Integration with CircleCI

CircleCI is a well-known CI/CD tool available on-premises and as SaaS.

Xray does not provide yet a plugin for CircleCI. However, it is easy to setup CircleCI in order to integrate it with Xray.

Since Xray provides a full REST API, you may interact with Xray, for submitting results for example.

JUnit example

In this scenario, we want to get visibility of the automated test results from some tests implemented in Java, using the JUnit framework.

This recipe could also be applied for other frameworks such as NUnit, TestNG or Robot.

We need to setup a project based on a Git repository containing the code along with the configuration for CircleCI build process.

The tests are implemented in a JUnit class as follows.

CalcTest.java

```
package com.xpand.java;

import org.junit.After;
import org.junit.Before;
import org.junit.Test;

import static org.hamcrest.CoreMatchers.is;
import static org.junit.Assert.assertThat;

public class CalcTest {

    @Before
    public void setUp() throws Exception {

    }

    @After
    public void tearDown() throws Exception {

    }

        @Test
    public void CanAddNumbers()
    {
        assertThat(Calculator.Add(1, 1), is(2));
        assertThat(Calculator.Add(-1, 1), is(0));
    }


    @Test
    public void CanSubtract()
    {
        assertThat(Calculator.Subtract(1, 1), is(0));
        assertThat(Calculator.Subtract(-1, -1), is(0));
        assertThat(Calculator.Subtract(100, 5), is(95));
    }


    @Test
    public void CanMultiply()
    {
        assertThat(Calculator.Multiply(1, 1), is(1));
        assertThat(Calculator.Multiply(-1, -1), is(1));
        assertThat(Calculator.Multiply(100, 5), is(500));
    }


    public void CanDivide()
    {
        assertThat(Calculator.Divide(1, 1), is(1));
        assertThat(Calculator.Divide(-1, -1), is(1));
        assertThat(Calculator.Divide(100, 5), is(20));
    }


    @Test
    public void CanDoStuff()
    {
        assertThat(true, is(true));
    }


}
```

The CircleCI configuration file `.circleci/config.yml` contains the definition of the build steps, including running the automated tests and submitting the results.

.circleci/config.yml

```
version: 2 # use CircleCI 2.0
jobs: # a collection of steps
  build: # runs not using Workflows must have a `build` job as entry point

    working_directory: ~/demo/java-junit-calc # directory where steps will run

    docker: # run the steps with Docker
      - image: circleci/openjdk:8-jdk-browsers # ...with this image as the primary container; this is where all
`steps` will run

    steps: # a collection of executable commands

      - checkout: # check out source code to working directory
          path: ~/demo

      - restore_cache: # restore the saved cache after the first run or if `pom.xml` has changed
          key: circleci-java-junit-calc-demo # circleci-java-junit-calc-demo-{{ checksum "pom.xml" }}

      - run: mvn dependency:go-offline # gets the project dependencies

      - run: mvn test # run the actual tests

      - save_cache: # saves the project dependencies
          paths:
            - ~/.m2
          key: circleci-java-junit-calc-demo # circleci-java-junit-calc-demo-{{ checksum "pom.xml" }}

      - store_test_results: # uploads the test metadata from the `target/surefire-reports` directory so that it
can show up in the CircleCI dashboard.
          path: target/surefire-reports

      - run: 'curl -H "Content-Type: multipart/form-data" -u $jira_user:$jira_password -F "file=@target
/surefire-reports/TEST-com.xpand.java.CalcTest.xml" "$jira_server_url/rest/raven/1.0/import/execution/junit?
projectKey=CALC"'
```

