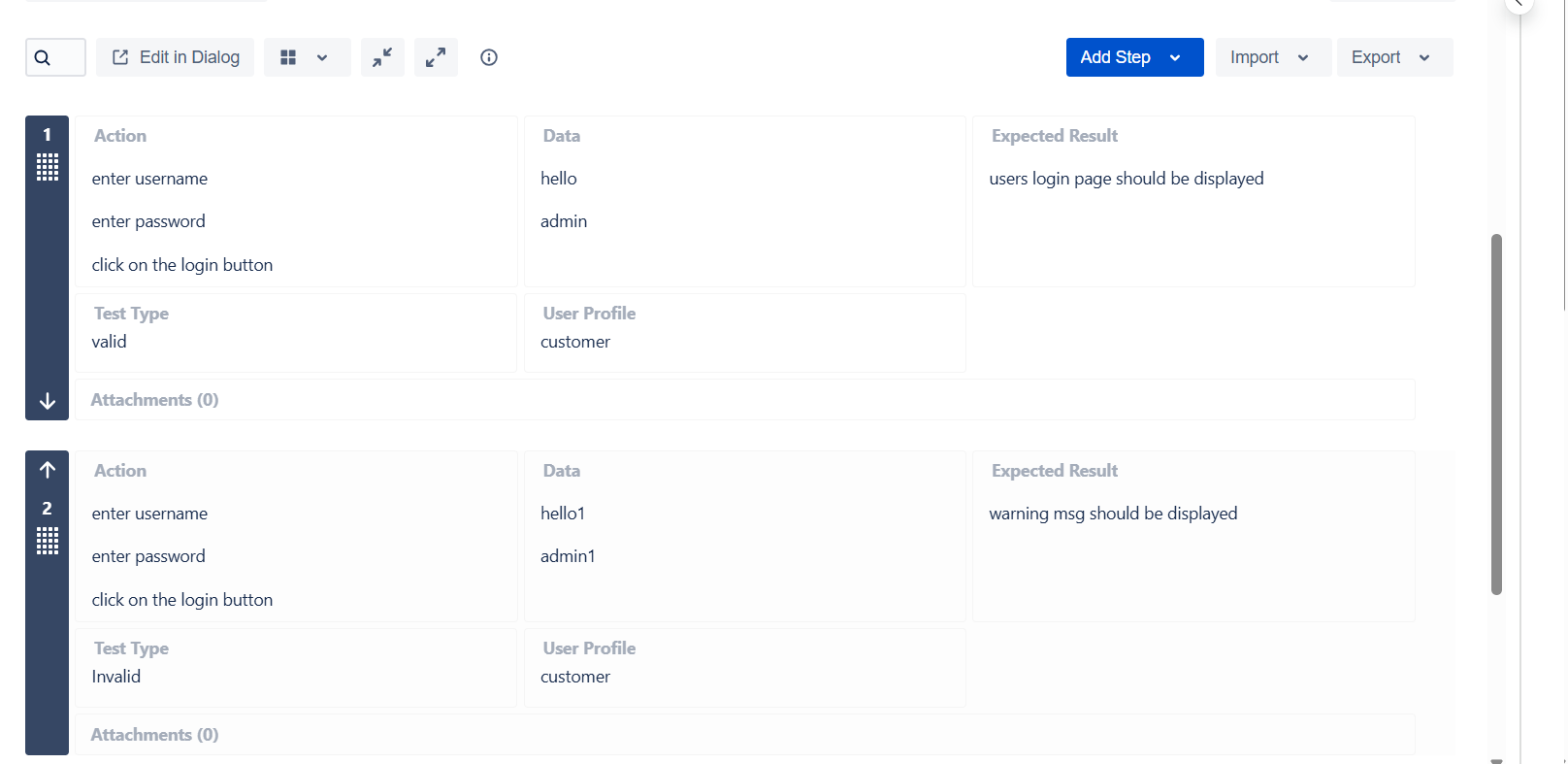




if the step is mandatory we have to make it required by checking the checkbox

