

Zephyr Cloud Integration document with Selenium Java framework

Table of Contents

Zephyr Cloud Integration document with Selenium Java framework	1
1. Overview	3
2. Pre-requisites for project creation.....	3
3. Generate required Zephyr authorization values	13
4. Changes needed in Sys.properties file	15
5. Changes needed in test suite	15
6. Execution and Result review	16
7. Code for customization of test rail API methods and integration of test results to test rail.....	16
7.1 Required ZfjCloudRestClient jar dependency added to pom.xml.....	16
7.2 Implementation code for customizing ZAPI methods	17
7.3 Integration code for updating test results to Zephyr	18

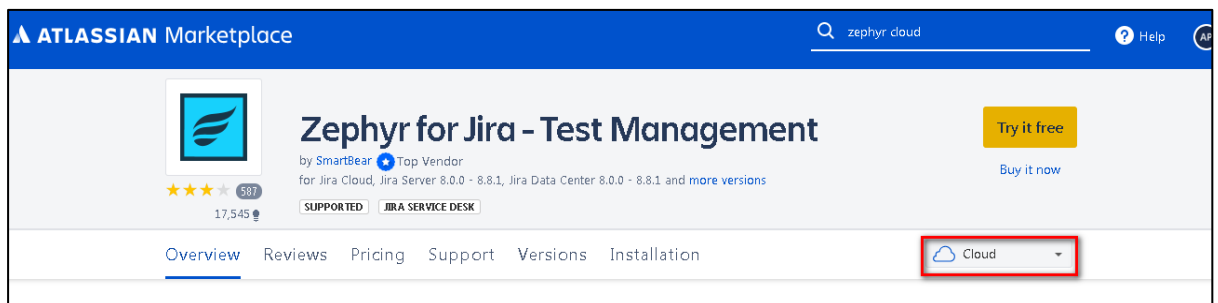
1. Overview

This document provides explanation about how to integrate the results of your selenium tests with corresponding test in Zephyr for Jira tool

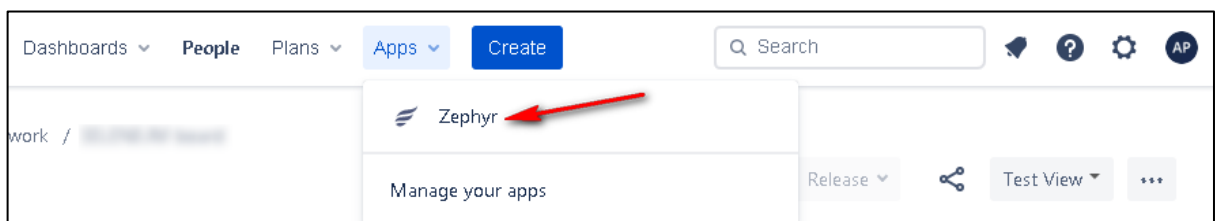
2. Pre-requisites for project creation

User need to follow below steps before using Zephyr tool such as creating a project, Creating a test case and test cycle, Adding Tests to respective cycle ...etc.

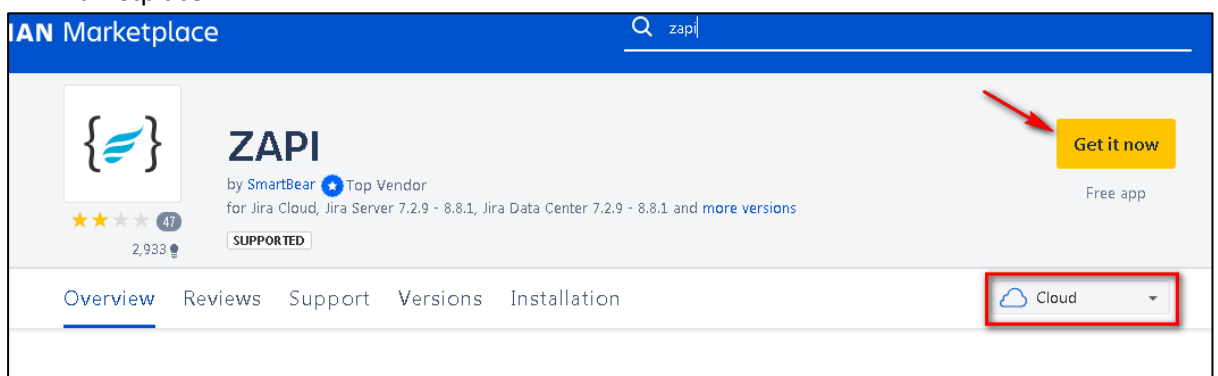
- Adding Zephyr cloud application:
 1. Go to Atlassian Marketplace at '<https://marketplace.atlassian.com/>', create account
 2. Search for 'Zephyr Cloud' application and click on 'Try it free' / 'Buy it now' buttons to add it from the Atlassian Marketplace



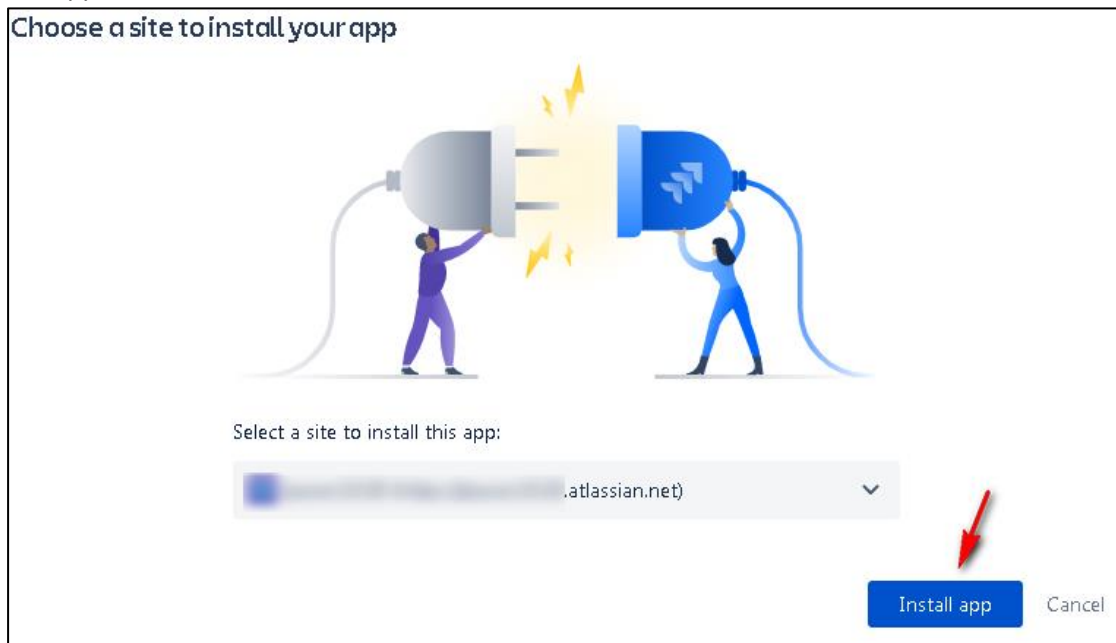
3. After clicking on 'Try it free', you can see the app added as shown below