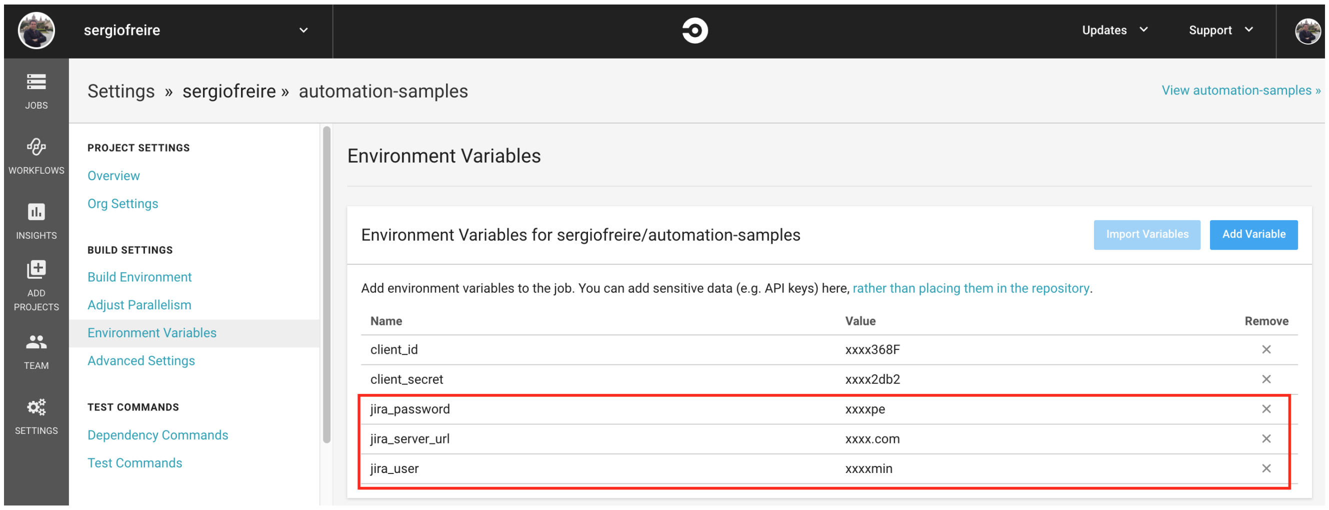
In order to submit those results, we'll just need to invoke the REST API (as detailed in Import Execution Results - REST).

However, we do not want to have the Xray credentials hardcoded in CircleCI's configuration file. Therefore, we'll use some environment variables defined in project settings, including:

- **jira_user**: for the Jira username
- **jira_password**: for the Jira user's password
- **jira_server_url**: for the Jira's base URL (e.g. http://yourjiraserver)

Please note

The user present in the configuration below must exist in the JIRA instance and have permission to Create Test and Test Execution Issues.



Environment Variables

Environment Variables for sergiofreire/automation-samples

Add environment variables to the job. You can add sensitive data (e.g. API keys) here, rather than placing them in the repository.

Name	Value	Remove
client_id	xxxx368F	×
client_secret	xxxx2db2	×
jira_password	xxxxpe	×
jira_server_url	xxxx.com	×
jira_user	xxxxmin	×

In `.circleci/config.yml` a "step" must be included that will use "curl" in order to submit the results to the REST API.

```
curl -H "Content-Type: multipart/form-data" -u $jira_user:$jira_password -F "file=@target/surefire-reports/TEST-
com.xpand.java.CalcTest.xml" "$jira_server_url/rest/raven/1.0/import/execution/junit?projectKey=CALC"
```

We're using "curl" utility that comes in Unix based OS'es but you can easily use another tool to make the HTTP request; however, "curl" is provided in the container used by CircleCI.

Robot Framework example

In this scenario, we want to get visibility of the automated test results from some UI tests implemented in Robot Framework (Python) together with Selenium (using the "robotframework-seleniumlibrary"), and using Chrome for testing.

We need to set up a Git repository containing the code along with the configuration for CircleCI build process.

The tests are implemented in Robot Framework .robot files as follows.

valid_login.robot

```
*** Settings ***
Documentation     A test suite with a single test for valid login.
...
...               This test has a workflow that is created using keywords in
...               the imported resource file.
Resource          resource.robot

*** Test Cases ***
Valid Login
    [Tags]  UI
    Open Browser To Login Page
    Input Username    demo
    Input Password    mode
    Submit Credentials
    Welcome Page Should Be Open
    [Teardown]    Close Browser
```

The CircleCI configuration file `.circleci/config.yml` contains the definition of the build steps, including running the automated tests and submitting the results.

