Extending and integrating with Xray using Automation for Jira

Overview

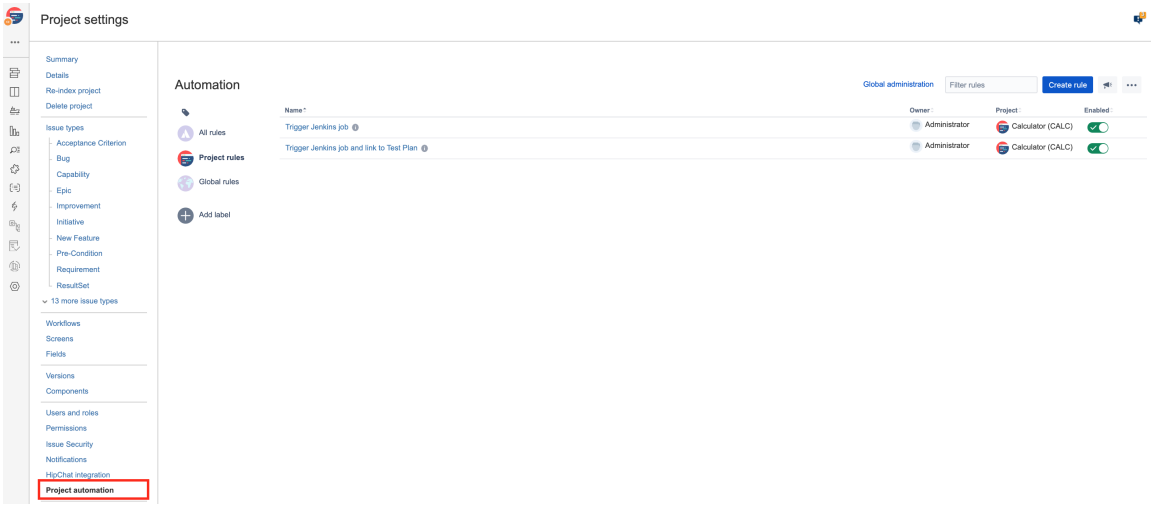
Automation for Jira app enables users to easily extend and implement automation in Jira without having to code.

This way, users can implement rules that are triggered upon some event, executed if certain condition(s) are met and, that perform certain action(s).

Rules can also be triggered manually or may be scheduled.

Since Xray uses issue types for most of its entities and since Xray provides many JQL functions that allow you to obtain testing-related information, Automation for Jira can be used with Xray in a very straightforward way.

Automation rules are available and, can be created, from the project settings, namely from the "Automation" tab.



> **Please note**
>
> The following examples are provided as-is, no warranties attached; use them carefully.
>
> Please feel free to adapt them to your needs.
>
> Note: We don't provide support for Automaton for Jira; if you have doubts concerning its usage, please contact Automation for Jira's support.

Usage Examples

Trigger a Jenkins project build from an issue

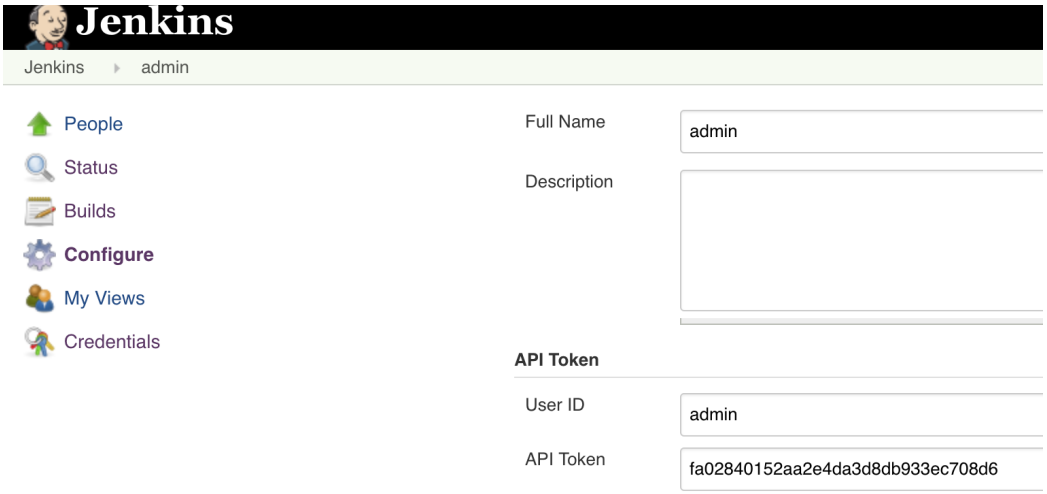
In this very simple scenario, we'll implement a rule, triggered manually, that will trigger a Jenkins project/job. The action will be available from within the "More" menu, in all issues of the selected project.