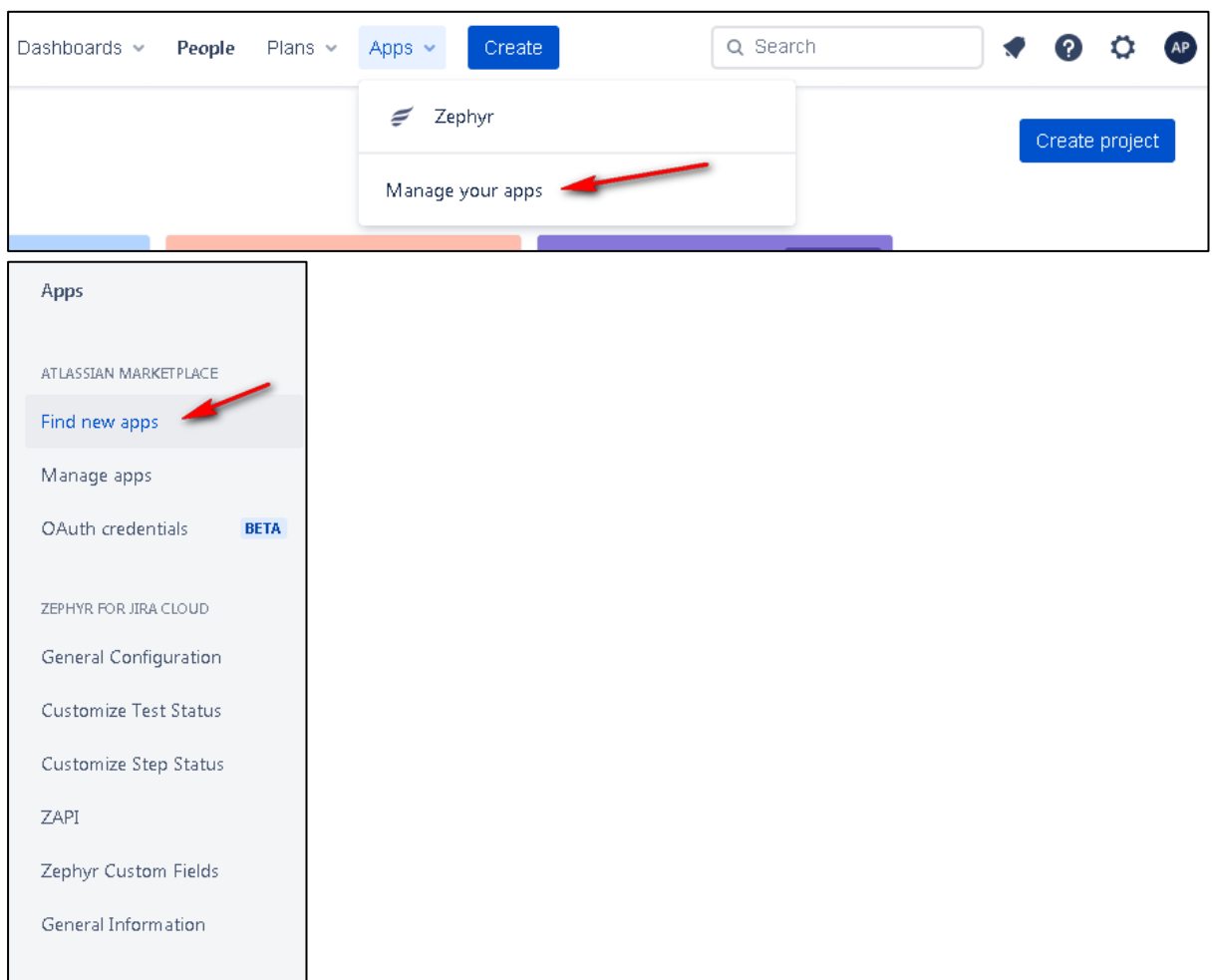
- Adding Zapi application :
 1. Search for 'ZAPI' application and click on 'Get it now' button to add it from the Atlassian Marketplace



