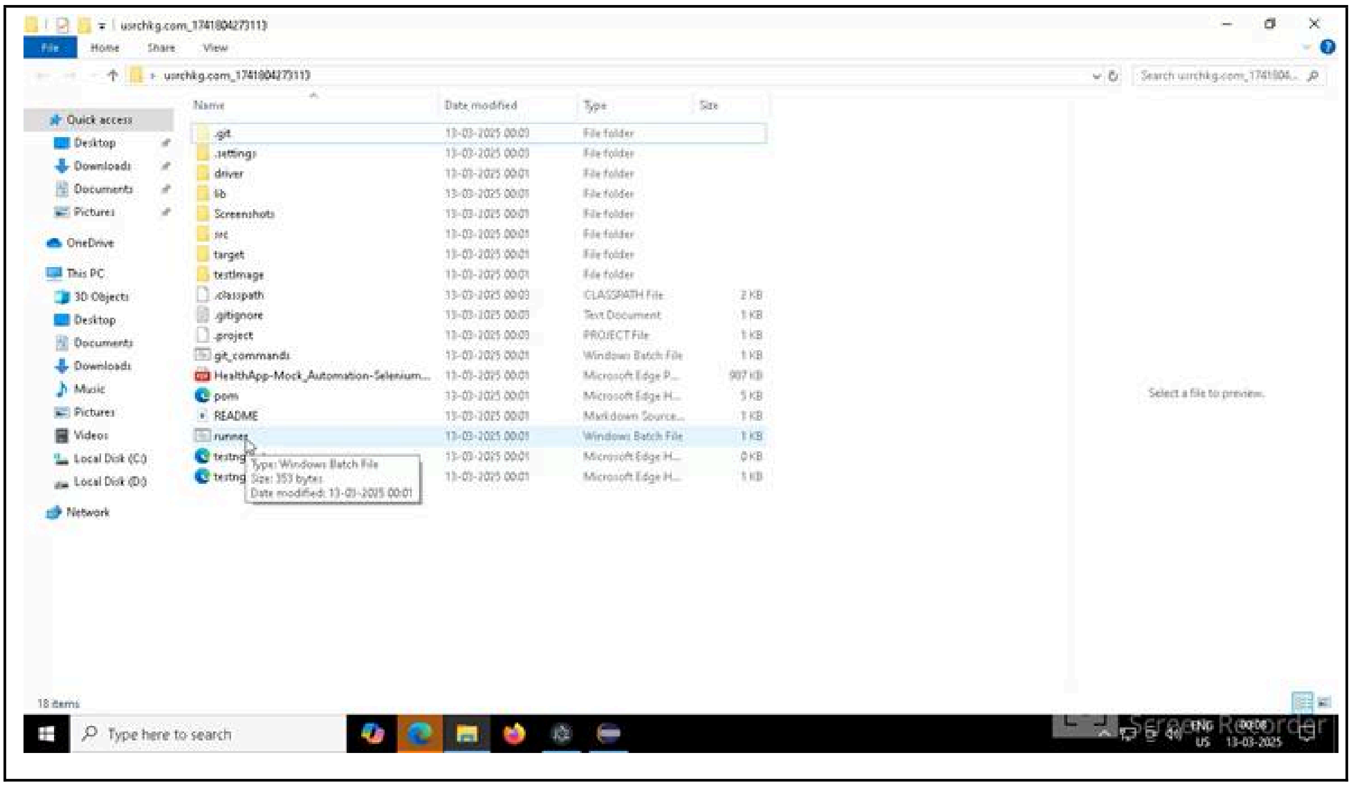
REST-ASSURED
(API - AUTOMATION PROJECT – PL1)

Pre-requisite:

Before you start working on your project, execute the runner file present in your project folder (Simply by double click). This is mandatory.