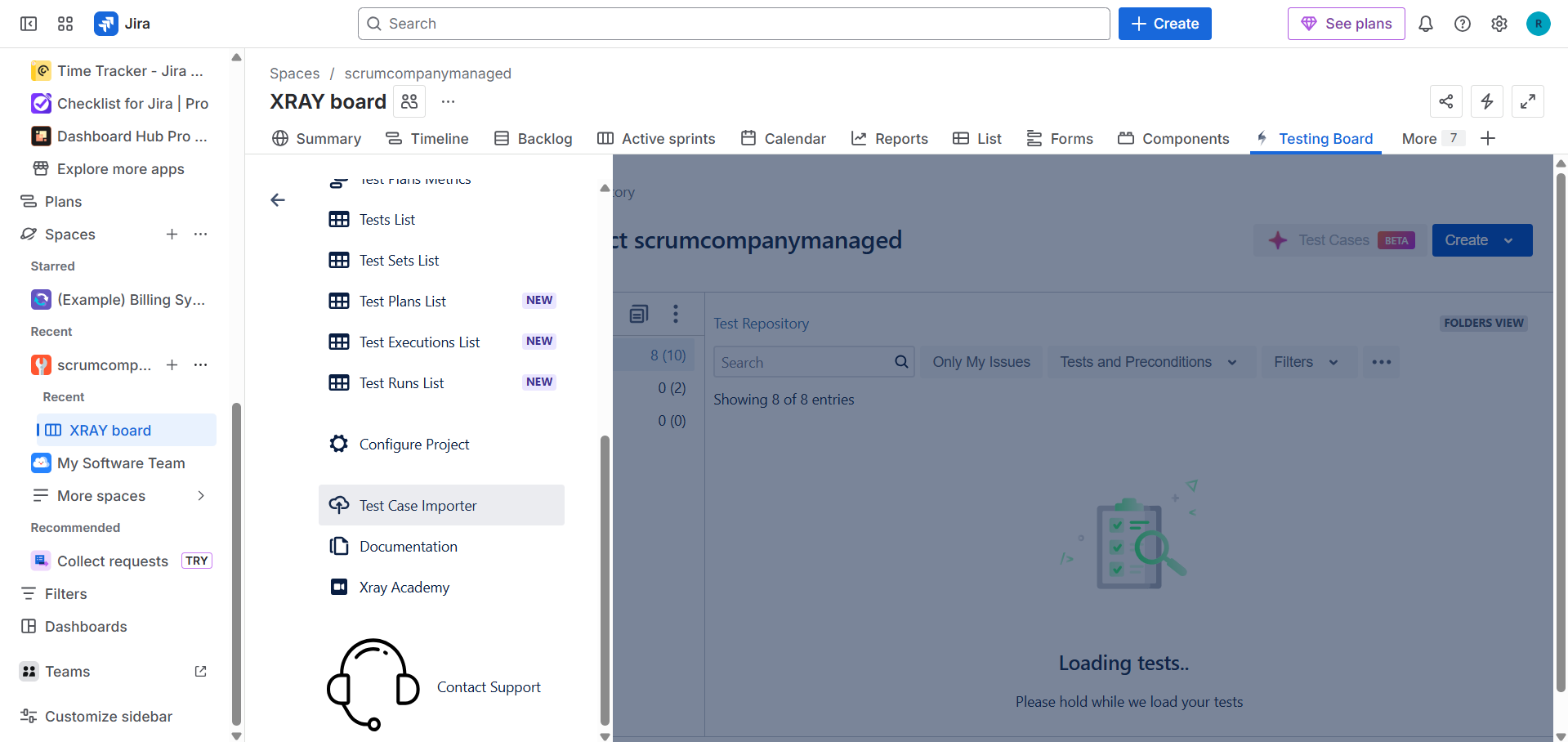




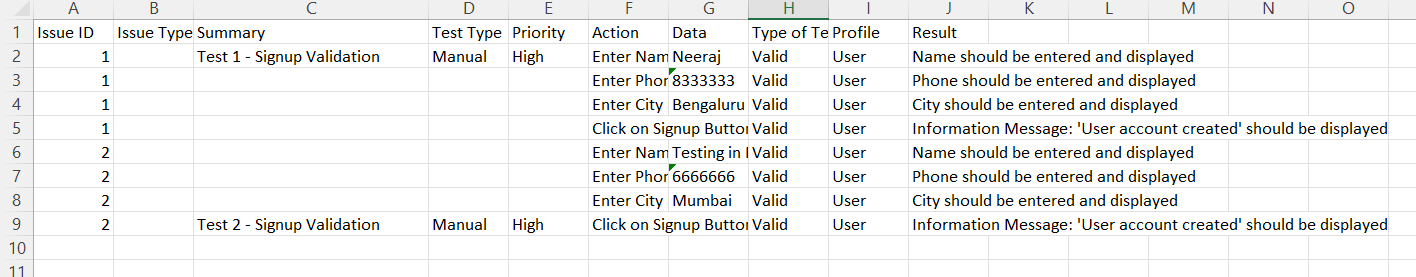


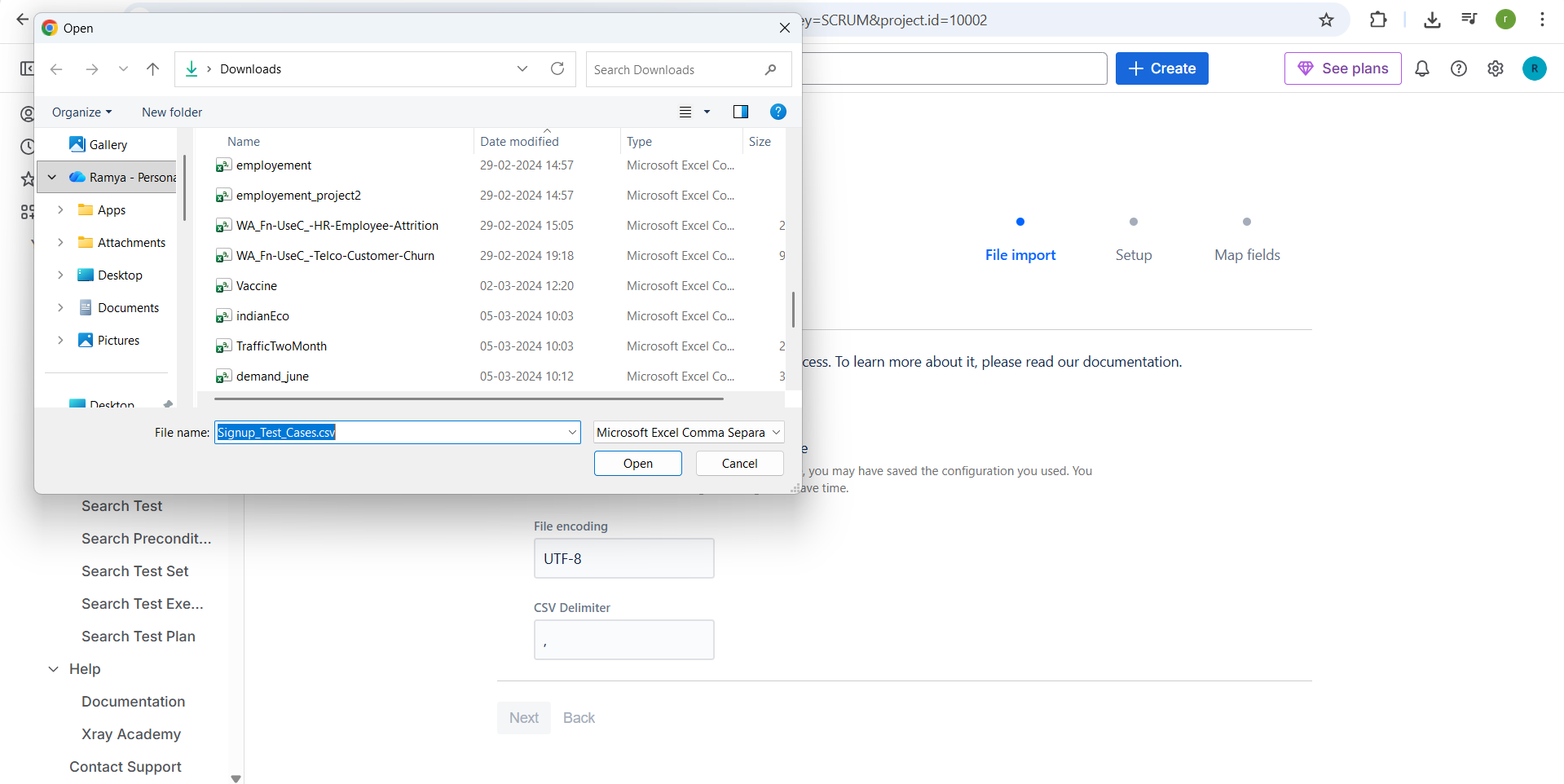