We're assuming that:

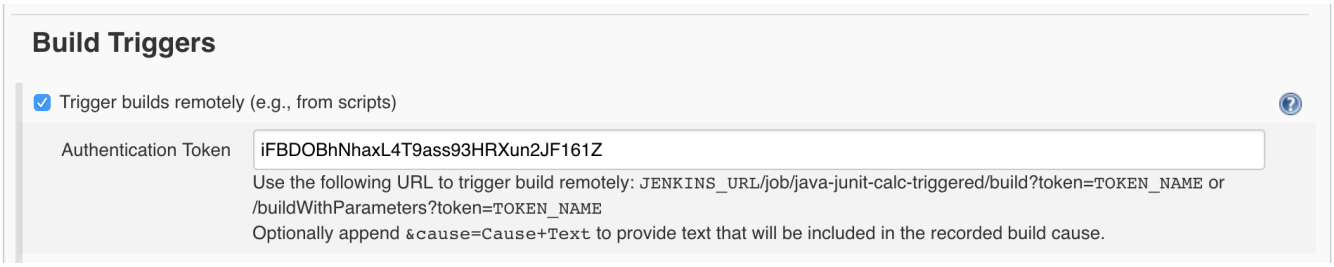
- you just want to trigger a CI job, period; this job may be totally unrelated to the issue from where you triggered it
- what the CI job will do, including if it will report the results back to Xray or not, is not relevant

Jenkins configuration

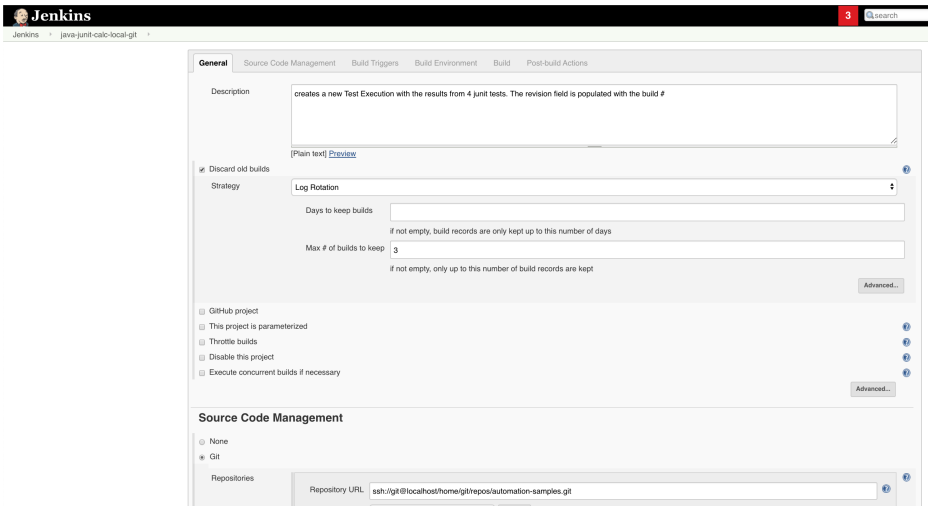
In Jenkins, we need to generate an API token for some user, which can be done from the profile settings page.



At the project level, we need to enable remote build triggers, so we can obtain an "authentication token" to be used in the HTTP request afterwards.