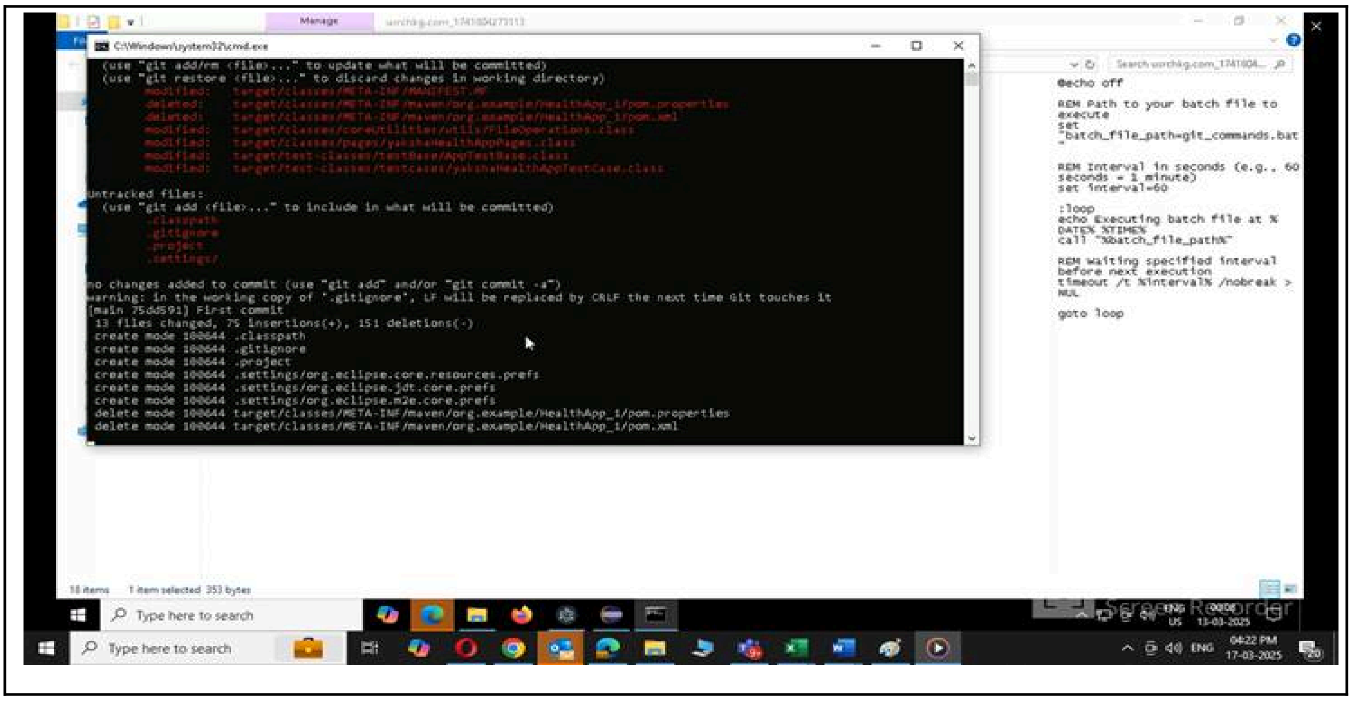
usrchkg.com_1741804273113

Name	Date modified	Type	Size
.git	13-03-2025 00:03	File folder	
.settings	13-03-2025 00:03	File folder	
driver	13-03-2025 00:01	File folder	
lib	13-03-2025 00:01	File folder	
Screenshots	13-03-2025 00:01	File folder	
src	13-03-2025 00:01	File folder	
target	13-03-2025 00:01	File folder	
testImage	13-03-2025 00:01	File folder	
.classpath	13-03-2025 00:03	CLASSPATH File	2 KB
.gitignore	13-03-2025 00:03	Text Document	1 KB
.project	13-03-2025 00:03	PROJECT File	1 KB
git_commands	13-03-2025 00:01	Windows Batch File	1 KB
HealthApp-Mock_Automation-Selenium...	13-03-2025 00:01	Microsoft Edge P...	907 KB
pom	13-03-2025 00:01	Microsoft Edge H...	5 KB
README	13-03-2025 00:01	Markdown Source...	1 KB
runner	13-03-2025 00:01	Windows Batch File	1 KB
testng	13-03-2025 00:01	Microsoft Edge H...	0 KB
testng	13-03-2025 00:01	Microsoft Edge H...	1 KB

Type: Windows Batch File
Size: 353 bytes
Date modified: 13-03-2025 00:01

18 items

This will launch a command terminal for you where it will keep on pushing your updated code to GIT on regular intervals. Keep that command terminal open at backend and you can continue working on your project.