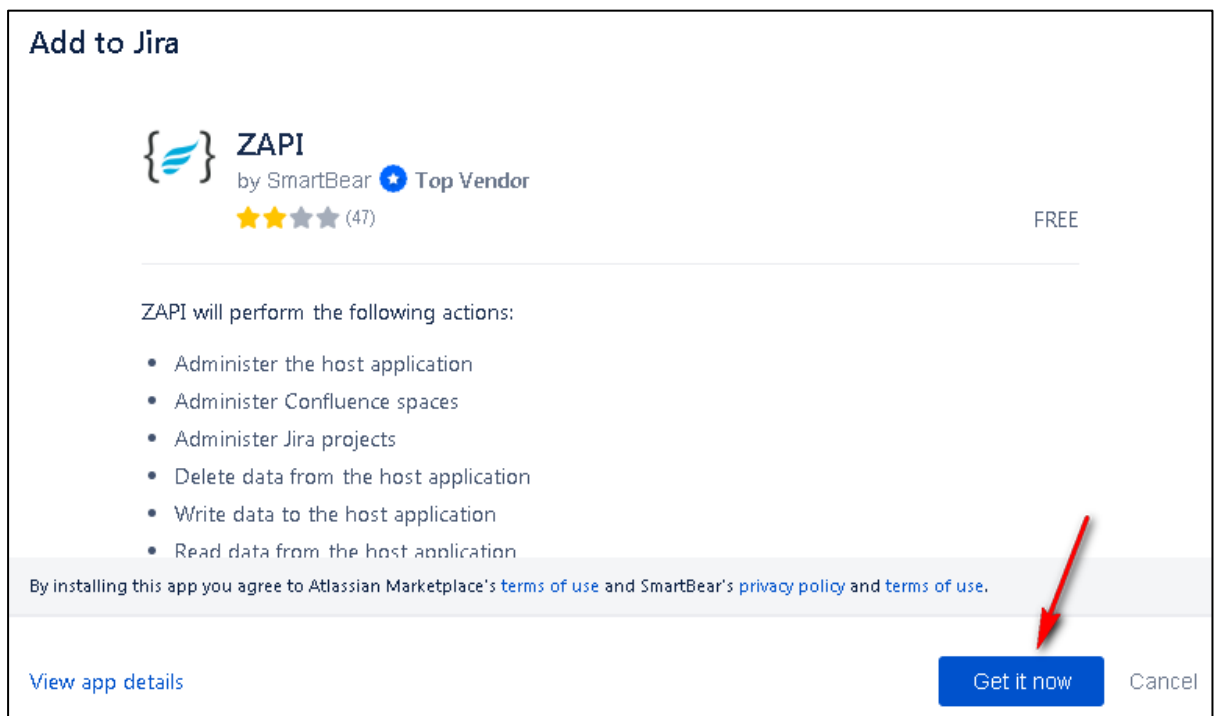
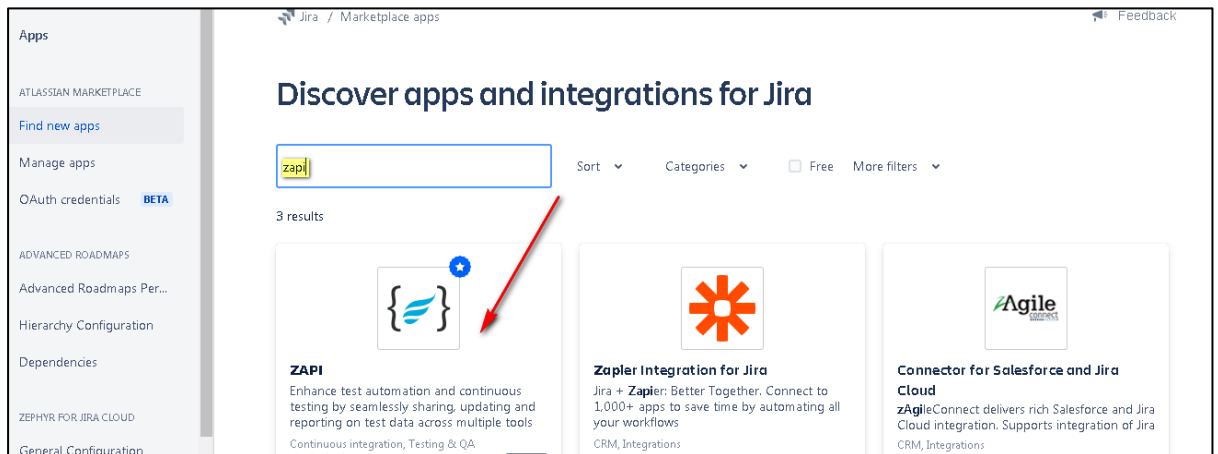
2. Now select your site (xxx.atlassian.net) from the dropdown and install the 'ZAPI' application



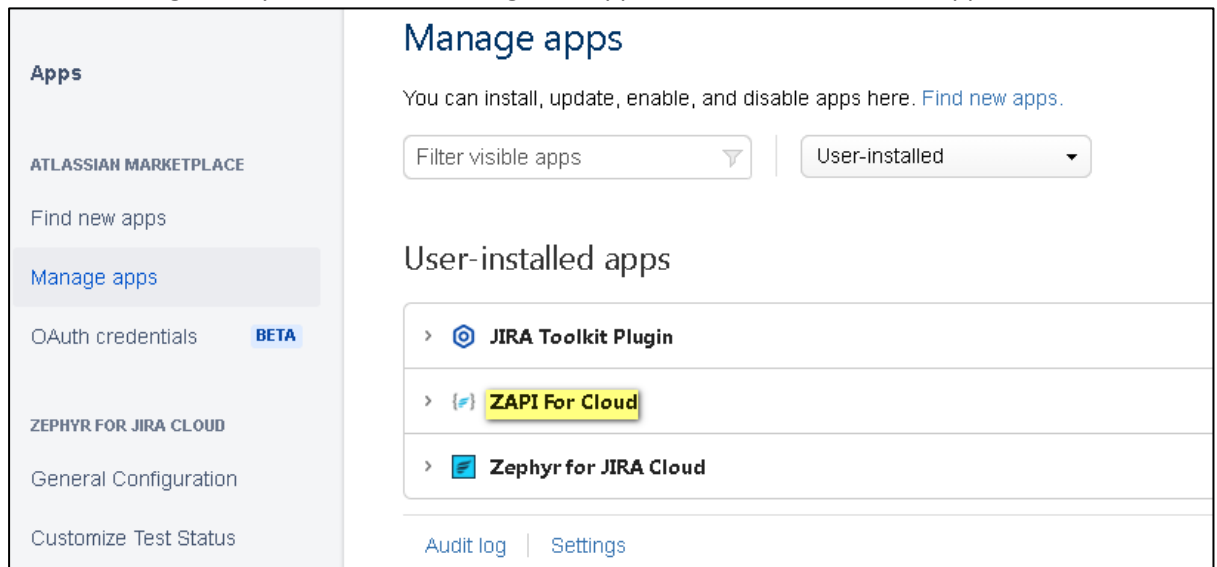
3. After installing the app goto 'Manage all apps' and navigate to 'Find new apps' from the left hand menu



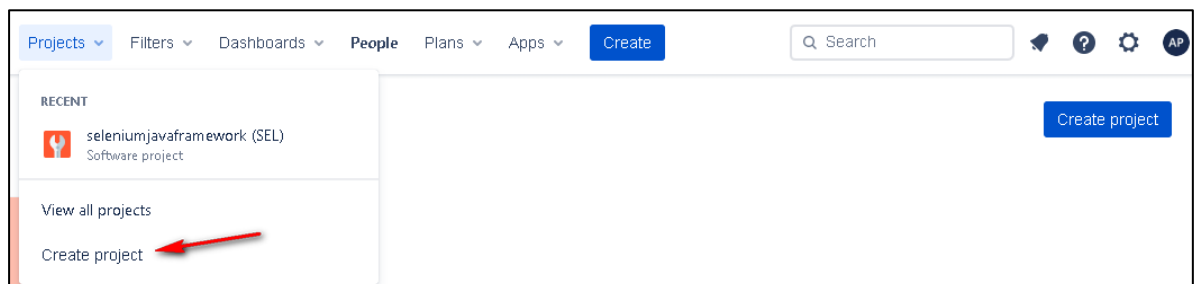
4. Now search for 'ZAPI' to integrate the application



5. After integration you can find the integrated application in 'User-installed apps' as below



- Create project:
 1. Navigate to Zephyr application and click on 'Projects' from the header to create new project