.circleci/config.yml

```
# Use the latest 2.1 version of CircleCI pipeline process engine.
# See: https://circleci.com/docs/configuration-reference

# For a detailed guide to building and testing with Python, read the docs:
# https://circleci.com/docs/language-python/ for more details
version: 2.1

# Orbs are reusable packages of CircleCI configuration that you may share across projects, enabling you to
create encapsulated, parameterized commands, jobs, and executors that can be used across multiple projects.
# See: https://circleci.com/docs/orb-intro/
orbs:
  # See the Python orb documentation here: https://circleci.com/developer/orbs/orb/circleci/python
  python: circleci/python@2.1.1
  browser-tools: circleci/browser-tools@1.4.6

# Define a job to be invoked later in a workflow.
# See: https://circleci.com/docs/jobs-steps/#jobs-overview & https://circleci.com/docs/configuration-reference
/#jobs
jobs:
  build-and-test:
    # Specify the execution environment. You can specify an image from Docker Hub or use one of our convenience
images from CircleCI's Developer Hub.
    # See: https://circleci.com/docs/executor-intro/ & https://circleci.com/docs/configuration-reference
/#executor-job
    docker:
      # Specify the version you desire here
      # See:https://circleci.com/developer/images/image/cimg/python
      - image: cimg/python:3.12-browsers

    # Add steps to the job
    # See: https://circleci.com/docs/jobs-steps/#steps-overview & https://circleci.com/docs/configuration-
reference/#steps
    steps:
      # Checkout the code as the first step.
      - checkout
      - python/install-packages:
          pkg-manager: pip
          # app-dir: ~/project/package-directory/  # If your requirements.txt isn't in the root directory.
          # pip-dependency-file: test-requirements.txt  # if you have a different name for your requirements
file, maybe one that combines your runtime and test requirements.
      # get server up and running in the background
      - run:
          name: Run webserver to be target by tests
          command: python demoapp/server.py
          background: true
      - run:
          name: Run tests
          # This assumes Robot Framework is installed via the install-package step above
          command: robot -x junit.xml -o output.xml login_tests || true
      - run:
          name: Upload results to Xray DC
          command: |
            echo uploading RF output.xml, if available, to Xray...
            [ -f output.xml ] && curl -H "Content-Type: multipart/form-data" -u $XRAY_USERNAME:$XRAY_PASSWORD -
F "file=@output.xml" "$XRAY_SERVER_URL/rest/raven/2.0/import/execution/robot?projectKey=$PROJECT_KEY"
      - store_test_results:
          path: junit.xml
          when: always

# Orchestrate jobs using workflows
# See: https://circleci.com/docs/workflows/ & https://circleci.com/docs/configuration-reference/#workflows
workflows:
  sample: # This is the name of the workflow, feel free to change it to better match your workflow.
    # Inside the workflow, you define the jobs you want to run.
    jobs:
      - build-and-test
```

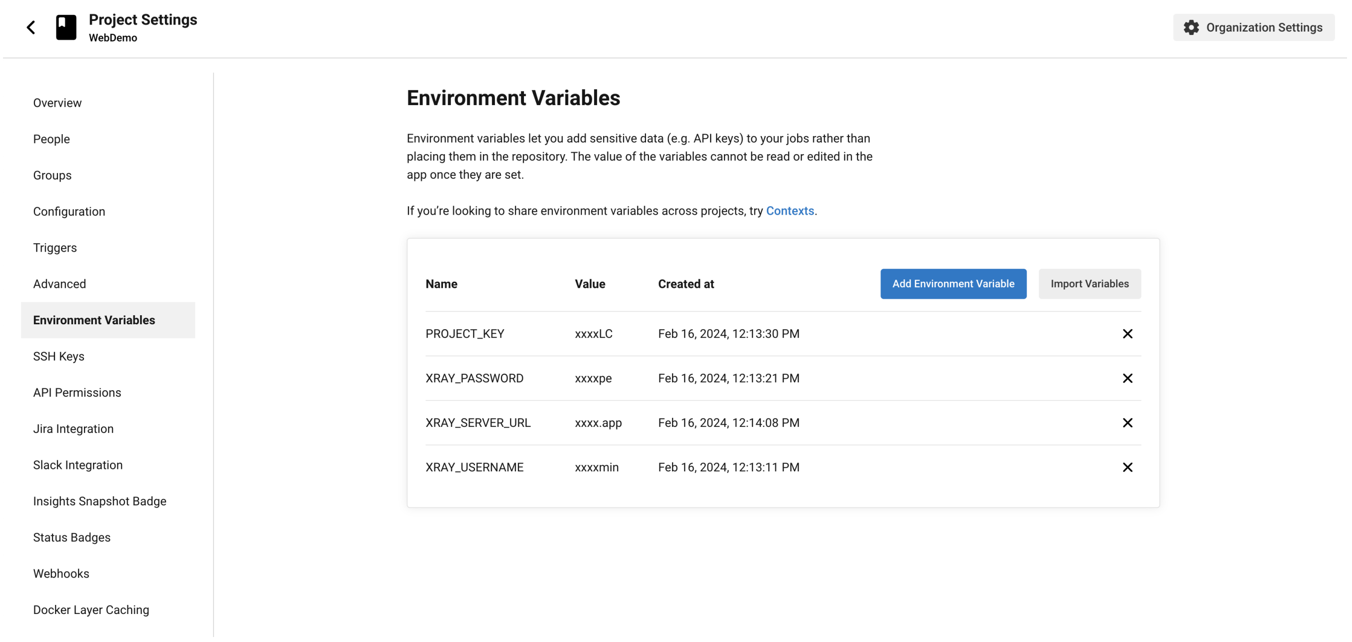
In order to submit those results, we'll just need to invoke the REST API (as detailed in Import Execution Results - REST).