C:\Windows\system32\cmd.exe

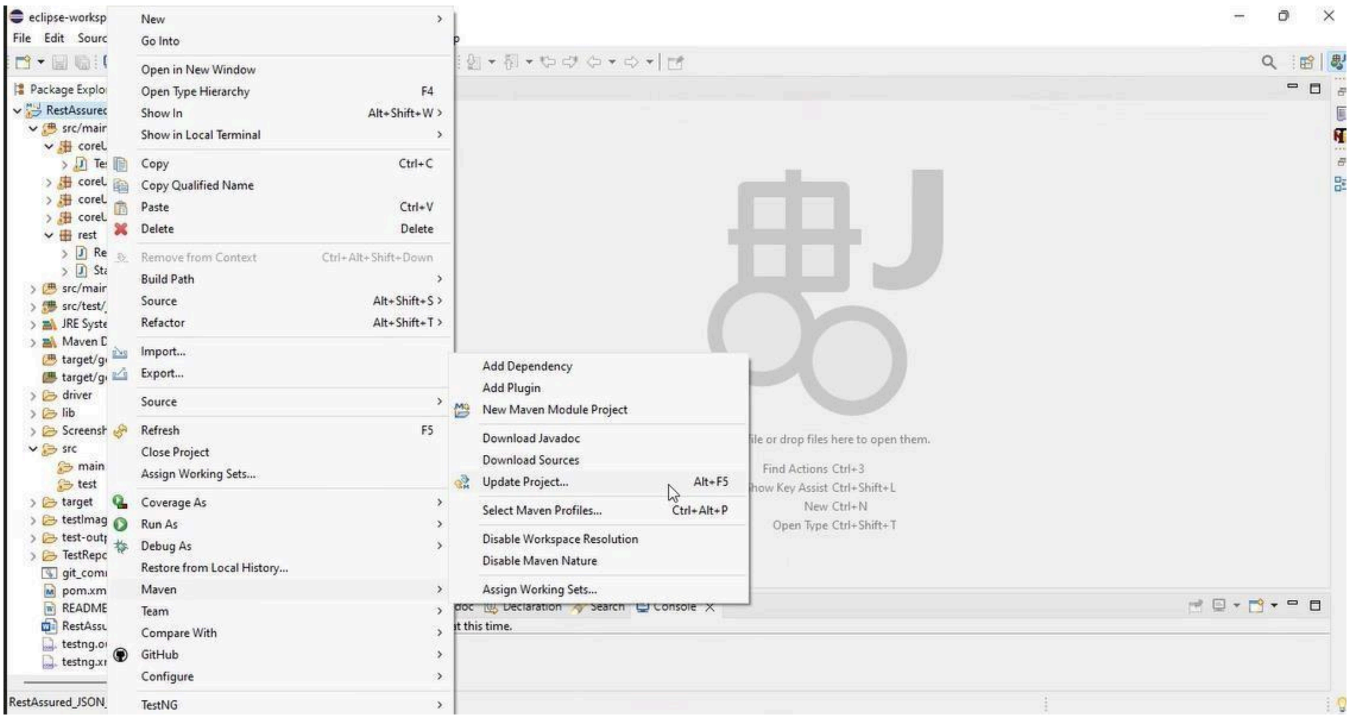
```
  (use "git add/rm <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   target/classes/META-INF/MANIFEST.MF
        deleted:    target/classes/META-INF/maven/org.example/HealthApp_1/pom.properties
        deleted:    target/classes/META-INF/maven/org.example/HealthApp_1/pom.xml
        modified:   target/classes/coreUtilities/utils/FileOperations.class
        modified:   target/classes/pages/yakshaHealthAppPages.class
        modified:   target/test-classes/testBase/AppTestBase.class
        modified:   target/test-classes/testcases/yakshaHealthAppTestCase.class

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .classpath
        .gitignore
        .project
        .settings/

no changes added to commit (use "git add" and/or "git commit -a")
warning: in the working copy of '.gitignore', LF will be replaced by CRLF the next time Git touches it
[main 75dd591] First commit
 13 files changed, 75 insertions(+), 151 deletions(-)
 create mode 100644 .classpath
 create mode 100644 .gitignore
 create mode 100644 .project
 create mode 100644 .settings/org.eclipse.core.resources.prefs
 create mode 100644 .settings/org.eclipse.jdt.core.prefs
 create mode 100644 .settings/org.eclipse.m2e.core.prefs
 delete mode 100644 target/classes/META-INF/maven/org.example/HealthApp_1/pom.properties
 delete mode 100644 target/classes/META-INF/maven/org.example/HealthApp_1/pom.xml
```

```
@echo off

REM Path to your batch file to
execute
set
"batch_file_path=git_commands.bat
"

REM Interval in seconds (e.g., 60
seconds = 1 minute)
set interval=60

:loop
echo Executing batch file at %
DATE% %TIME%
call "%batch_file_path%"

REM Waiting specified interval
before next execution
timeout /t %interval% /nobreak >
NUL

goto loop
```