Create project

Name

seleniumjavaframework

Key

SELENIUM



☐ Share settings with an existing project

Template



Kanban

Monitor work in a continuous flow for agile teams • Suits teams who control work volume from a backlog

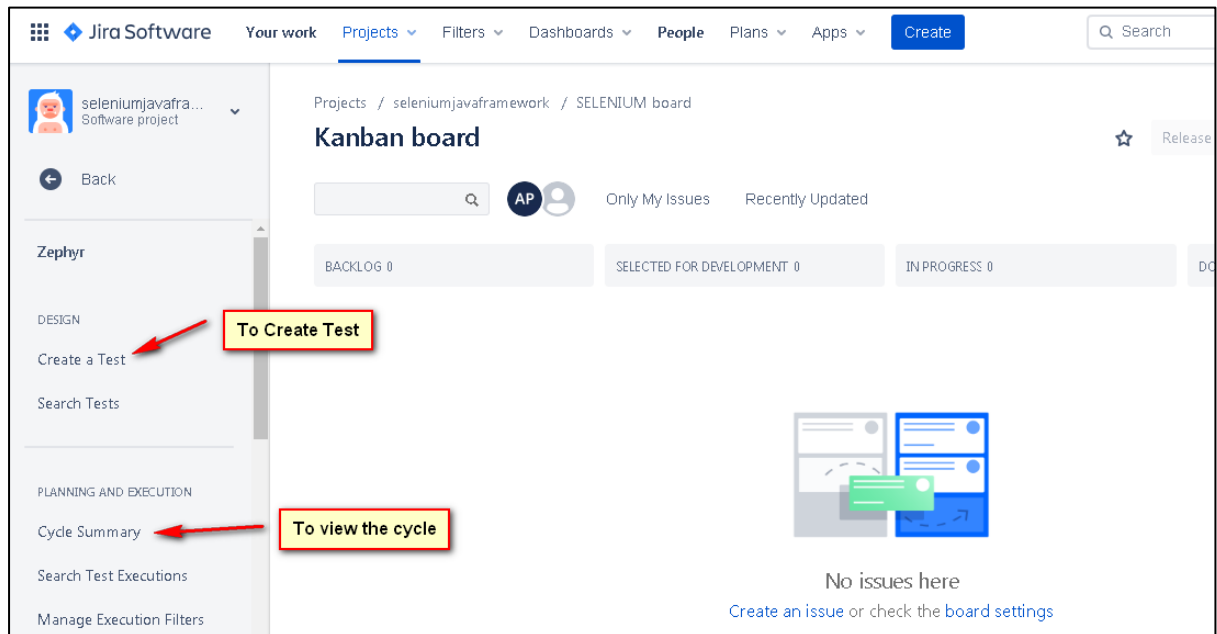
Change template

Create



- Create Test:

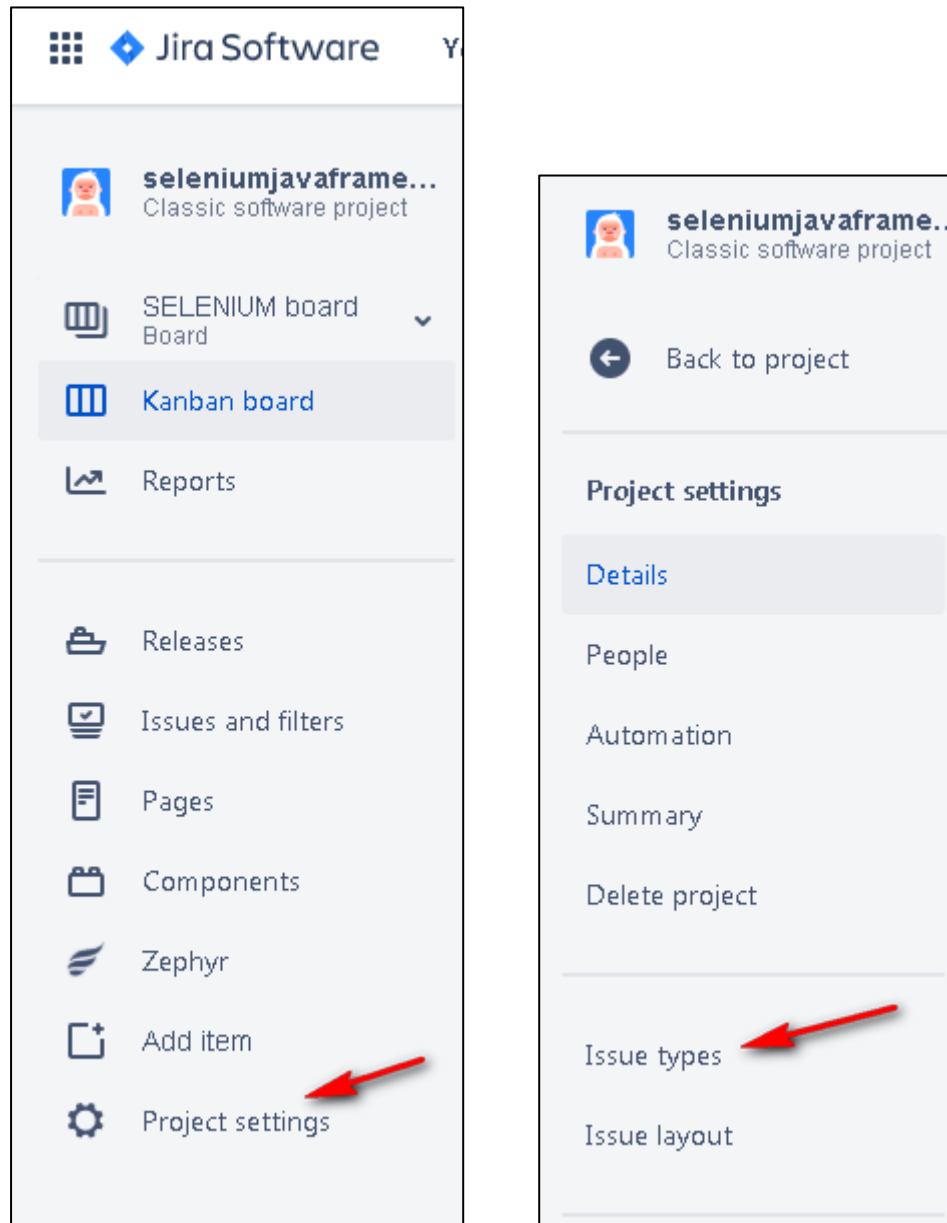
1. Navigate to 'Zephyr' from the left hand menu and click on 'Crate a Test' to Create Test in corresponding project.