**Importing test cases from the excel**

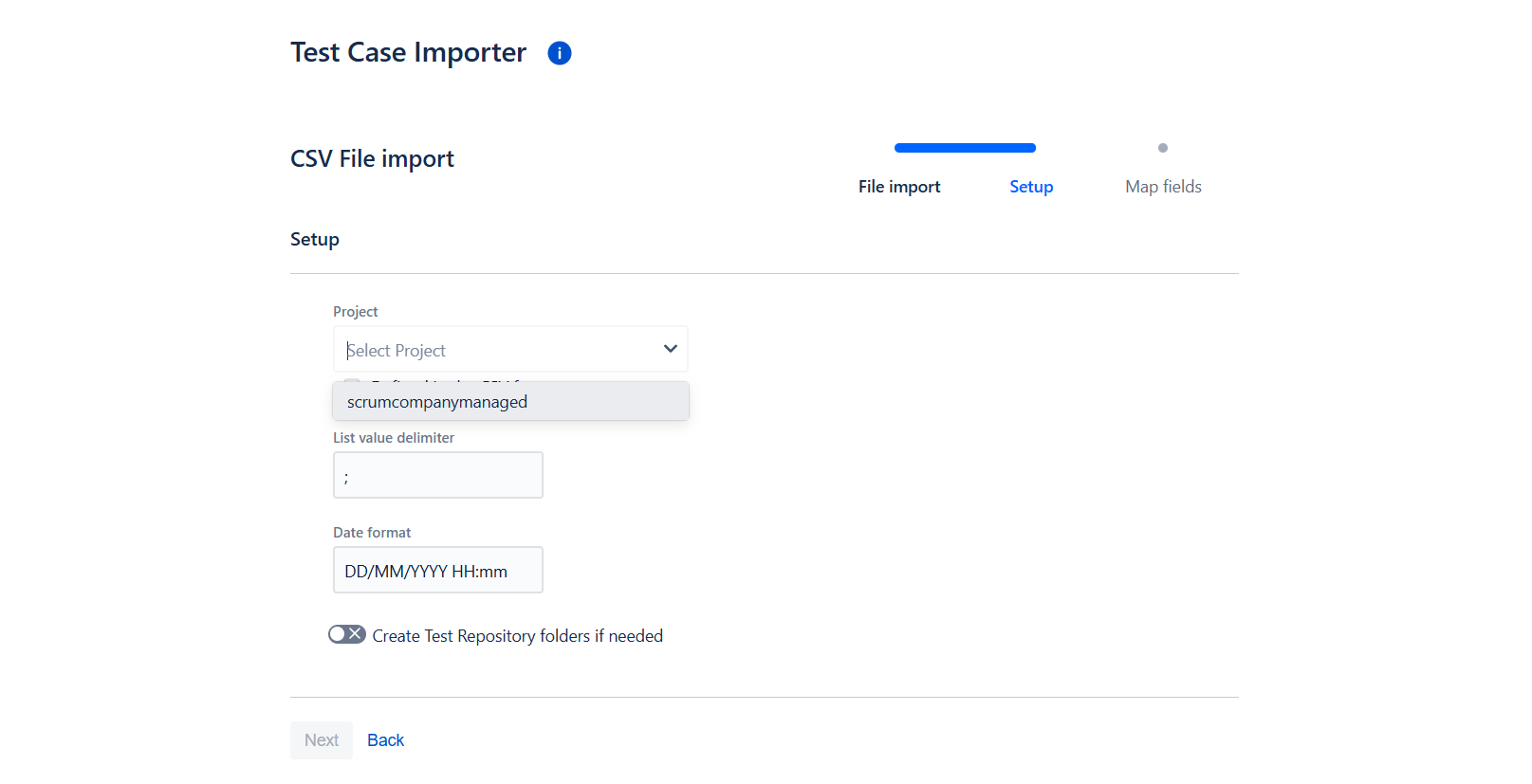
Navigate to the testing board

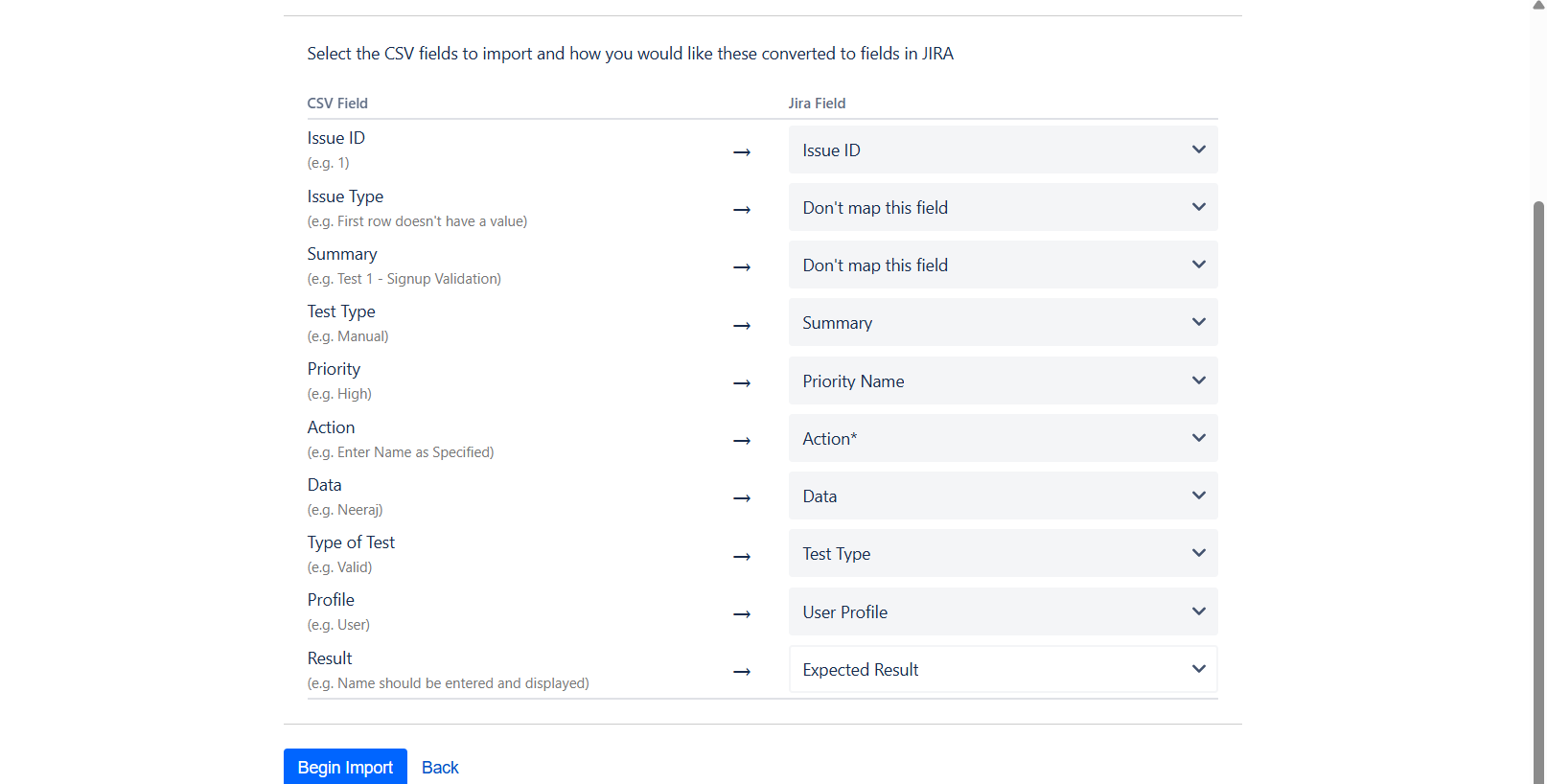