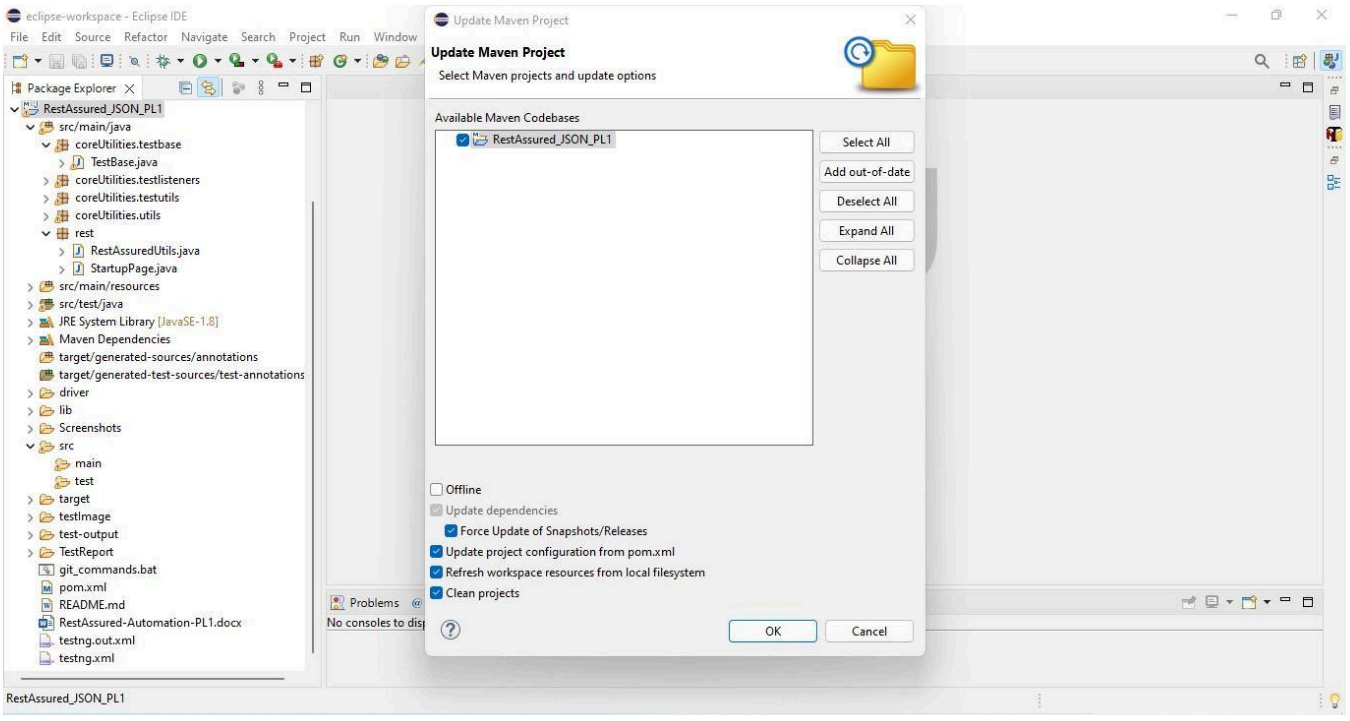
18 items   1 item selected 353 bytes

As soon as you import project in eclipse, update the project using maven update option as below. This is to resolve issue if any maven dependency not downloaded properly:

1. Right click on project : Go to “Maven” : Select “Update Project”



2. In Update Maven Project Box Select “Force Update of Snapshots/Releases” and click OK


Update Maven Project

Select Maven projects and update options

Available Maven Codebases

- [✓] RestAssured_JSON_PL1

- [ ] Offline
- [✓] Update dependencies
- [✓] Force Update of Snapshots/Releases
- [✓] Update project configuration from pom.xml
- [✓] Refresh workspace resources from local filesystem
- [✓] Clean projects

Template Code Structure:

- Below are the packages and files you will be required to work upon.
- Other Files and packages you can ignore.
- In other Files and packages do not do any changes. It would affect your evaluation.
- You are not required to work in “Test” Folder. Files there are noneditable. Editing those files and trying to save them will throw error and would affect your evaluation.

Package	**Class/File**	**Description**
src/main/java/coreUtilities/utils/	FileOperations.java	1. It contains methods to read data from Excel files. 2. The method is in templated form. 3. You will be required to implement these methods as the first activity, because even the URL to navigate to is read using these methods.
/src/main/java/rest	ApiUtil.java	1. All core activities to be performed here. 2. The comments associated with each templated method here describe the expectation. 3. Declare any 4. variable/object you need to share data/status between different methods. 5. Do not modify the signature of methods declared here. 6. You can create additional supportive common methods in 7. CommonEvents class.
/src/main/resources/	Config.xlsx	1. Data present to be used in creating a new Post.
/src/main/resources/	TestData.xlsx	1. Data is present for creating requests.
/src/main/java/coreUtilities/utils	CommonEvents.java	1. Contains all common activities. 2. Certain templated common methods are declared here. 3. You implement them as per your needs. 4. You can add any additional method for common activity here
	Testng.xml	Execution needs to be kick-started from TestNG.xml

PROBLEM STATEMENT

Need to automate the following activities using RestAssured.

Key Activities to implement:

#	Summary	Action	Expected Result

1	Retrieve Localization Details with Valid Cookie Authentication using method: getLocalizationValid(String endpoint, String cookieValue, Map<String, String> body)	1. Construct the final URL by combining the BASE_URL and the provided endpoint parameter:
		  a. https://opensource-demo.orangehrmlive.com/web/index.php/api/v2/admin/localization
		2. Initialize a RequestSpecification using RestAssured:
		  a. Add a cookie with name "orangehrm" and the value passed as cookieValue.
		  b. Set the "Content-Type" header to "application/json".
		3. Trigger a GET request to the above URL.
		4. Wrap language and dateFormat values in List<Object> containers.
		5. Create a CustomResponse object with:
		  a. Full Response
		  b. Extracted statusCode
		  c. Extracted statusLine
		  d. languages list
		  e. dateFormat list
		6. Return the CustomResponse object.
			1. Method should return a fully initialized CustomResponse object with:
			  a. statusCode = 200
			  b. status = OK (or valid status line like "HTTP/1.1 200 OK")
			  c. A non-null and non-empty languages list
			  d. A non-null and non-empty dateFormat list
			  e. The complete raw Response object