The screenshot shows the 'Create issue' form in Jira. The form has fields for 'Project' (set to 'seleniumjavaframework'), 'Issue Type' (set to 'Test'), and 'Summary' (set to 'tc001_addItemToCartAndPurchaseItem'). There are also fields for 'Components' (set to 'None'), 'Attachment' (with a 'Drop files to attach, or browse.' button), and 'Description'. At the bottom right, there is a 'Create' button highlighted with a red arrow, and a 'Cancel' button next to it. A checkbox for 'Create another' is also visible.

If you didn't find issue type 'Test', please follow the procedure to edit the issue type scheme -

Navigate to Project / Project Settings / Issue Types and in 'Actions' dropdown select "Edit issue types"




Projects / seleniumjavaframework / Project settings

Issue types

SELENIUM: Kanban Issue Type Scheme









Keep track of different types of issues, such as bugs or tasks. Each issue type can be configured differently.

The issue type scheme defines which issue types apply to this project. To change the issue types used, you can select a different issue type scheme or use a different scheme.



Actions

- Edit issue types
- Use a different scheme

Issue Type	Description	Workflow	Field configuration	Screen
 Story	Functionality or a feature expressed as a user goal.	 Software Simplified Workflow for Project SELENIUM	 Default Field Configuration	 SELENIUM: Kanban Default Screen Scheme
 Bug	A problem or error.	 Software Simplified Workflow for Project SELENIUM	 Default Field Configuration	 SELENIUM: Kanban Bug Screen Scheme


Now in the 'Issue Types of Current Scheme', drag and drop the "Test" issue type from 'Available issue types' and save.

Now you can create a test with issue type 'Test' as mentioned above.

- Create Test Cycle and add tests:
 1. Click on 'Cycle Summary' from left hand menu to navigate to Cycle Summary and click on 'Create New Test Cycle' to create a Test Cycle




Projects / seleniumjavaframework / Cycle Summary

Cycle Summary



Create New Test Cycle

UNRELEASED

- UNSCHEDULED
 
-  Ad hoc
 
- RELEASED

Ad hoc

Build	:	Total Executions	: 0
Environment	:	Cycle Executions	: 0
Created By	:	Total Executed	: 0
Total Execution Time	:	Total Logged Time	: 0m

☐ Select All

ID	Status	Summary

Create New Test Cycle

Version : UNSCHEDULED

Name * : OrangeHRMRuns

Maximum 210 characters are allowed

Description :

Build :

Environment :

From :

To :

[Save](#) [Cancel](#)

2. Click on '...' dots on created test cycle to add tests (Created in Step 3) to the cycle (Created at Step 4)

Projects / seleniumjavaframework / Cycle Summary

Cycle Summary

Search

Create New Test Cycle

UNRELEASED

UNCHEDULED

OrangeHRMRuns

Ad hoc

RELEASED

...

Add Tests

Add New Folder

Edit

Move

Clone

Export

Delete

OrangeHRMRuns

Build	:	Total Executions	: 0
Environment	:	Cycle Executions	: 0
Created By	:	Total Executed	: 0
Total Execution Time	:	Total Logged Time	: 0m

☐ Select All

Status	Summary	De
--------	---------	----


Add Tests to Cycle: OrangeHRMRuns

Individually Via Search Filter From Another Cycle

Test :

Assign To

HISTORY SEARCH

 SELENIUM-1 - tc001_addItemToCartAndPurc...

CURRENT SEARCH

- Get TestID, TestCycleName:

1. Save 'Test Cycle Name', 'Test ID' at temporary file.

Projects / seleniumjavaframework / Cycle Summary

Cycle Summary

OrangeHRMRuns TestCycleName

seleniumjavaframework / Unscheduled / OrangeHRMRuns / SELENIUM-1

SELENIUM-1 TestID

Test Execution

Execution Status : UNEXECUTED Assigned To : Choose Value...

Defects : Select... OR Create New Issue

Comment :