However, we do not want to have the Xray API credentials hardcoded in the CircleCI's configuration file. Therefore, we'll use environment variables defined in the project settings, including:

- **XRAY_SERVER_URL**: Jira's base URL
- **XRAY_USERNAME**: the username used in the REST API
- **XRAY_PASSWORD**: the password used in the REST API
- **PROJECT_KEY**: Jira project

Please note

The user associated with the credentials must have permissions to Create Test and Test Execution Issues.



Environment Variables

Environment variables let you add sensitive data (e.g. API keys) to your jobs rather than placing them in the repository. The value of the variables cannot be read or edited in the app once they are set.

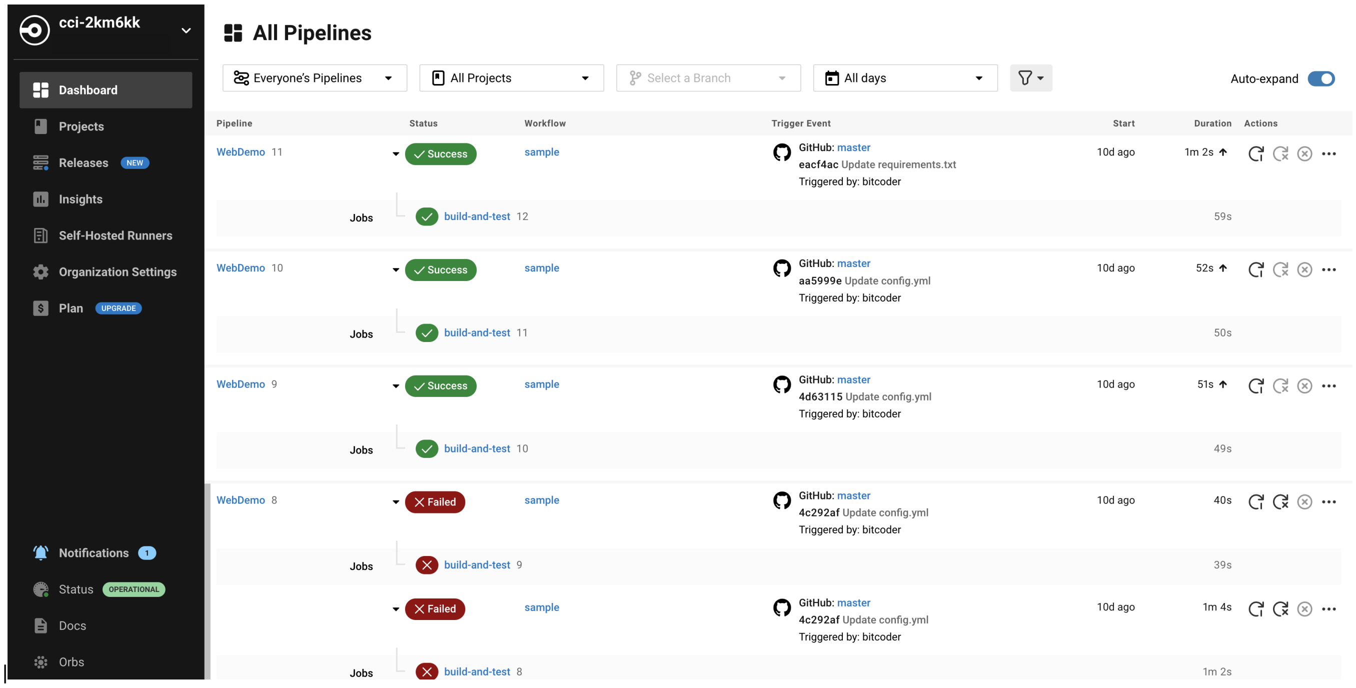
If you're looking to share environment variables across projects, try Contexts.

Name	Value	Created at
PROJECT_KEY	xxxxLC	Feb 16, 2024, 12:13:30 PM
XRAY_PASSWORD	xxxxpe	Feb 16, 2024, 12:13:21 PM
XRAY_SERVER_URL	xxxx.app	Feb 16, 2024, 12:14:08 PM
XRAY_USERNAME	xxxxmin	Feb 16, 2024, 12:13:11 PM

In `.circleci/config.yml` a "step" must be included that will use "curl" in order to submit the results to the REST API, using the Xray/Jira credentials.

```
curl -H "Content-Type: multipart/form-data" -u $XRAY_USERNAME:$XRAY_PASSWORD -F "file=@output.xml"
"$XRAY_SERVER_URL/rest/raven/2.0/import/execution/robot?projectKey=$PROJECT_KEY"
```

