Test Automation Project for “OpenProject” Software

Installation of OpenProject using Docker

To install OpenProject locally on your machine, please follow the instructions here: https://docs.openproject.org/installation-and-operations/installation/docker/

Note that for this installation you need to have Docker Desktop installed on your Windows machine: https://www.docker.com/products/docker-desktop

API Automation

You may find the official OpenProject API documentation here: https://docs.openproject.org/api/

Tools and frameworks to be used for automation development

1. Use Python’s “Requests” library to implement an HTTP client https://requests.readthedocs.io/en/master/
2. Use the pytest framework to implement individual test cases https://docs.pytest.org/en/stable/
3. Postman - for manual experimentation with the API
4. Chrome Developer Tools “Network” tab - to investigate the API calls issued from the web UI.

Authentication

In order to interact with the server over REST API, we must use one of the supported authentication methods: https://docs.openproject.org/api/introduction/
For this exercise we’ll be using the “Basic Auth” method. Carefully read and follow the instructions for setting up Basic Auth here: https://docs.openproject.org/api/example/

Once you’ve created an access token for basic authentication, use Postman to create a new GET request and set it to use “Basic Auth”. Set request URL to:
http://<server_ip>:<port_number>/api/v3/work_packages/
Send the request and make sure that you get a proper response with status 200.

General Guidelines for API Tests Automation

1. Learn each required API endpoint by reading the official documentation **and** by investigating the API calls issued by the browser when executing the same operation from the UI. (Use Chrome Developer Tools “Network” tab).
2. Before writing any code, use Postman to experiment with the various API calls manually. Make sure that you are able to construct a proper request and that you receive a successful response.
3. Following the execution of different API calls, you should manually validate that you can see the expected result in the UI. For example: After using the API to create a new project named “XYZ Project”, login to the UI and verify that indeed a new project with that name was created.
4. Once succeeded in Postman, implement the same request using Python and the Requests library.
5. Don’t use hard coded values in your project. Values such as server IP address, port, username, password, etc. should be read from a dedicated configuration file.

Test Cases for Automation

Test 001 - API - Get Project by ID

Prerequisites:
A project named “TestProject1” with description “This is the first test project” already exists.

	Step	**Expected Result**
1	Send a request to get a project by ID. Use the ID of “TestProject1”	1. Response status: 200 2. Project name is “TestProject1” 3. Project description is “This is the first test project”

Test 002 - API - Update Project

Prerequisites:
A project named “TestProject1” already exists.

	Step	**Expected Result**
1	Send a request to update a project by ID. Use a unique string to set as the new project description.	Response contains a “project” object with description matching the value set in the request.

Test 003 - API - Create Project

	Step	Expected Result
1	Send a request to create a new project with a unique name.	Response contains a “project” object with name and identifier matching the values set in the request.

Test 004 - API - Delete Project

	Step	Expected Result
1	Send a request to create a new project with a unique name.	
2	Delete the newly created project. Send a request to delete a project by ID.	
3	Verify the project was deleted by sending a request to get a project by ID	Response status: 404

Test 005 - API - Get Work Package by ID

Prerequisites:
A “Task” work package with subject “My Task 1” already exists under a project named “TestProject1”.

	Step	Expected Result
1	Send a request to get a work package by ID. Use the ID of “My Task 1”	1. Response status: 200 2. Work package type is “Task” 3. Work package subject is “My Task 1”

Test 006 - API - Update Work Package

Prerequisites:
A “Task” work package with subject “My Task 1” already exists under a project named “TestProject1”.

	Step	Expected Result
1	Send a request to update the description of a work package. Use a unique string to set as the new	Response contains a “work package” object with description matching the value set in the request.

	description. **NOTE!** You must set the correct value for “lockVersion”. Read the API documentation.	

Test 007 - API - Create Work Package

Prerequisites:
A project named “TestProject1” already exists.

	Step	Expected Result
1	Send a request to create a new work package with a unique subject. **NOTE!** The work package should be created under the “TestProject1” project.	Response contains a “work package” object with subject matching the value set in the request.

Test 008 - API - Delete Work Package

	Step	Expected Result
1	Send a request to create a new work package with a unique subject.	
2	Delete the newly created work package. Send a request to delete a work package by ID.	
3	Verify the work package was deleted by sending a request to get a work package by ID	Response status: 404

UI Automation

General Guidelines for UI Tests Automation

1. Use Selenium WebDriver for browser automation.
2. Implement the automation using the “Page Object” design pattern.
3. Use “asserts” to implement validations for each expected result specified in the test cases below

Test Cases for Automation

Test 009 - UI - Create Project

	Step	**Expected Result**
1	Login to OpenProject	
2	On “Home” page, click “ + Project “ green button	
3	On the “New project” page, type a unique value for the project name. The name should contains letters (upper & lower case), numbers, spaces and some special characters (,/#@$%)	
4	Click “ADVANCED SETTINGS” title	More options are revealed
5	Type some text to the description text box	
6	Verify the value of the “Identifier” field	The identifier matches the value entered for the project name in step 3, with all lower case letters, and all spaces and special characters replaced with a dash (‘-’). Example: “Hello World 1$2@3” → “hello-world-1-2-3”
7	Select status “On track”	
8	Click “Create”	
9	On “Work packages” page, top left corner: verify the text on the button	Button text matches the project name entered in step 3

Test 010 - UI - Create Task

Prerequisites:
A project named “TestProject1” already exists.

	Step	Expected Result
1	Login to OpenProject	
2	On “Home” page, top-left corner, click “Select a project“ menu button, and select “TestProject1” from the drop-down	
3	On the “Project Overview” page, left side menu, click “Work packages”. Once on the “Work packages” page, note the number of rows displayed in the work packages table.	
4	Click “+ Create” green button and select “TASK”	
5	Verify the text “New TASK” on top of the form that got opened on the right side	The title of the form is “New TASK”
6	Type unique strings into the subject and description boxes	
7	Click “Save” button	
8	Verify that a new row was added to the work packages table	The number of table rows should be: initial number of rows (step 3) + 1 row
9	Verify the subject and type of the last table row	The subject should match the subject entered in step 6. The type should be “TASK”