Summary

tc001_addItemToCartAndPurchaseItem

Description

Date and time

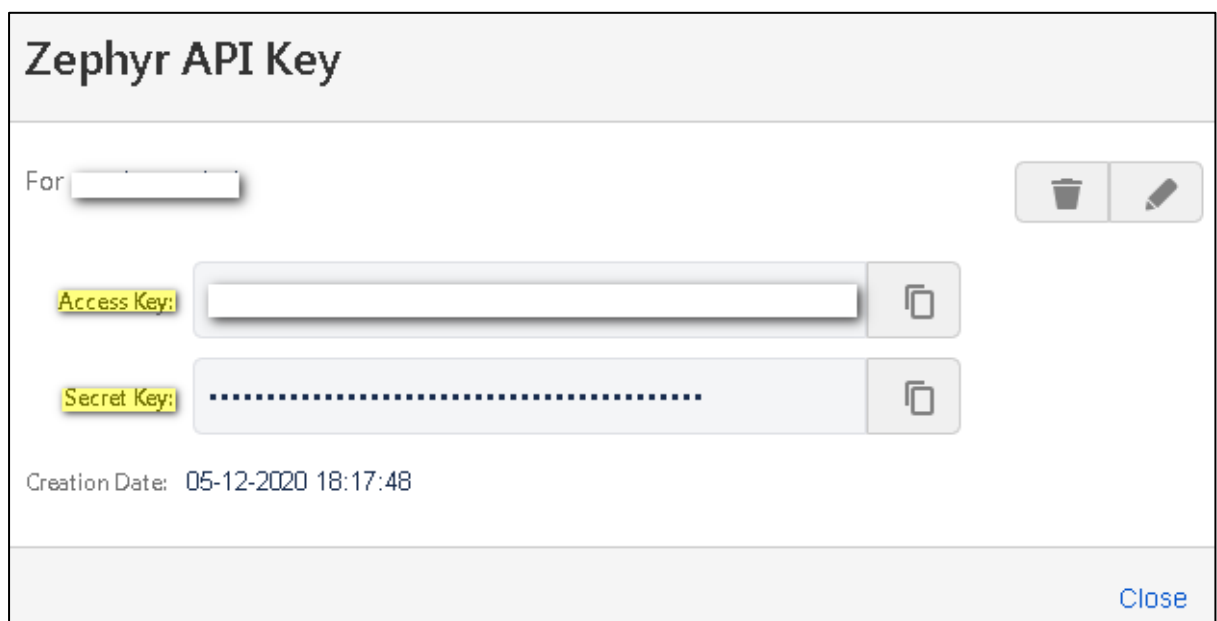
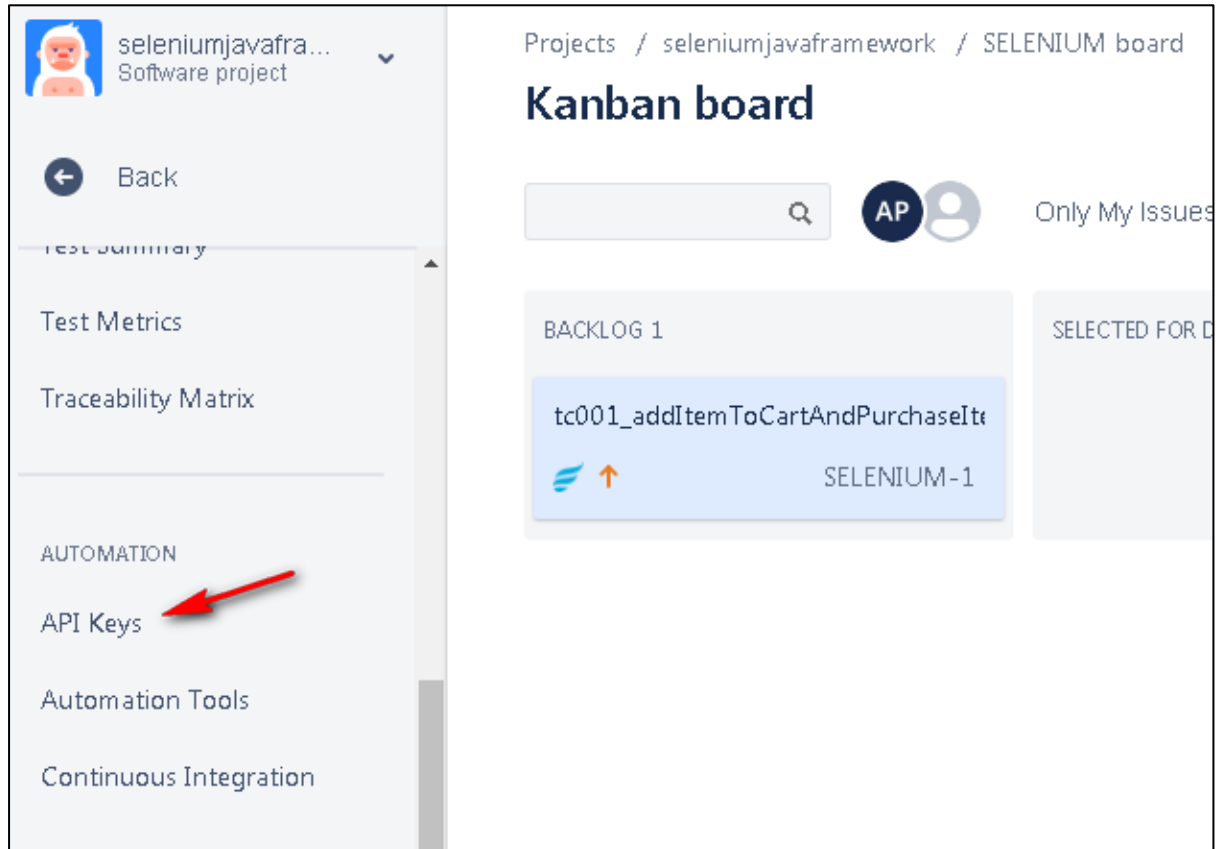
Execution History

No Data to display

3. Generate required Zephyr authorization values

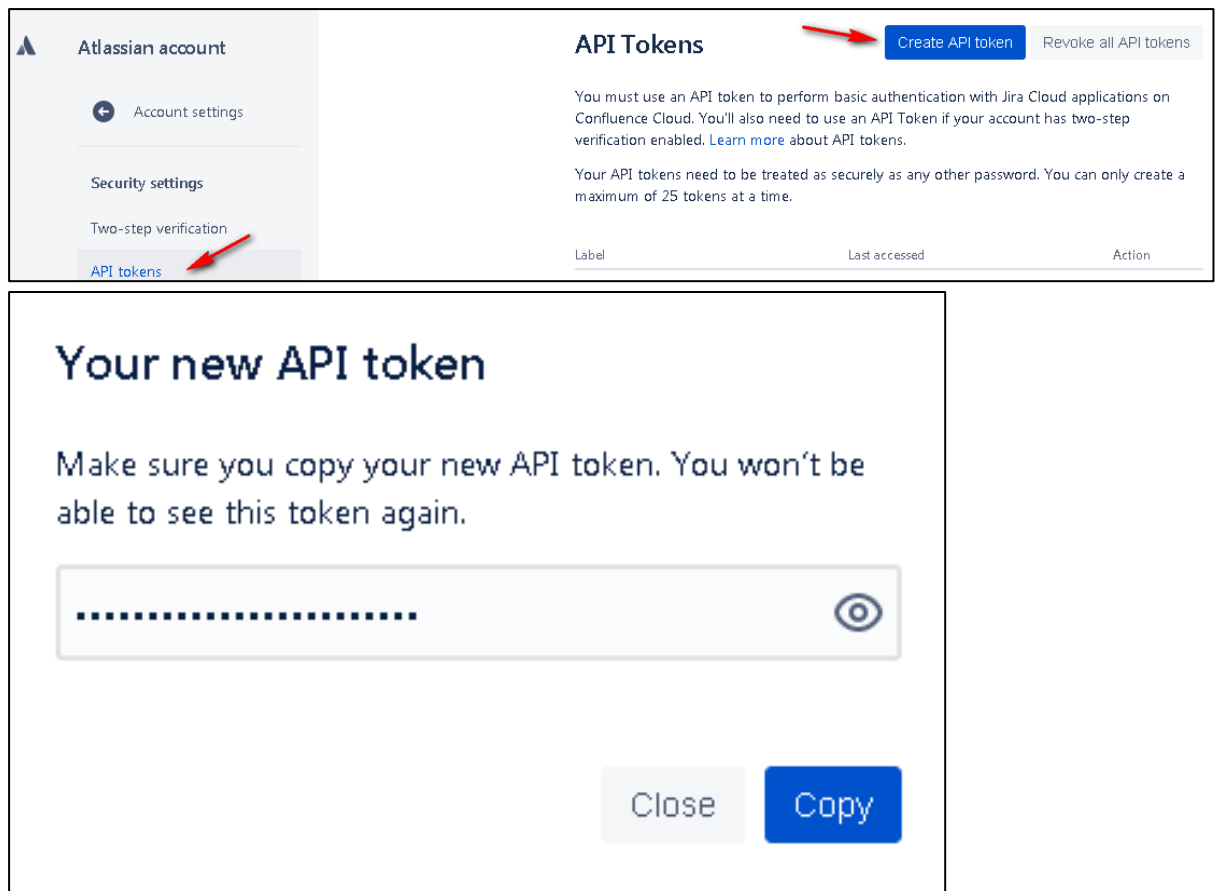
- Generate API Keys:

1. Navigate to 'API Keys' in the left hand menu and click on 'Generate API Key'



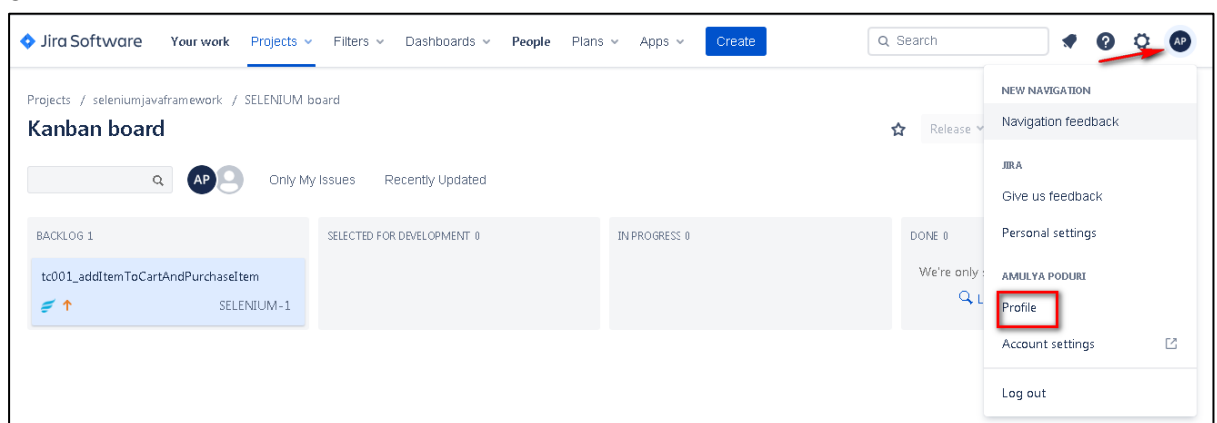
Note: Please save Access Key , Secret key values in temporary file for further use.

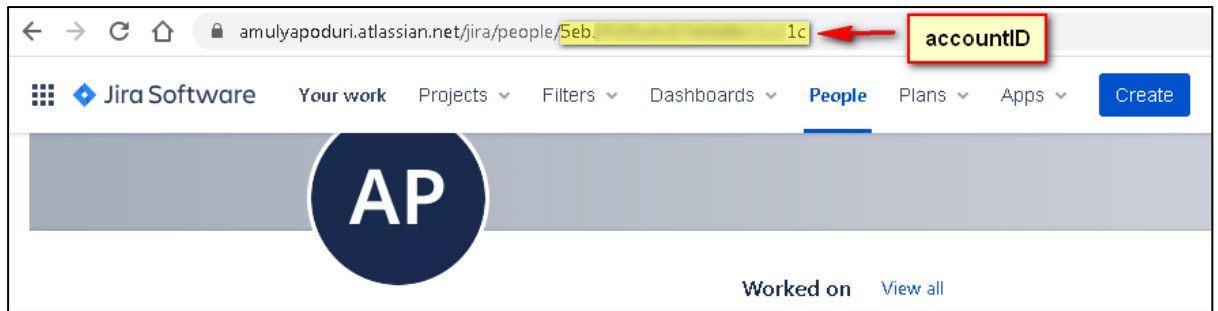
2. Generate API token by navigating to Atlassian account / Security / API Token



Note: Please save API token in a temporary file for further use.

3. Generate Account ID – Click on User Icon / Profile and get the account ID from the end of URL



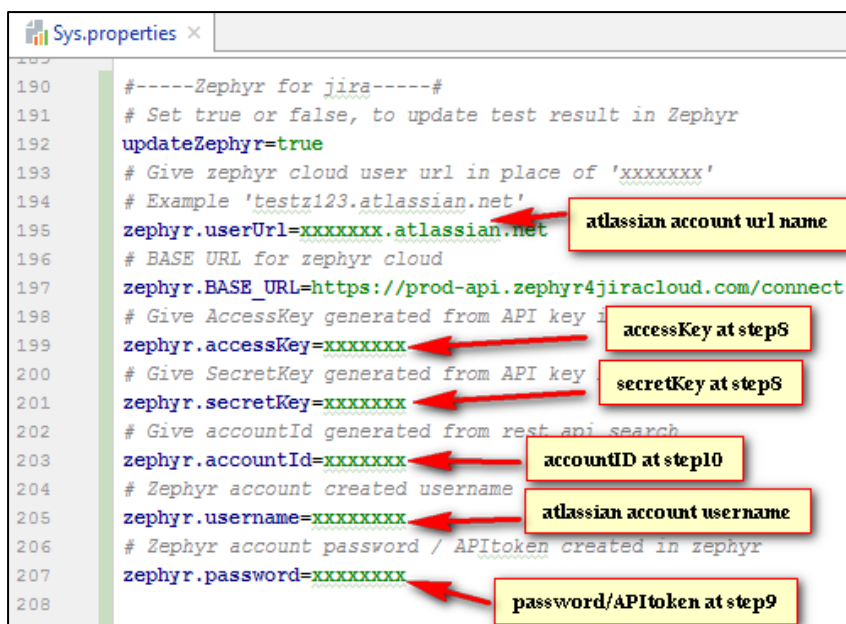


Note: Please save the accountID in temporary file for further use.