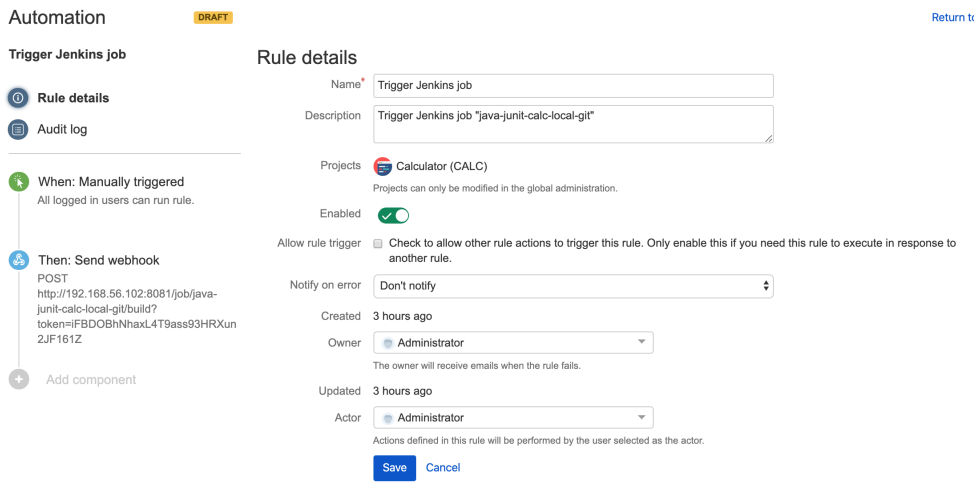


The project itself is a normal one, without parameters.

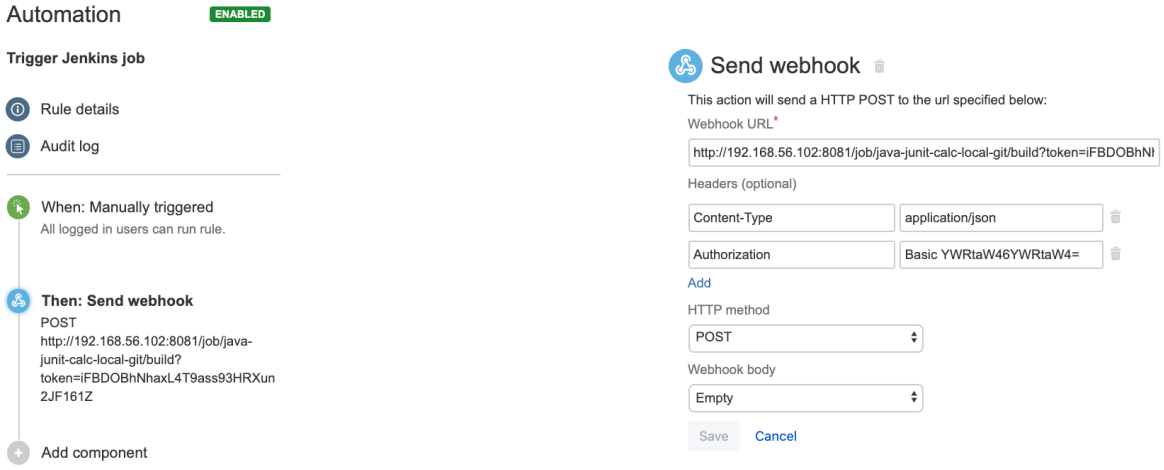


Automation for Jira configuration

1. create a new rule and define the "When" (i.e. when it to should be triggered), to be "Manually triggered"