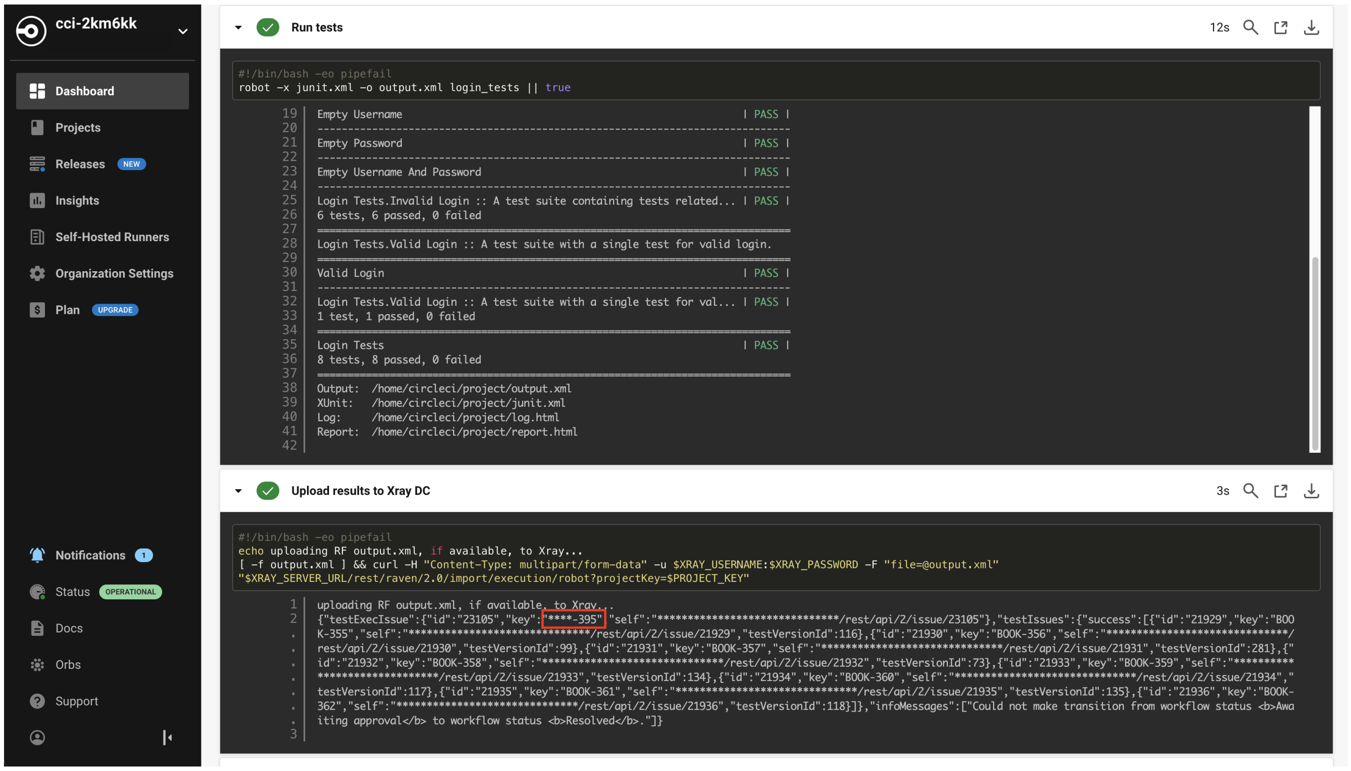
We're using "curl" utility that comes in Unix based OS'es but you can easily use another tool to make the HTTP request; however, "curl" is provided in the container used by CircleCI.



All Pipelines

Everyone's Pipelines | All Projects | Select a Branch | All days | Auto-expand

Pipeline	Status	Workflow	Trigger Event	Start	Duration	Actions
WebDemo 11	Success	sample	GitHub: master eacf4ac Update requirements.txt Triggered by: bitcoder	10d ago	1m 2s	
Jobs	✓	build-and-test 12			59s	
WebDemo 10	Success	sample	GitHub: master aa5999e Update config.yml Triggered by: bitcoder	10d ago	52s	
Jobs	✓	build-and-test 11			50s	
WebDemo 9	Success	sample	GitHub: master 4d63115 Update config.yml Triggered by: bitcoder	10d ago	51s	
Jobs	✓	build-and-test 10			49s	
WebDemo 8	Failed	sample	GitHub: master 4c292af Update config.yml Triggered by: bitcoder	10d ago	40s	
Jobs	✗	build-and-test 9			39s	
	Failed	sample	GitHub: master 4c292af Update config.yml Triggered by: bitcoder	10d ago	1m 4s	
Jobs	✗	build-and-test 8			1m 2s	