Preparation of data



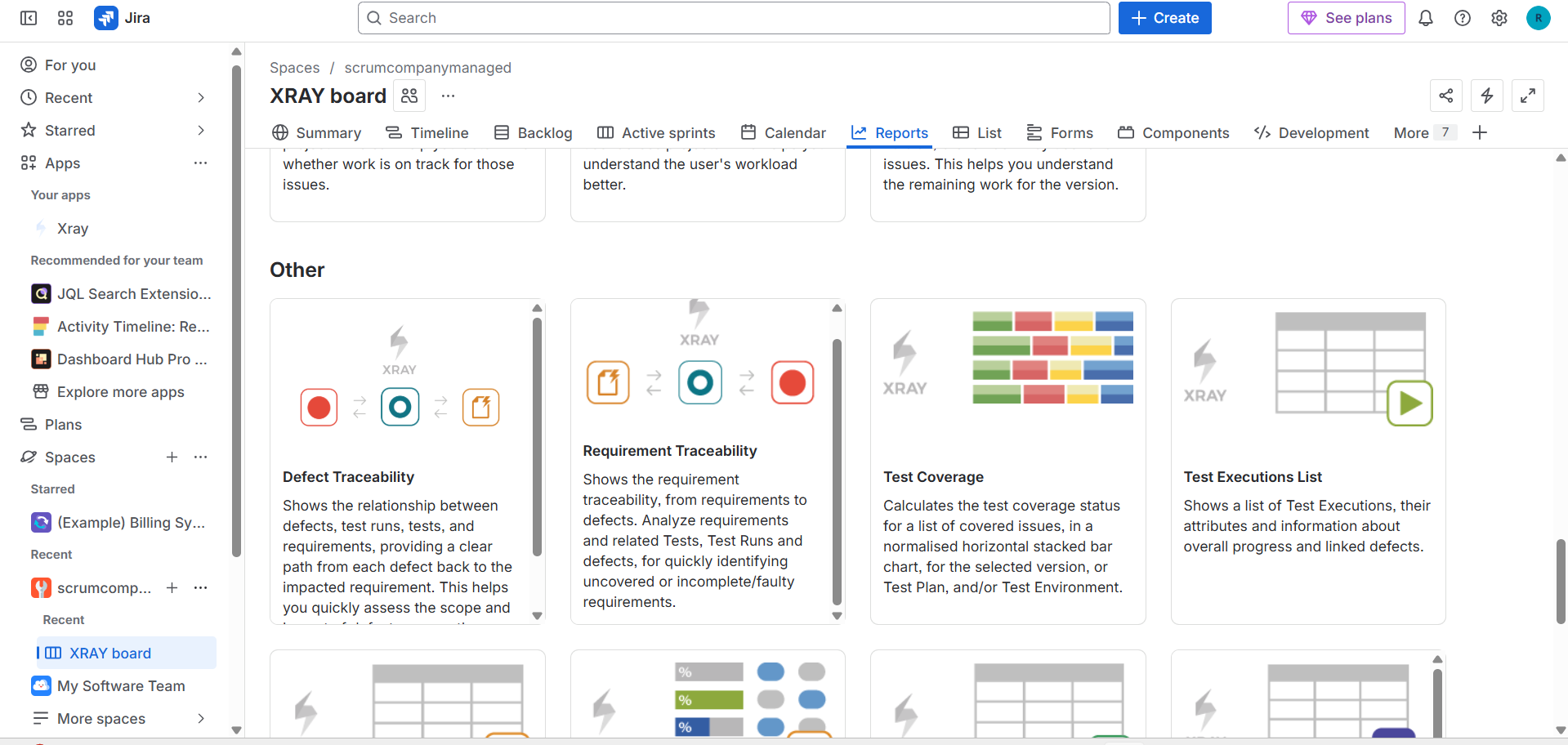


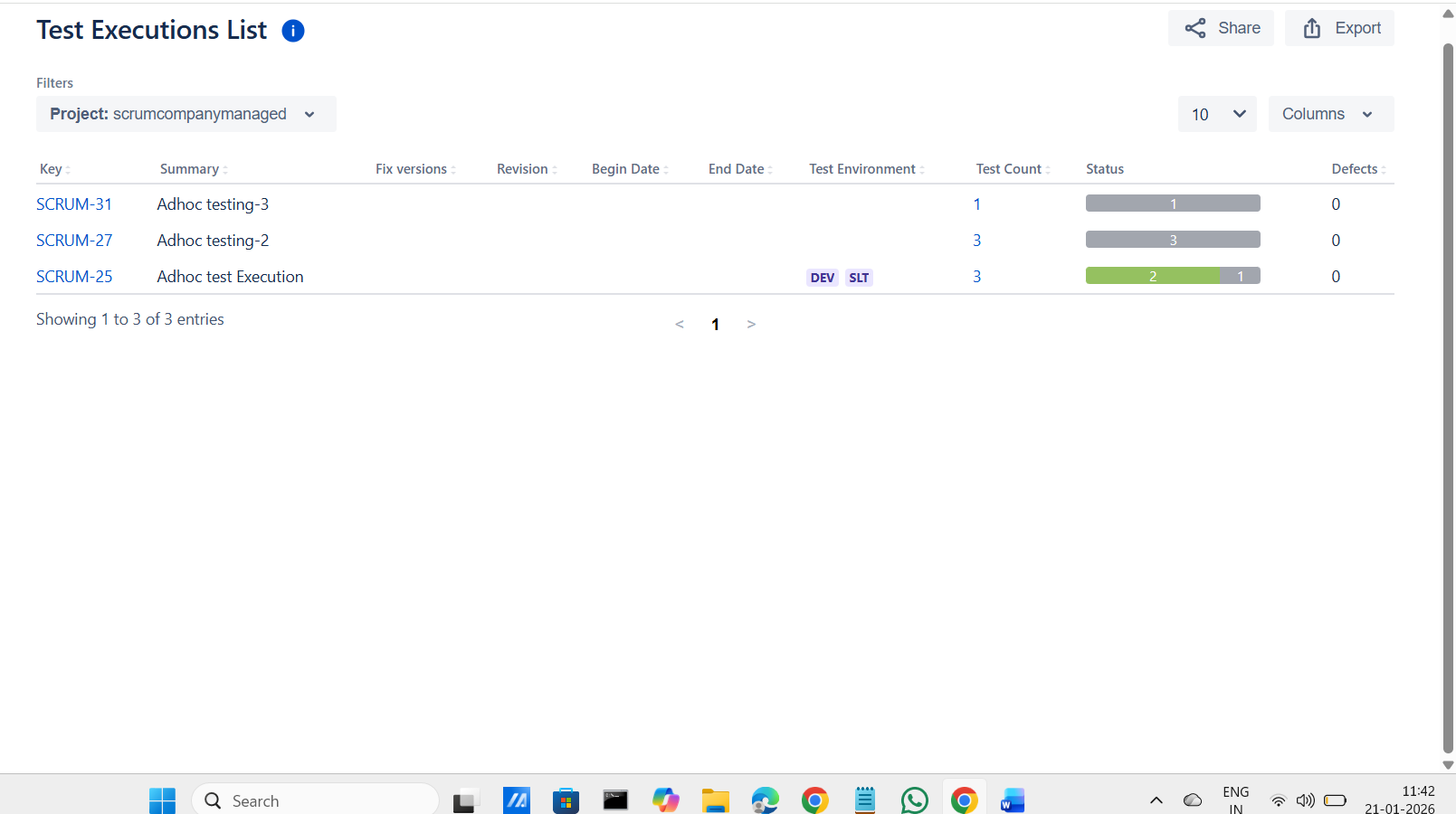
map as per the excel