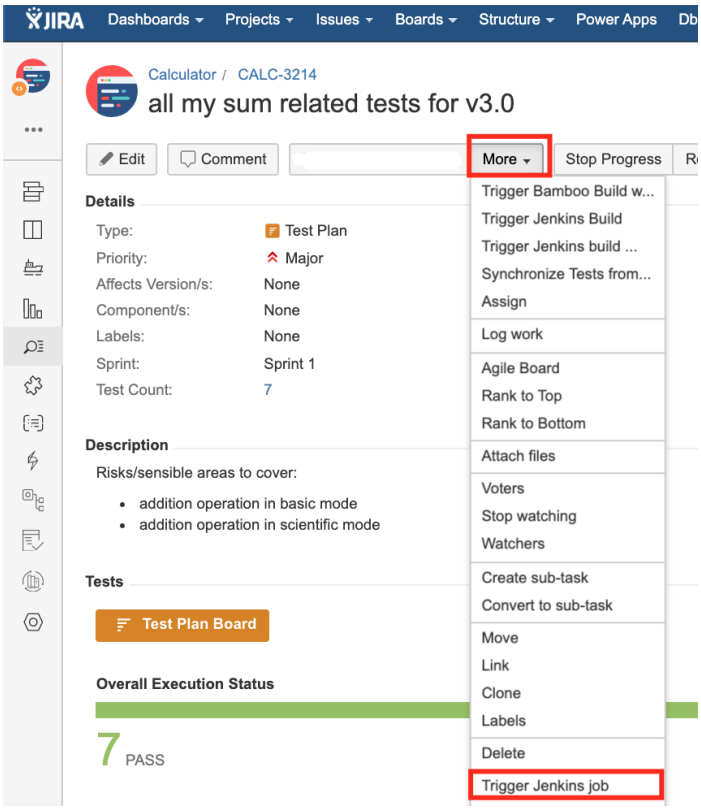


2. define an action (i.e. the "Then") as "Send webhook" and configure it as follows



- the Webhook URL provided above follows this syntax:
  - <jenkins_base_url>/job/<name_of_jenkins_project_job>/build?token=<token>
- besides the "Content-Type" header that should be "application/json", define also an "Authorization" header having the value "Basic <auth>", where the base64 encoded <auth> can be generated using your Jenkins API credentials

After publishing the rule, you can go to the screen of an issue and trigger the Jenkins project/job.



Trigger a Jenkins project build from a Test Plan and report the results back to it

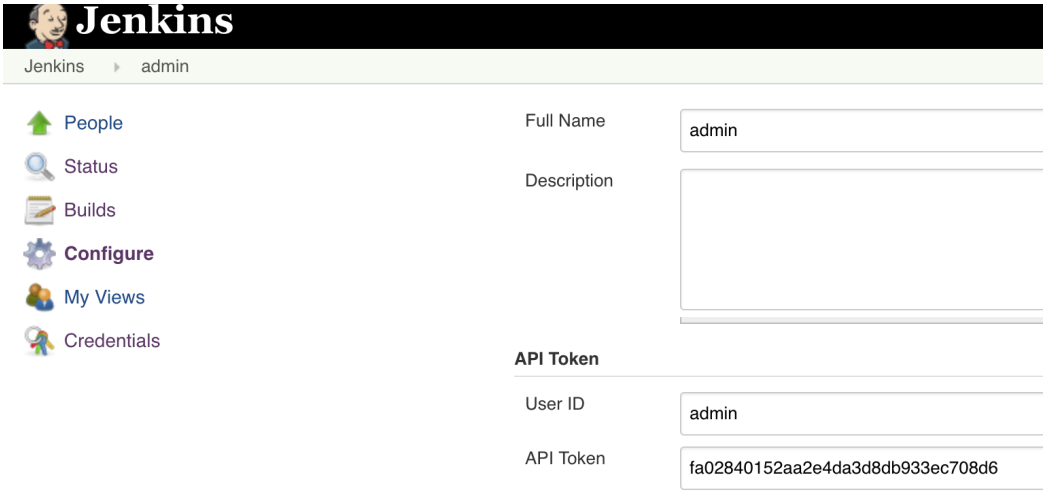
In this simple scenario, we'll implement a rule, triggered manually, that will trigger a Jenkins project/job. The action will be available from within the "More" menu, for all Test Plan issues of the selected project.