Run tests 12s

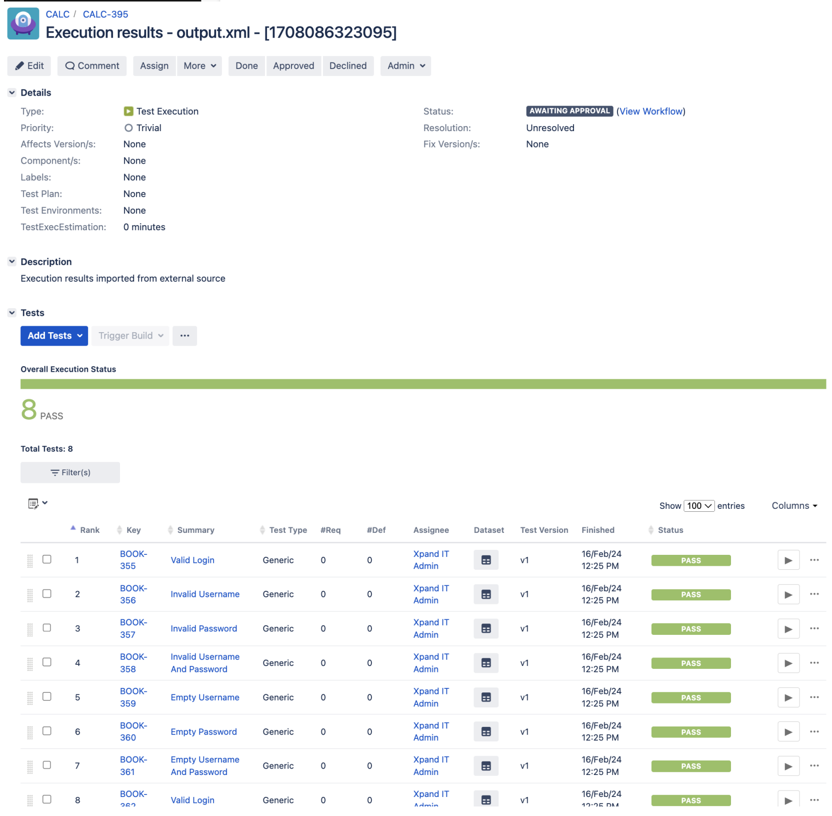
```
#!/bin/bash -eo pipefail
robot -x junit.xml -o output.xml login_tests || true
```

```
Empty Username                                                        | PASS |
------------------------------------------------------------------------------
Empty Password                                                        | PASS |
------------------------------------------------------------------------------
Empty Username And Password                                           | PASS |
------------------------------------------------------------------------------
Login Tests.Invalid Login :: A test suite containing tests related... | PASS |
6 tests, 6 passed, 0 failed
==============================================================================
Login Tests.Valid Login :: A test suite with a single test for valid login.
==============================================================================
Valid Login                                                           | PASS |
------------------------------------------------------------------------------
Login Tests.Valid Login :: A test suite with a single test for val... | PASS |
1 test, 1 passed, 0 failed
==============================================================================
Login Tests                                                           | PASS |
8 tests, 8 passed, 0 failed
==============================================================================
Output:  /home/circleci/project/output.xml
XUnit:   /home/circleci/project/junit.xml
Log:     /home/circleci/project/log.html
Report:  /home/circleci/project/report.html
```

Upload results to Xray DC 3s

```
#!/bin/bash -eo pipefail
echo uploading RF output.xml, if available, to Xray...
[ -f output.xml ] && curl -H "Content-Type: multipart/form-data" -u $XRAY_USERNAME:$XRAY_PASSWORD -F "file=@output.xml"
"$XRAY_SERVER_URL/rest/raven/2.0/import/execution/robot?projectKey=$PROJECT_KEY"
```

```
uploading RF output.xml, if available, to Xray...
{"testExecIssue":{"id":"23105","key":"****-395","self":"****************************/rest/api/2/issue/23105"},"testIssues":{"success":[{"id":"21929","key":"BOO
K-355","self":"****************************/rest/api/2/issue/21929","testVersionId":116},{"id":"21930","key":"BOOK-356","self":"****************************/
rest/api/2/issue/21930","testVersionId":99},{"id":"21931","key":"BOOK-357","self":"****************************/rest/api/2/issue/21931","testVersionId":281},{"
id":"21932","key":"BOOK-358","self":"****************************/rest/api/2/issue/21932","testVersionId":73},{"id":"21933","key":"BOOK-359","self":"**********
******************/rest/api/2/issue/21933","testVersionId":134},{"id":"21934","key":"BOOK-360","self":"****************************/rest/api/2/issue/21934","
testVersionId":117},{"id":"21935","key":"BOOK-361","self":"****************************/rest/api/2/issue/21935","testVersionId":135},{"id":"21936","key":"BOOK-
362","self":"****************************/rest/api/2/issue/21936","testVersionId":118}]},"infoMessages":["Could not make transition from workflow status <b>Awa
iting approval</b> to workflow status <b>Resolved</b>."]}
```