2	Retrieve a list of active languages using session cookie authentication through method: getActiveLanguages(String endpoint, String cookieValue, Map<String, String> body)	1. Construct the complete URL by concatenating the BASE_URL and the provided endpoint:
		  a. https://opensource-demo.orangehrmlive.com/web/index.php/api/v2/admin/i18n/languages?limit=50&offset=0&sortOrder=ASC&activeOnly=true
		2. Create a RequestSpecification using RestAssured:
			1. The method should return a CustomResponse object with:
			  a. statusCode = 200
			  b. status equal to a valid HTTP status line like "HTTP/1.1 200 OK"
			  c. Non-null and non-empty lists for:
			    i. languageId
			    ii. languageName
			    iii. languageCode

		a. Add a session cookie named "orangehrm" with the provided cookieValue. b. Add a header "Content-Type" with the value "application/json". 3. If a request body is provided (not null), attach it to the request using .body(). 4. Send a GET request to the constructed URL. 5. Wrap the extracted fields into lists<Object>: a. languageId, languageName, languageCode 6. Create and return a CustomResponse object initialized with: a. The complete Response b. Extracted statusCode c. Extracted statusLine d. languageId, languageName, languageCode lists	d. The complete raw Response object
3	Retrieve dashboard shortcut access permissions using session-based authentication through the method: getDashboardShortcuts(String endpoint, String cookieValue, Map<String, String> body)	1. Construct the final URL by combining the BASE_URL and the provided endpoint: ● https://opensource-demo.orangehrmlive.com/web/index.php/api/v2/dashboard/shortcuts 2. Create a RequestSpecification using RestAssured: ● Add a session cookie named "orangehrm" using the provided cookieValue. ● Set the request header "Content-Type" to "application/json". 3. If a request body (body) is provided (i.e., not null), attach it using .body() method. 4. Wrap each of these fields into separate List<Object> collections: ● leaveAssignLeave, leaveLeaveList, leaveApplyLeave, leaveMyLeave, timeEmployeeTimesheet, timeMyTimesheet	1. The method should return a CustomResponse object containing: ● statusCode = 200 ● A valid status line such as "HTTP/1.1 200 OK" ● Non-null and non-empty lists for the following keys: ● leave.assign_leave ● leave.leave_list ● leave.apply_leave ● leave.my_leave ● time.employee_timesheet ● time.my_timesheet ● The complete Response object from the API call

		5. Create a CustomResponse object using: ● The complete Response object ● Extracted statusCode and statusLine ● All six field lists 6. Return the populated CustomResponse object.	
4	Retrieve a list of OpenID authentication providers using session-based authentication through method: getAuthProviders(String endpoint, String cookieValue, Map<String, String> body)	1. Ensure valid data exists for the GET call by first triggering a POST request using createAuthProviderTest()to create an OpenID provider entry. 2. Construct the final URL by concatenating BASE_URL and the given endpoint: https://opensource-demo.orangehrmlive.com/web/index.php/api/v2/auth/openid-providers?limit=50&offset=0 3. Create a RequestSpecification using RestAssured: ● Include the session cookie "orangehrm" using the provided cookieValue ● Add the header "Content-Type" with value "application/json" 4. If a request body (body) is provided, attach it using .body(). 5. Trigger a GET request to the above URL. 6. Store each field as a List<Object>: ● providerIds, providerNames, providerUrls, providerStatuses, clientIds 7. Return a CustomResponse object populated with:	● Method must return a valid CustomResponse object containing: ● statusCode = 200 ● status like "HTTP/1.1 200 OK" ● Non-null and non-empty lists for: ▪ providerIds ▪ providerNames ▪ providerUrls ▪ providerStatuses ▪ clientIds o The full raw Response object