We're assuming that:

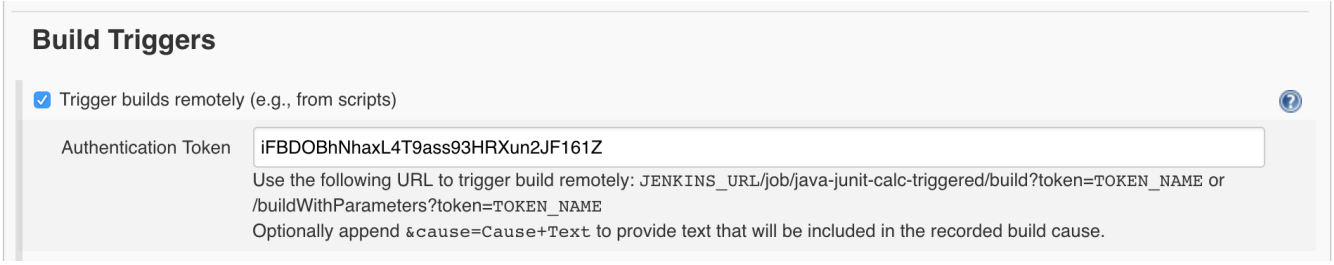
- you just want to trigger a CI job, period; this job may be totally unrelated to the issue from where you triggered it
- the results will be submitted back to Xray, if the project is configured to do so in Jenkins

Jenkins configuration

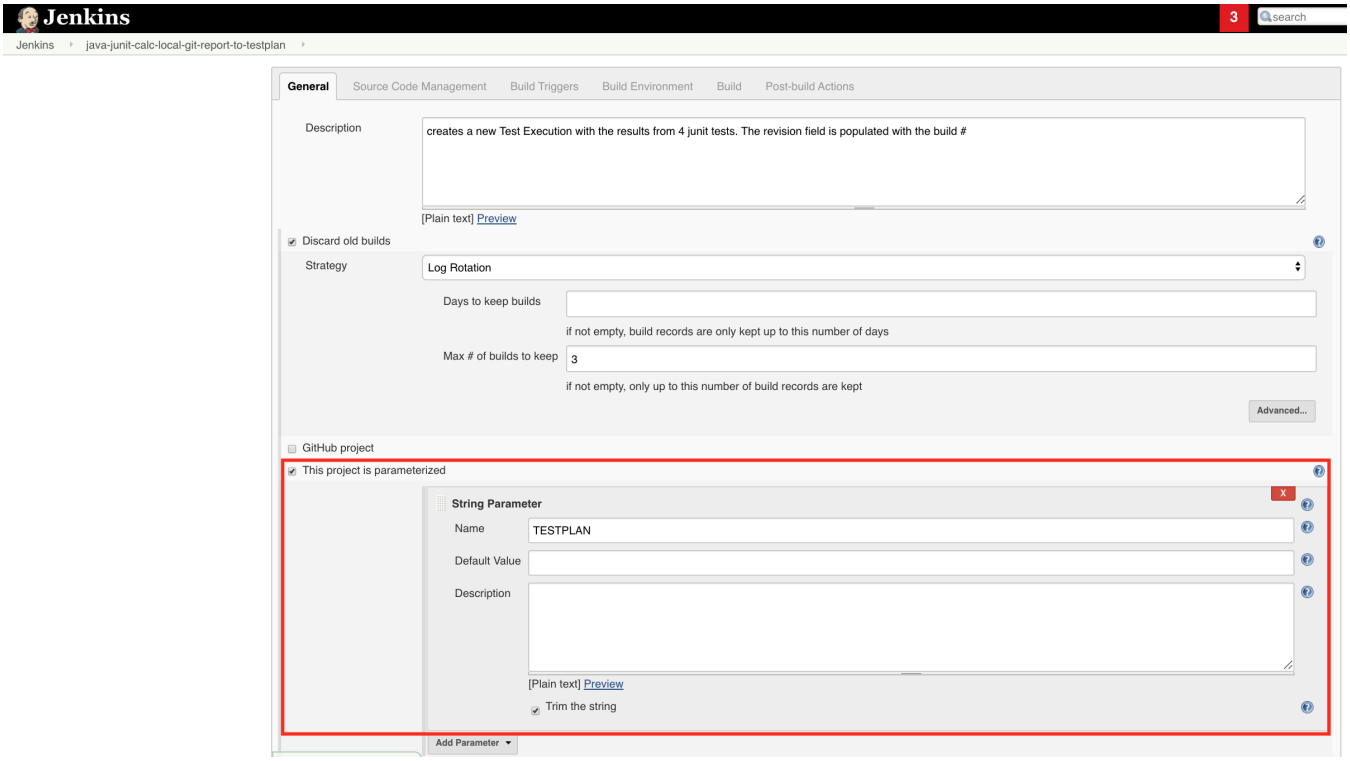
In Jenkins, we need to generate an API token for some user, which can be done from the profile settings page.



