URBAN LADDER-PL1

(JAVA,SELENIUM & CUCUMBER)

Table of Contents

Pre-Requisite

1. Before you start working on your project, **execute the runner file** present in your project folder (Simply by double click) and keep it running. <mark>This is mandatory.</mark>
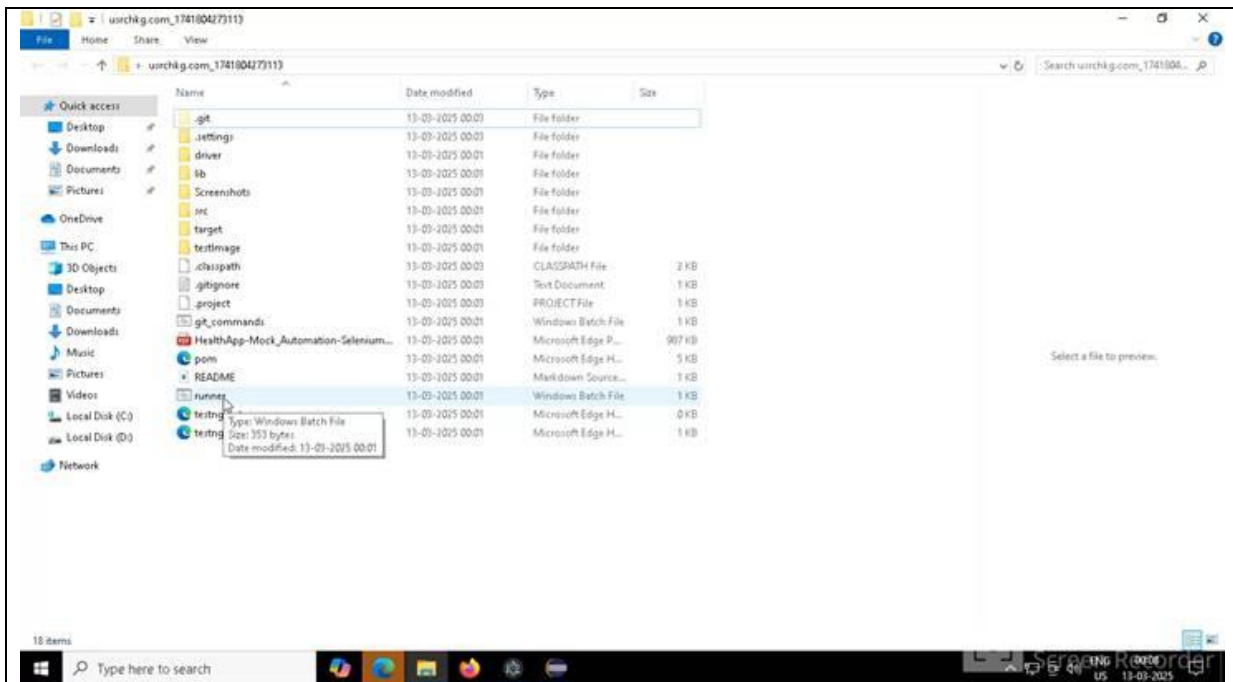