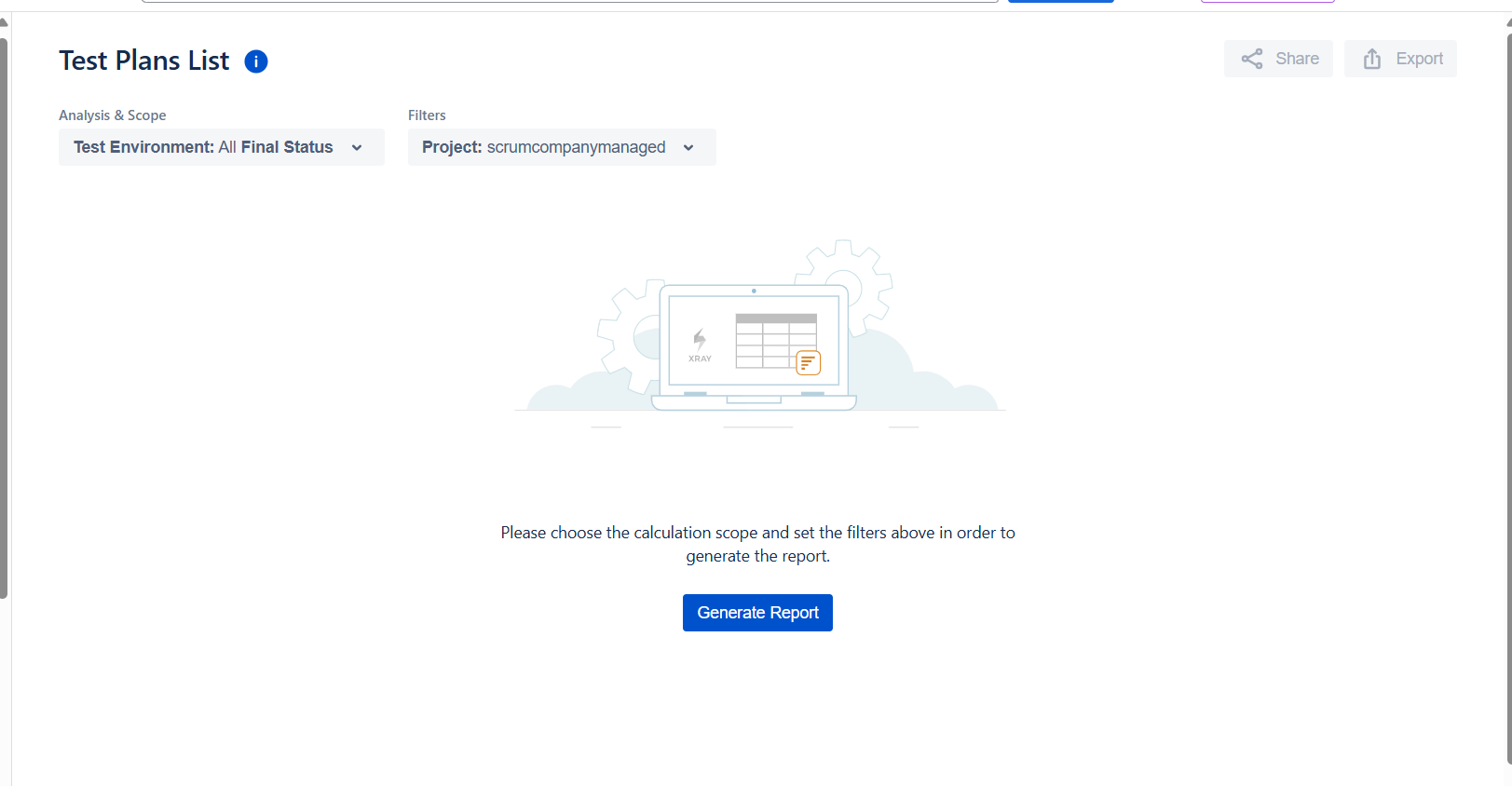
**Xray reports**

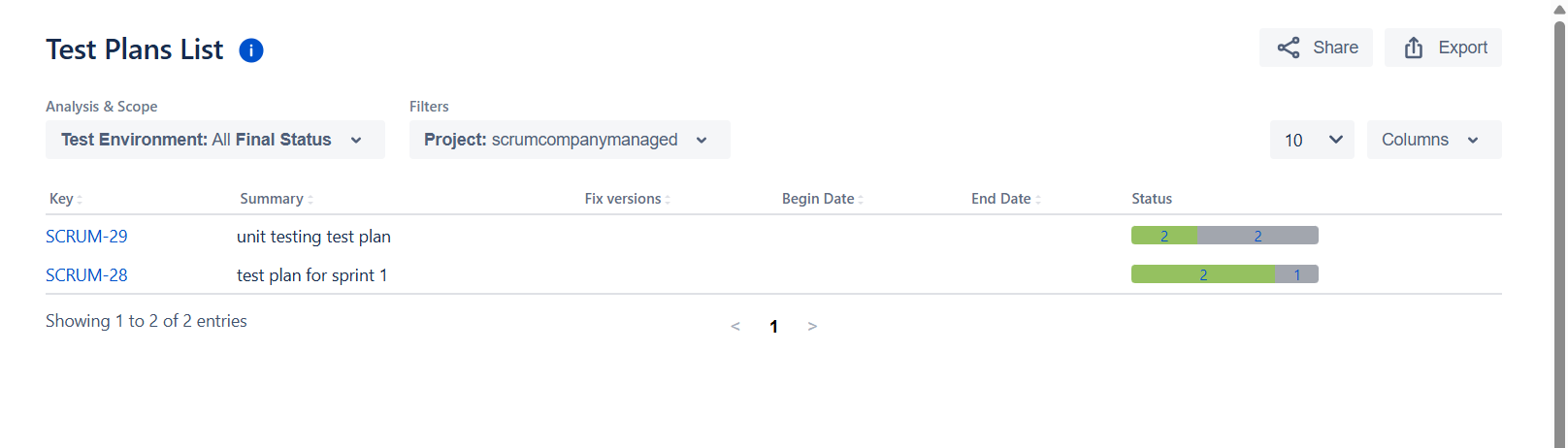
Test execution list