At the project level, we need to enable remote build triggers, so we can obtain an "authentication token" to be used in the HTTP request afterwards.

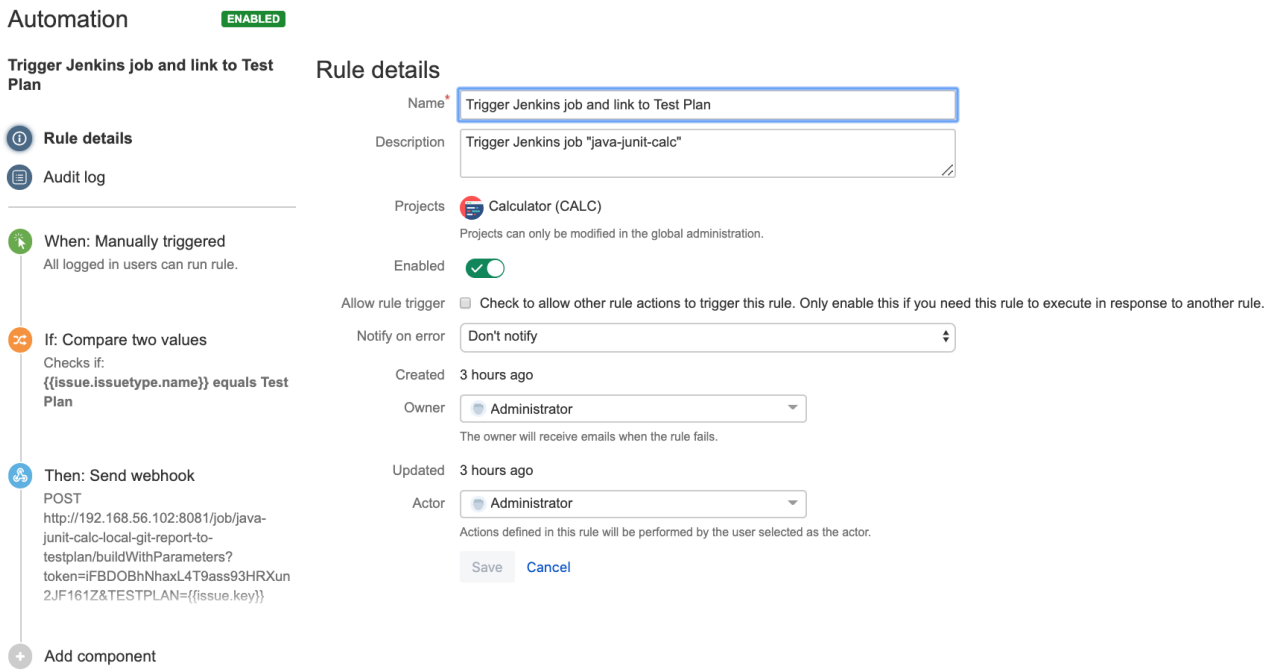


The project itself is a normal one; the only thing relevant to mention is that this project is a parameterized one, so it receives TESTPLAN, that in our case will be coming from Jira.