4. Changes needed in Sys.properties file

Set appropriate values for below properties in '/ConfigFiles/Sys.properties' file of project

- Include Zephyr properties before executing a test. Fill all the zephyr properties (saved above)



Note: Please ensure that 'updateZephyr' value is "true" before starting execution to update Zephyr results

5. Changes needed in test suite

Add Zephyr test case id and test cycle name as the parameters for annotation '@ZephyrTestID' above the selenium test in test suite

- Add TestID , TestCycleName (Saved in post Step 6) before test as annotation parameters.

```
@ZephyrTestID(TestID = "SELENIUM-1", TestCycleName = "OrangeHRMRuns")
@Test(groups = "regression")
public void tc001_addItemToCartAndPurchaseItem() {

    // code for test case

}
```

6. Execution and Result review

- After setting up all the above pre-requisites, generating required authorization values and enabling 'updatingZephyr' value as 'true' simply execute your test either directly or through batch file.
- After the execution of test completes the test status is automatically updated in the tool along with execution time and date as shown below

The screenshot displays the 'Cycle Summary' page in Zephyr. The breadcrumb trail is 'Projects / test project / Cycle Summary'. The page title is 'Cycle Summary'. On the left, there's a sidebar with 'Test Cycles and Folders' containing 'OrangeHRMRuns' and 'TP-3'. The main content area shows 'Test Execution' with 'Execution Status' as 'PASS' (highlighted with a red arrow). Below it are fields for 'Defects' (a dropdown menu) and 'Comment'. To the right, the 'Summary' section shows 'tc001_additemToCartAndPurchaseItem'. The 'Date and time' section shows 'Executed By: amulya poduri' and 'Executed On: 05-13-2020 09:49:10' (highlighted with a red box).

7. Code for customization of test rail API methods and integration of test results to test rail

This section provides information about location of Zephyr's ZAPI code, implementation of customized methods using ZAPI and integration code for updating test results into Zephyr in selenium java framework.

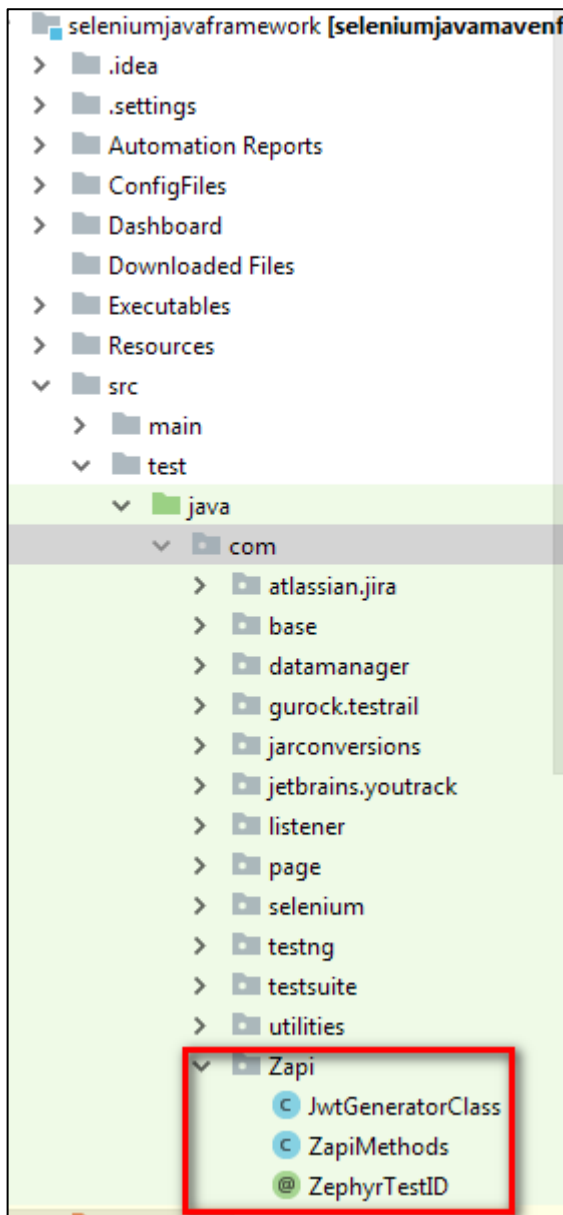
7.1 Required ZfjCloudRestClient jar dependency added to pom.xml

To start with code customization please ensure that 'ZfjCloudRestClient' jar dependency is added in "pom.xml".

```
<dependency>
  <groupId>org.ZfjCloudRestClient</groupId>
  <artifactId>ZfjCloudRestClient</artifactId>
  <version>1.0</version>
  <scope>system</scope>
  <systemPath>${pom.basedir}/Resources/lib/ZfjCloudRestClient.jar</systemPath>
</dependency>
```


7.2 Implementation code for customizing ZAPI methods

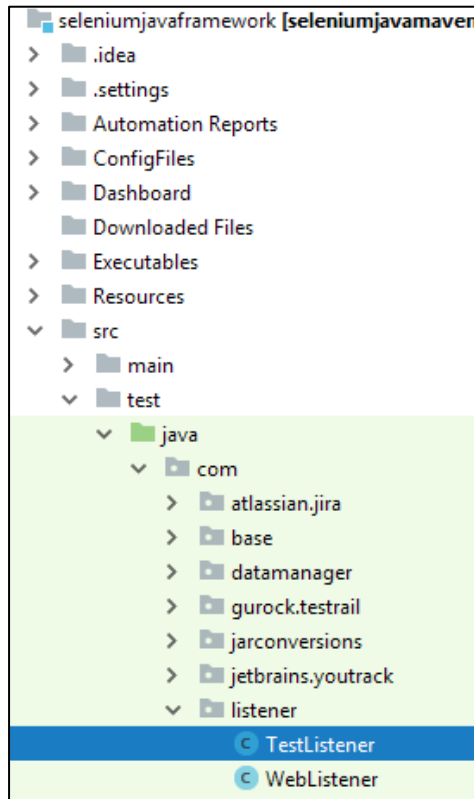
ZAPI code exists inside package '**com.Zapi**' and consists of files shown in below screenshot.



1. **JwtGeneratorClass.java :**
This file contains code to generate the JWT token and returns it using Zephyr API url, AccessKey, SecretKey, AccountID values mentioned in 'Sys.properties' file. This token is used in every basic authenticate API requests.
2. **ZapiMethods.java :**
This file contains code of ZAPI binding for Java. It provides the basic functionality to authenticate rest API requests like 'GET', 'PUT' ...etc., It provides response which can be converted to JSON for encoding/decoding and has generic support.
3. **ZephyrTestID.interface :**
This file is to define user specific annotations like '@ZephyrTestID' with specified parameter types and variables.

7.3 Integration code for updating test results to Zephyr

Integration code for updating test results to zephyr is implemented in '**TestListener.java**' file. Please see location of file is shown in below screenshot.



1. OnTestStart:
As the test starts, the "projectID", "issueID" will be returned and saved based on the 'testID' through ZAPI calls
2. OnTestSuccess / OnTestFailure :
After the test completes, based on the result (pass or fail) respective listener is called and "executionID" will be returned for specified 'testID', 'testCycleName', 'projectID', 'issueID' (saved at test start) through ZAPI calls. And the the test result is updated as PASS / FAIL (based on result) in Zephyr cloud tool in the respective Test Cycle Summary.