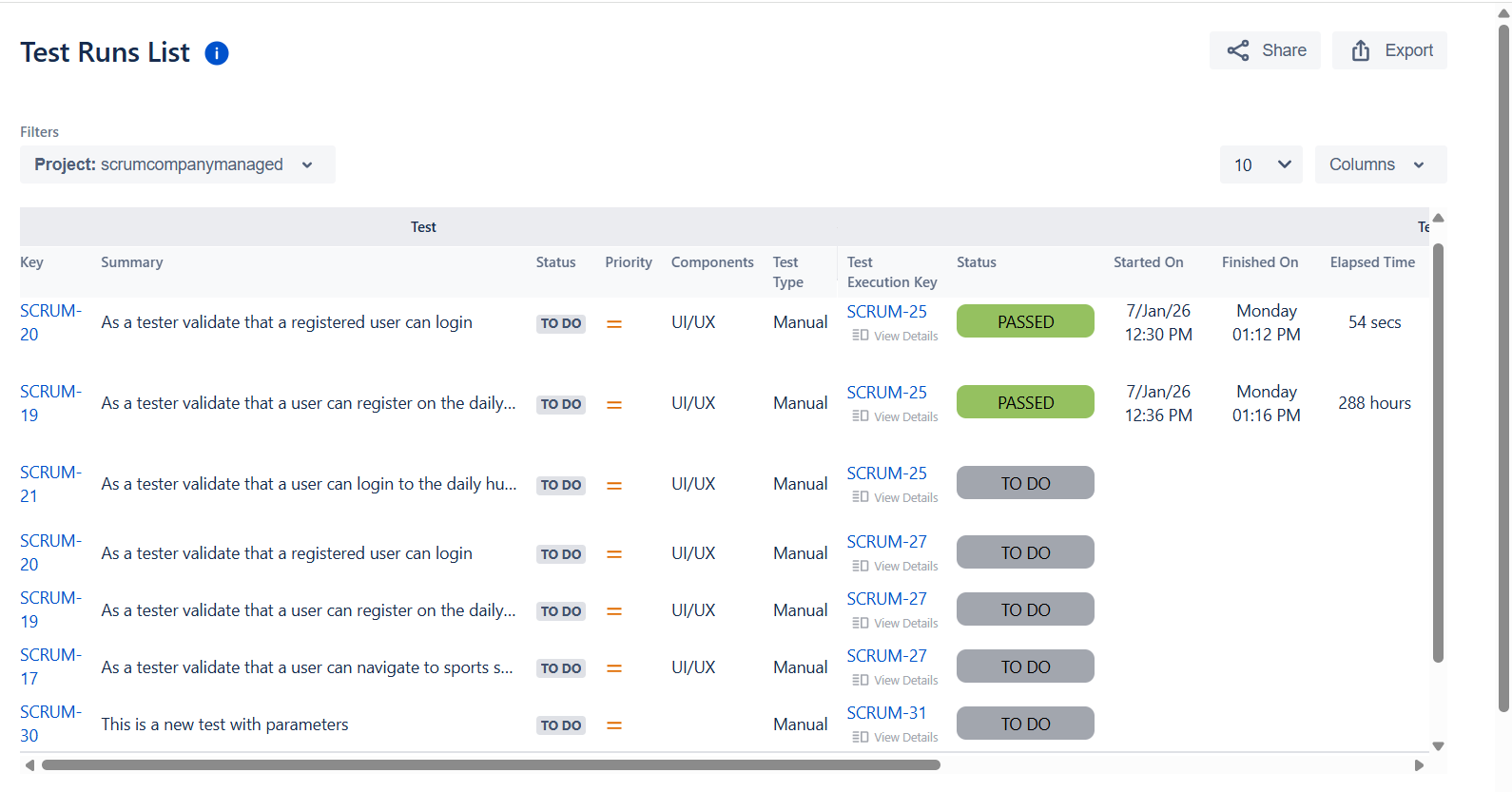


Test plan list

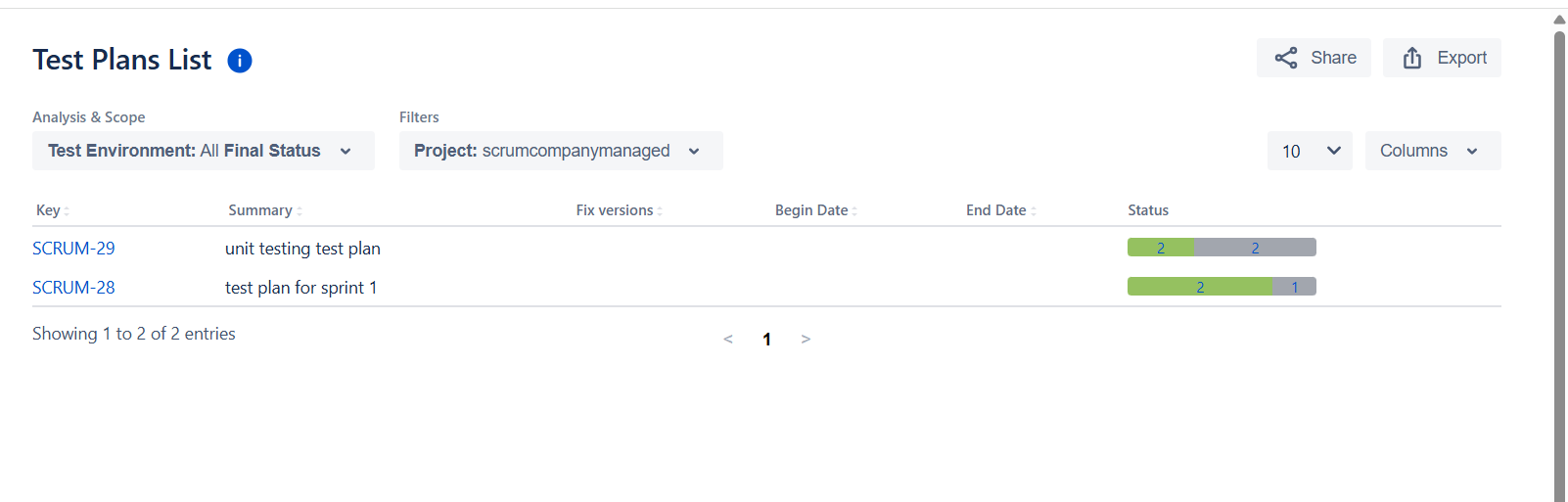


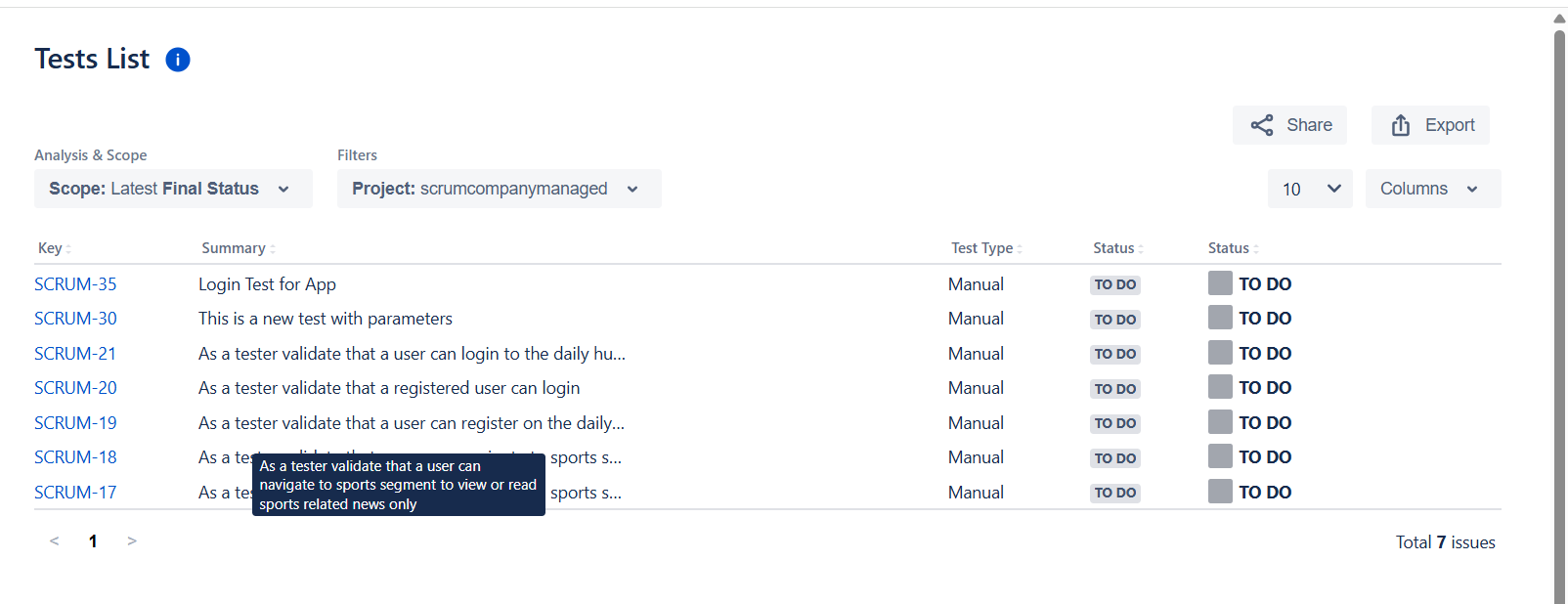