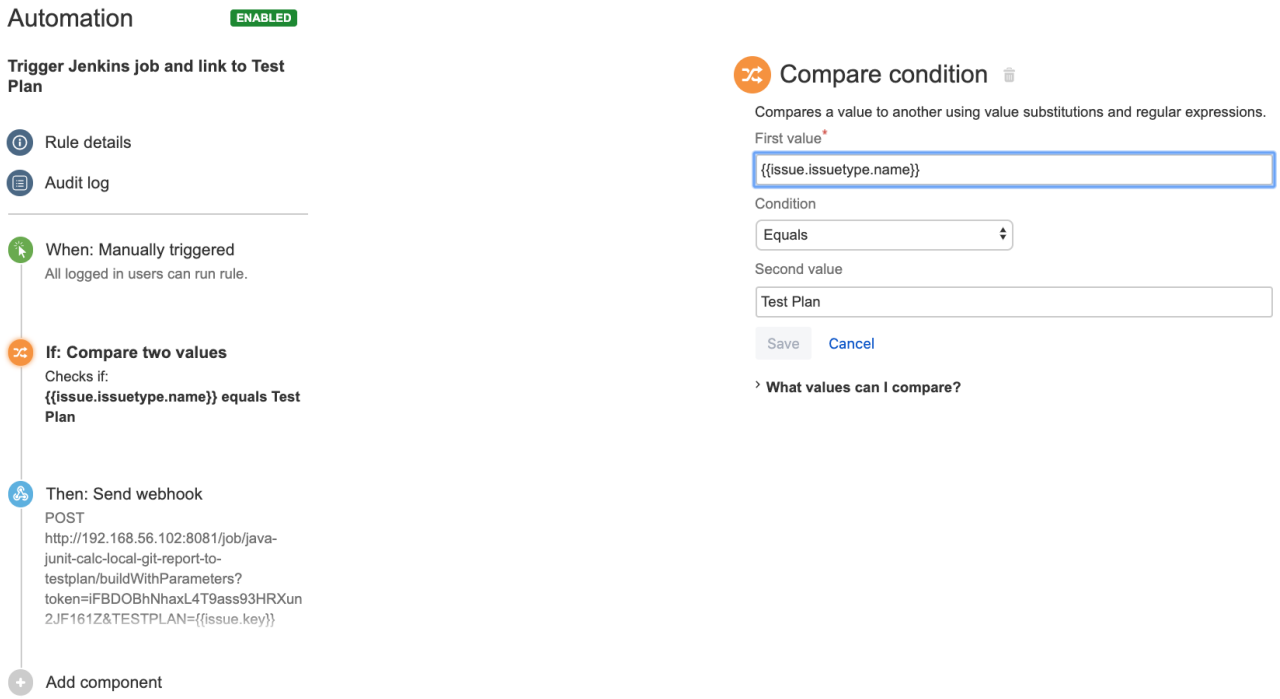


Automation for Jira configuration

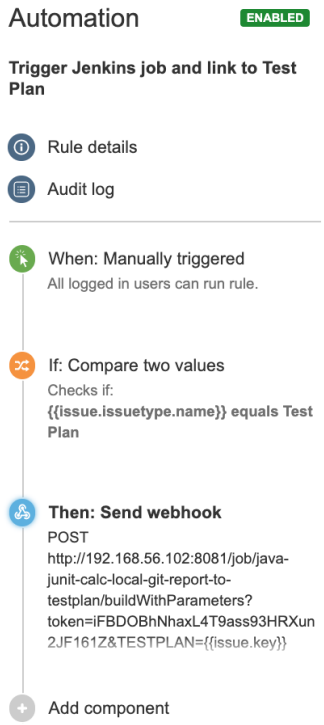
1. create a new rule and define the "When" (i.e. when it to should be triggered), to be "Manually triggered"