Execution results - output.xml - [1708086323095]

Details

Type:	Test Execution	Status:	AWAITING APPROVAL (View Workflow)
Priority:	Trivial	Resolution:	Unresolved
Affects Version/s:	None	Fix Version/s:	None
Component/s:	None		
Labels:	None		
Test Plan:	None		
Test Environments:	None		
TestExecEstimation:	0 minutes		

Description

Execution results imported from external source

Tests

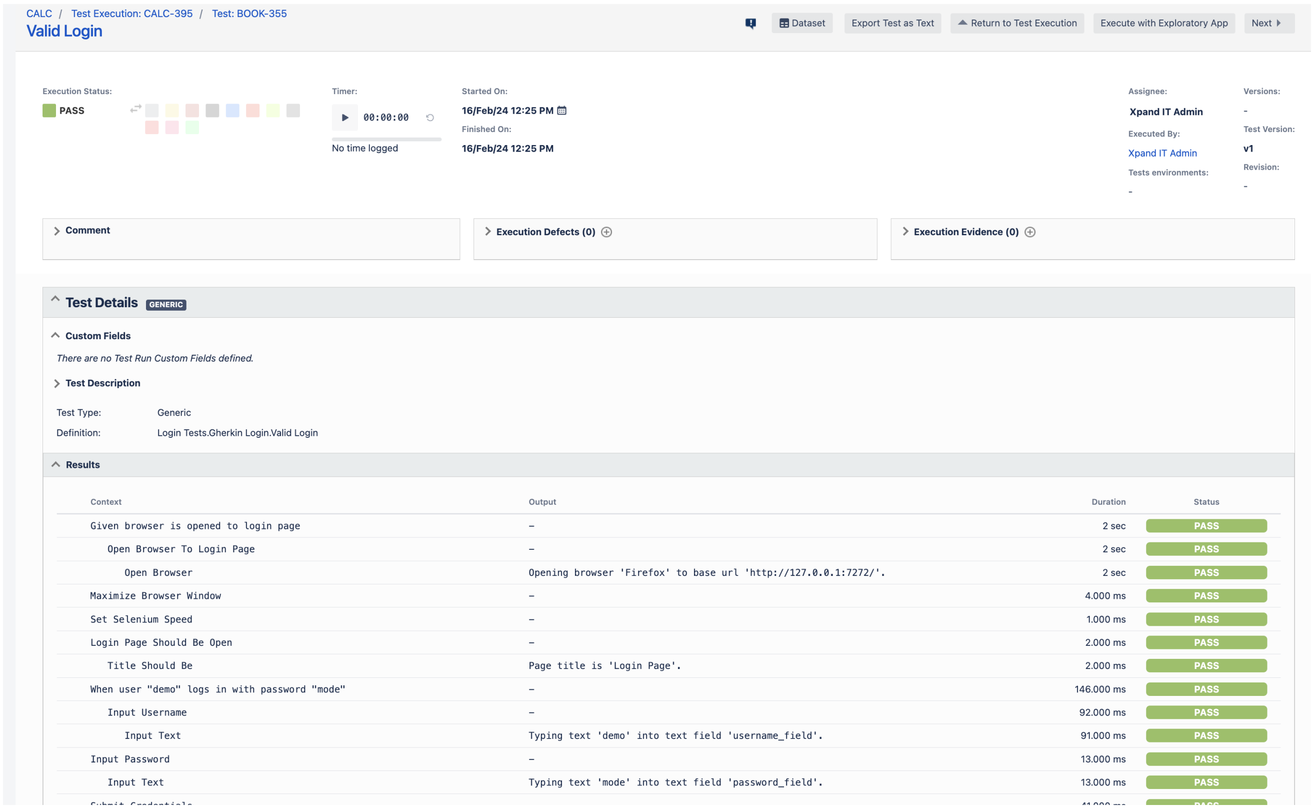
Overall Execution Status

8 PASS

Total Tests: 8

Show 100 entries

Rank	Key	Summary	Test Type	#Req	#Def	Assignee	Dataset	Test Version	Finished	Status
1	BOOK-355	Valid Login	Generic	0	0	Xpand IT Admin		v1	16/Feb/24 12:25 PM	PASS
2	BOOK-356	Invalid Username	Generic	0	0	Xpand IT Admin		v1	16/Feb/24 12:25 PM	PASS
3	BOOK-357	Invalid Password	Generic	0	0	Xpand IT Admin		v1	16/Feb/24 12:25 PM	PASS
4	BOOK-358	Invalid Username And Password	Generic	0	0	Xpand IT Admin		v1	16/Feb/24 12:25 PM	PASS
5	BOOK-359	Empty Username	Generic	0	0	Xpand IT Admin		v1	16/Feb/24 12:25 PM	PASS
6	BOOK-360	Empty Password	Generic	0	0	Xpand IT Admin		v1	16/Feb/24 12:25 PM	PASS
7	BOOK-361	Empty Username And Password	Generic	0	0	Xpand IT Admin		v1	16/Feb/24 12:25 PM	PASS
8	BOOK-	Valid Login	Generic	0	0	Xpand IT		v1	16/Feb/24	PASS


CALC / Test Execution: CALC-395 / Test: BOOK-355

Valid Login

Execution Status: PASS

Timer: 00:00:00

No time logged

Started On: 16/Feb/24 12:25 PM

Finished On: 16/Feb/24 12:25 PM