Test run list



test set lists

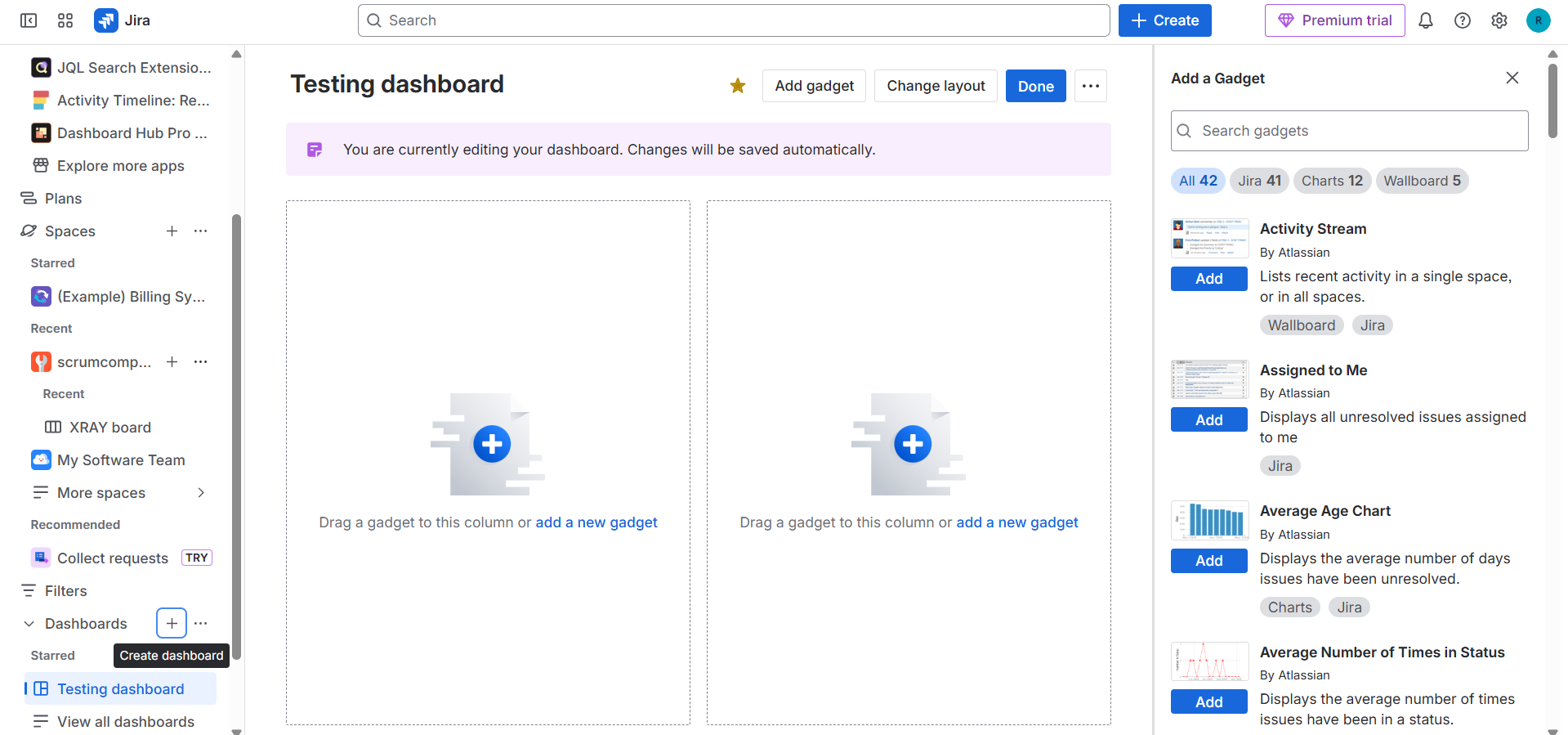




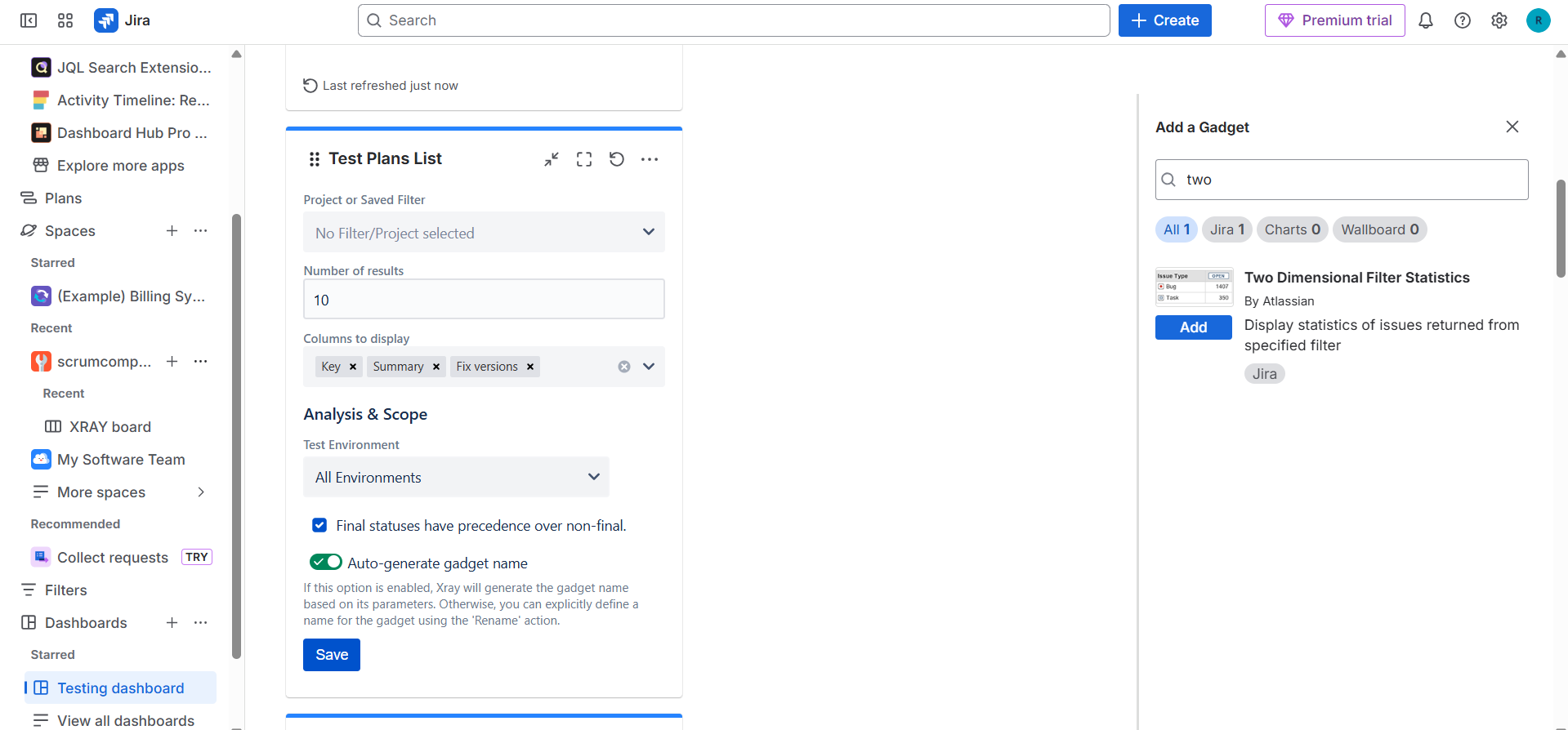
Test coverage report



**Dashboards**



Adding gadgets to the dashboard



Configuration of test coverage gadget in the dashboard