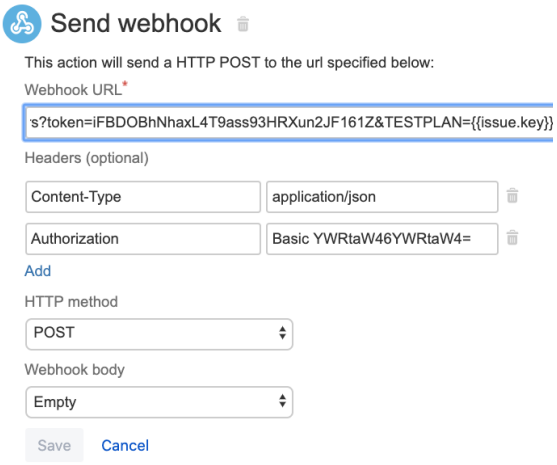


2. define the condition so that this rule can only be executed from Test Plan issue



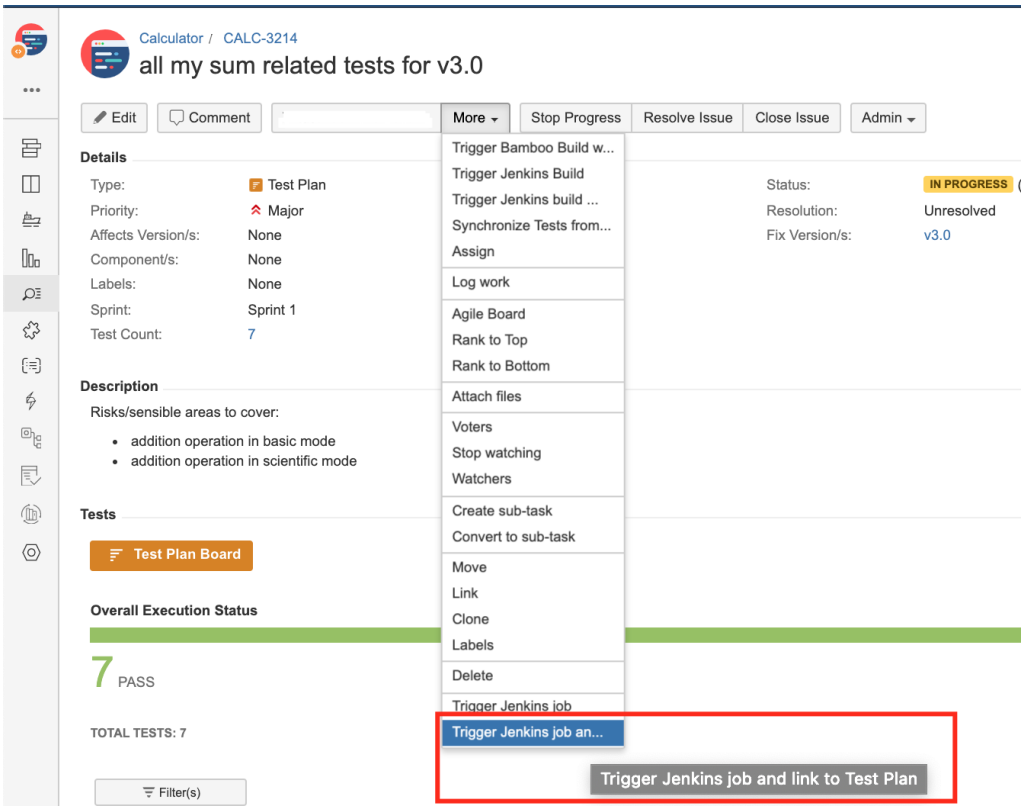
3. define an action (i.e. the "Then") as "Send webhook" and configure it as follows





- the Webhook URL provided above follows this syntax:
  - <jenkins_base_url>/job/<name_of_jenkins_project_job>/buildWithParameters?token=<token>&TESTPLAN={{issue.key}}
- besides the "Content-Type" header that should be "application/json", define also an "Authorization" header having the value "Basic <auth>", where the base64 encoded <auth> can be generated using your Jenkins API credentials

After publishing the rule, you can go to the screen of an issue and trigger the Jenkins project/job.



References

- Automation for Jira in the Atlassian Marketplace
- Automation for Jira documentation