		<ul><li>Full Response object</li><li>statusCode and statusLine</li><li>Extracted field lists</li></ul>	
5	Retrieve a list of registered OAuth clients in the admin panel using session-based cookie authentication via method: getAdminOAuthList(String endpoint, String cookieValue, Map<String, String> body)	1. Construct the final URL by appending the endpoint to the base URL: https://opensource-demo.orangehrmlive.com/web/index.php/api/v2/admin/oauth-clients?limit=50&offset=0 2. Use RestAssured to initialize a RequestSpecification:<ul><li>Add the session cookie "orangehrm" using the provided cookieValue</li><li>Set the request header "Content-Type" to "application/json"</li></ul>3. If a request body (body) is not null, include it in the request. 4. Store all extracted fields in respective List<Object> collections. 5. Create an instance of OAuthClientInfo and populate it with the extracted fields. 6. Return a CustomResponse object initialized with:<ul><li>The full Response</li><li>statusCode and statusLine</li><li>The OAuthClientInfo object</li></ul>	<ul><li>The method must return a CustomResponse containing:<ul><li>statusCode = 200</li><li>A valid status line like "HTTP/1.1 200 OK"</li><li>An OAuthClientInfo object with non-null and non-empty lists for:<ul><li>id</li><li>name</li><li>clientId</li><li>redirectUri</li><li>enabled</li><li>confidential</li></ul></li><li>The full raw Response object</li></ul></li></ul>
6	Retrieve a list of employees with detailed information using session-based authentication via method: getEmployeeList(String endpoint, String cookieValue, Object body)	1. Construct the complete URL by appending the endpoint to the base URL: https://opensource-demo.orangehrmlive.com/web/index.php/api/v2/pim/employees?limit=50&offset=0&model=detailed&includeEmployees=onlyCurrent&sortField=employee.firstName&sortOrder=ASC	<ul><li>The method should return a CustomResponse object with:<ul><li>statusCode = 200</li><li>A valid status string (e.g., "HTTP/1.1 200 OK")</li><li>Non-null and non-empty lists for:<ul><li>empNumber</li></ul></li></ul></li></ul>

		2. Initialize a RequestSpecification using RestAssured: ● Add the session cookie "orangehrm" with the provided cookieValue. ● Set the header "Content-Type" to "application/json". 3. If a request body (body) is provided (not null), include it in the request using .body(). 4. Trigger a GET request to the constructed URL and extract the Response. 5. Wrap all extracted values into individual List<Object> collections. 6. Create and return a CustomResponse object containing: ● The full Response ● The extracted statusCode and statusLine ● The employee detail lists (empNumber, firstName, lastName, employeeId)	▪ firstName ▪ lastName ▪ employeeId ● The complete raw Response object
7	Update an existing language package using session-based authentication through method: putLanguagePackage(String endpoint, String cookieValue, Object body)	1. Retrieve a language package ID not present in the current language list using helper method: getMissingLanguageIdLessThan450(). 2. Construct the final URL using the base URL and the language package ID as a path variable: https://opensource-demo.orangehrmlive.com/web/index.php/api/v2/admin/i18n/languages/{lang_package_id} 3. Initialize a RequestSpecification using RestAssured:	● The method should return a valid CustomResponse containing: ● statusCode = 200 ● A valid HTTP status (e.g., "HTTP/1.1 200 OK") ● Non-empty string values for: ▪ adminLangId ▪ adminLangName ▪ adminLangCode

		<ul><li>Add the "orangehrm" cookie with the given cookieValue.</li><li>Set the request header "Content-Type" to "application/json".</li></ul>4. If the body is not null, include it using .body(). 5. Trigger a PUT request to the endpoint and extract the Response. 6. Store these values in individual string variables:<ul><li>adminLangId, adminLangName, adminLangCode</li></ul>7. Return a CustomResponse object initialized with:<ul><li>The full Response</li><li>statusCode and statusLine</li><li>Extracted string values: adminLangId, adminLangName, adminLangCode</li></ul>	<ul><li>The complete raw Response object</li></ul>
8	Delete one or more language packages using session-based authentication via method: deleteLangById(String endpoint, String cookieValue, String requestBody)	1. Retrieve the first available language ID using helper method getfirstlangid(). 2. Add the request body while executing the api to get the correct response. 3. Compose the full URL by appending the endpoint to the base URL: https://opensource-demo.orangehrmlive.com/web/index.php/api/v2/admin/i18n/languages 4. Initialize a RequestSpecification using RestAssured:<ul><li>Add a session cookie named "orangehrm" with the provided cookieValue</li></ul>	<ul><li>The method should return a CustomResponse containing:<ul><li>statusCode = 200</li><li>A valid HTTP status line such as "HTTP/1.1 200 OK"</li><li>A non-null and non-empty List<Object> containing the deleted language IDs</li><li>The full raw Response object</li></ul></li></ul>

		• Set the "Content-Type" header to "application/json" 5. Attach the constructed requestBody to the request. 6. Send a DELETE request to the full URL and extract the Response object. 7. Return a CustomResponse object with: • The full Response object • Extracted statusCode and statusLine • The deletedIds list	
9	Update the enabled/disabled status of various system modules via a PUT request with session-based cookie authentication using method: putModulesSettings(String endpoint, String cookieValue, String requestBody)	1. Construct the complete URL by appending the endpoint to the base URL: https://opensource-demo.orangehrmlive.com/web/index.php/api/v2/admin/modules 2. Initialize a RequestSpecification using RestAssured: • Add the cookie "orangehrm" with the value provided by cookieValue. • Set the header "Content-Type" to "application/json". • Attach the JSON request body using .body(requestBody). 3. Trigger a PUT request to the above endpoint and extract the Response. 4. Parse the response using JsonPath and retrieve the "data" object as a Map<String, Object> which holds the current module status.	• The method must return a CustomResponse object that includes: • statusCode = 200 • A valid status line (e.g., "HTTP/1.1 200 OK") • A non-null Map<String, Object> containing updated module state entries • The complete raw Response object