Assignee: Xpand IT Admin

Executed By: Xpand IT Admin

Tests environments: -

Versions: -

Test Version: v1

Revision: -

Comment

Execution Defects (0)

Execution Evidence (0)

Test Details GENERIC

Custom Fields

There are no Test Run Custom Fields defined.

Test Description

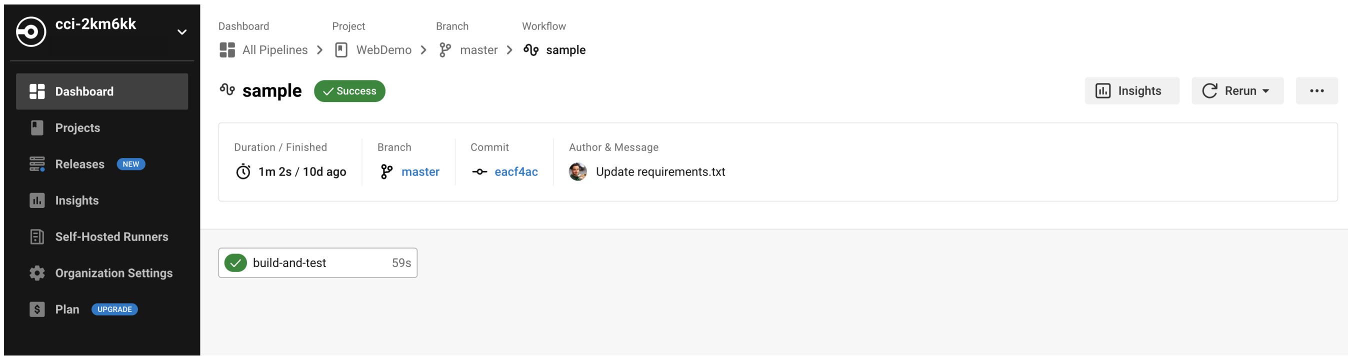
Test Type:	Generic
Definition:	Login Tests.Gherkin Login.Valid Login

Results

Context	Output	Duration	Status
`Given browser is opened to login page`	–	2 sec	PASS
`  Open Browser To Login Page`	–	2 sec	PASS
`    Open Browser`	`Opening browser 'Firefox' to base url 'http://127.0.0.1:7272/'.`	2 sec	PASS
`Maximize Browser Window`	–	4.000 ms	PASS
`Set Selenium Speed`	–	1.000 ms	PASS
`Login Page Should Be Open`	–	2.000 ms	PASS
`  Title Should Be`	`Page title is 'Login Page'.`	2.000 ms	PASS
`When user "demo" logs in with password "mode"`	–	146.000 ms	PASS
`  Input Username`	–	92.000 ms	PASS
`    Input Text`	`Typing text 'demo' into text field 'username_field'.`	91.000 ms	PASS
`Input Password`	–	13.000 ms	PASS
`  Input Text`	`Typing text 'mode' into text field 'password_field'.`	13.000 ms	PASS

References

- https://circleci.com/docs/2.0/configuration-reference/



Dashboard: All Pipelines > Project: WebDemo > Branch: master > Workflow: sample

sample Success

Duration / Finished	Branch	Commit	Author & Message
1m 2s / 10d ago	master	eacf4ac	Update requirements.txt

build-and-test 59s