		5. Return a CustomResponse object initialized with: ● The full Response object ● statusCode and statusLine ● Parsed moduleSettings map containing key–boolean pairs (e.g., "admin": true)	

10	Create a new OpenID provider entry using a POST request with session-based authentication via method: postOpenIdProvide(String endpoint, String cookieValue, String requestBody)	1. Read OpenID provider data (name, url, clientId, clientSecret) from Excel (PostData10) using FileOperations.readExcelPOI.
		2. Generate a unique provider name by appending a random suffix to the base name to avoid duplication:
		String uniqueName = baseName + UUID.randomUUID().toString().substring(0, 8);
		3. Construct a JSON requestBody with the following format:
		json
		{
		  "name": "unique_name",
		  "url": "provider_url",
		  "clientId": "client_id_value",
		  "clientSecret": "client_secret_value"
		}
		4. Build the full API endpoint URL:
		https://opensource-demo.orangehrmlive.com/web/index.php/api/v2/auth/openid-providers
		5. Initialize a RequestSpecification using RestAssured:
		● Add the "orangehrm" session cookie
		● Set "Content-Type" header to "application/json"
		● Attach the constructed requestBody using .body()
			● The method should return a valid CustomResponse object containing:
			● statusCode = 200
			● A valid status string like "HTTP/1.1 200 OK"
			● Non-null, correctly mapped values for:
			▪ name
			▪ url
			▪ clientId
			▪ clientSecret
			● The full Response object

		6. Trigger a POST request to the endpoint and extract the Response object. 7. Return a CustomResponse object with: ● The full Response ● statusCode, statusLine ● Extracted fields: name, url, clientId, clientSecret	

NOTE: "Please do not delete any file in the src folder. But you are free to add any other file".

Expectations:

1) **Learners should write automation scripts using Java and REST Assured to automate the API testing for all the provided methods (e.g., GET, POST, PUT, DELETE).** In other words, the automation script should perform all mentioned API interactions, including validation of responses.
2) **Learners should not use any pre-built libraries or tools to validate API responses (e.g., JSON schema validation tools).** They should manually validate the response content (e.g., status codes, response body, etc.) by writing their own logic for assertion.

IMPLEMENTATION/FUNCTIONAL REQUIREMENT

1.1 CODE QUALITY/OPTIMIZATIONS

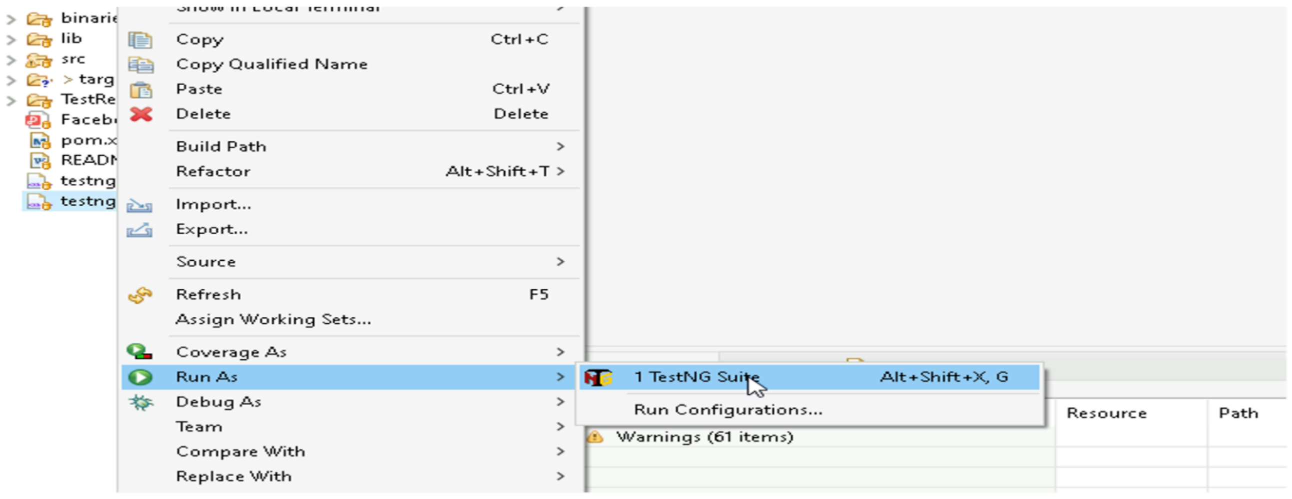
1. Associates should have written clean code that is readable.
2. Associates need to follow SOLID programming principles.

EXECUTION STEPS TO FOLLOW

---

1. **You are mandatory required to run test cases for applications before final submission. Without this project evaluation will not happen.**

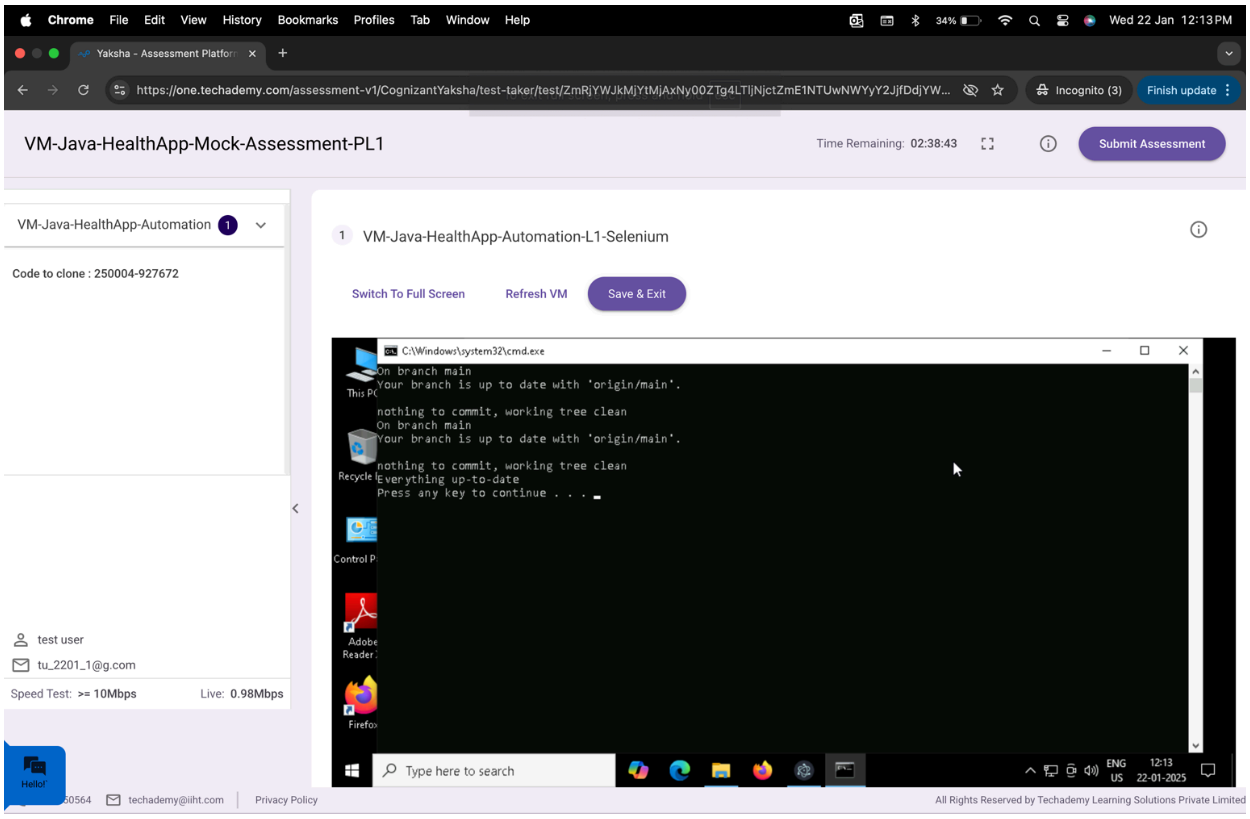
2. **You can launch test cases any time as follows: Right-click on testng.xml and run TestNGSuite.**



NOTE:

- **When executing test cases via TestNG.xml, it launches the application UI using Selenium WebDriver. This is intentional behavior to facilitate cookie extraction for authentication.**

3. To do the final submission of the assessment :
    a. Press escape to come out of Fullscreen mode.
    b. Submit the assessment.


Chrome File Edit View History Bookmarks Profiles Tab Window Help
Wed 22 Jan 12:13 PM
Yaksha - Assessment Platform
https://one.techademy.com/assessment-v1/CognizantYaksha/test-taker/test/ZmRjYWJkMjYtMjAxNy00ZTg4LTljNjctZmE1NTUwNWYyY2JjfDdjYW...
Incognito (3)
Finish update
VM-Java-HealthApp-Mock-Assessment-PL1
Time Remaining: 02:38:43
Submit Assessment
VM-Java-HealthApp-Automation
Code to clone : 250004-927672
1 VM-Java-HealthApp-Automation-L1-Selenium
Switch To Full Screen
Refresh VM
Save & Exit
C:\Windows\system32\cmd.exe
test user
tu_2201_1@g.com
Speed Test: >= 10Mbps
Live: 0.98Mbps
Type here to search
techademy@iiht.com
Privacy Policy
All Rights Reserved by Techademy Learning Solutions Private Limited

```
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
Everything up-to-date
Press any key to continue . . .
```

After the successful submission of the assessment, you will get a confirmation message displayed on your screen.

==

All the Best