This will launch a command terminal for you where it will keep on pushing your updated code to GIT on regular intervals. Keep that command terminal open at backend and you can continue working on your project.
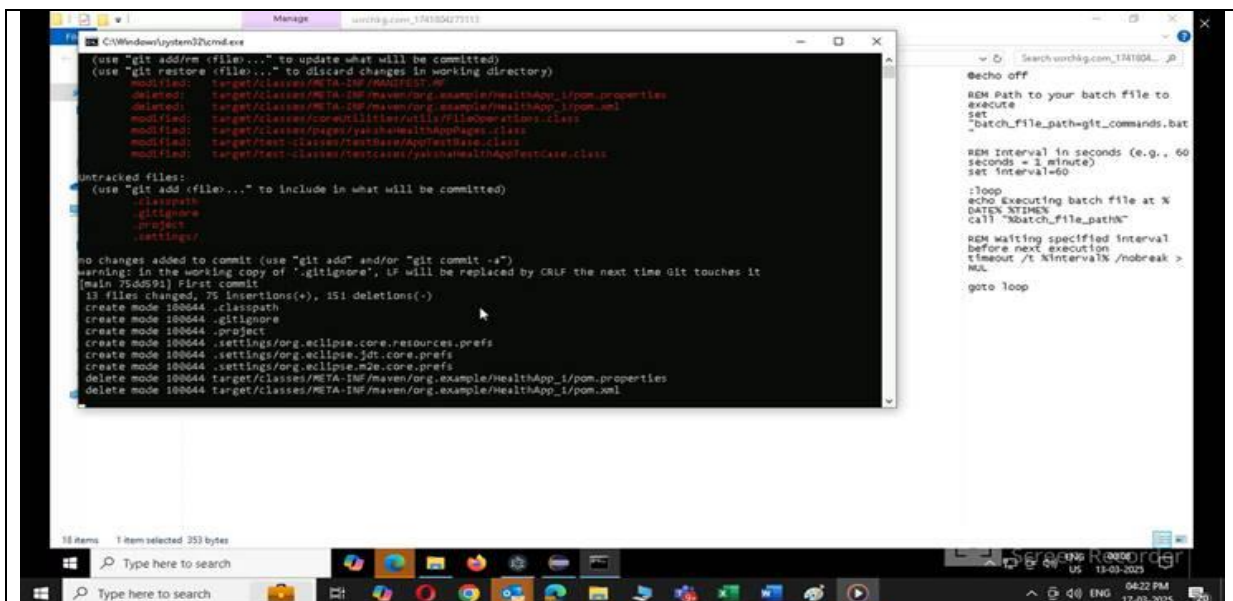
2. Launch the Eclipse IDE(Shortcut icon on VM's Desktop) and import your cloned project to Eclipse as an Existing maven project.
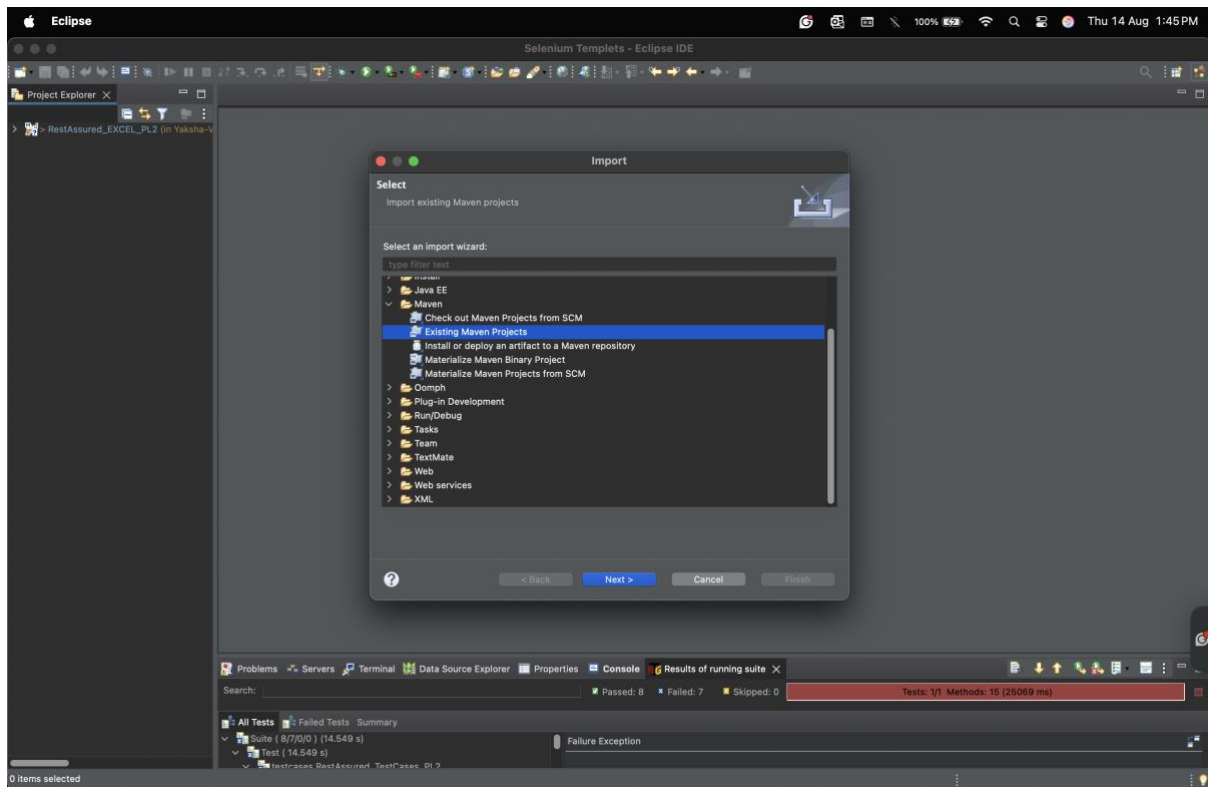
3. As soon as you import the project in Eclipse, update the project using the maven update option as below(if required) . This is to resolve the issue if any Maven dependency is not downloaded properly.
    a. Right-click on the project: Go to "Maven" and select "Update Project".
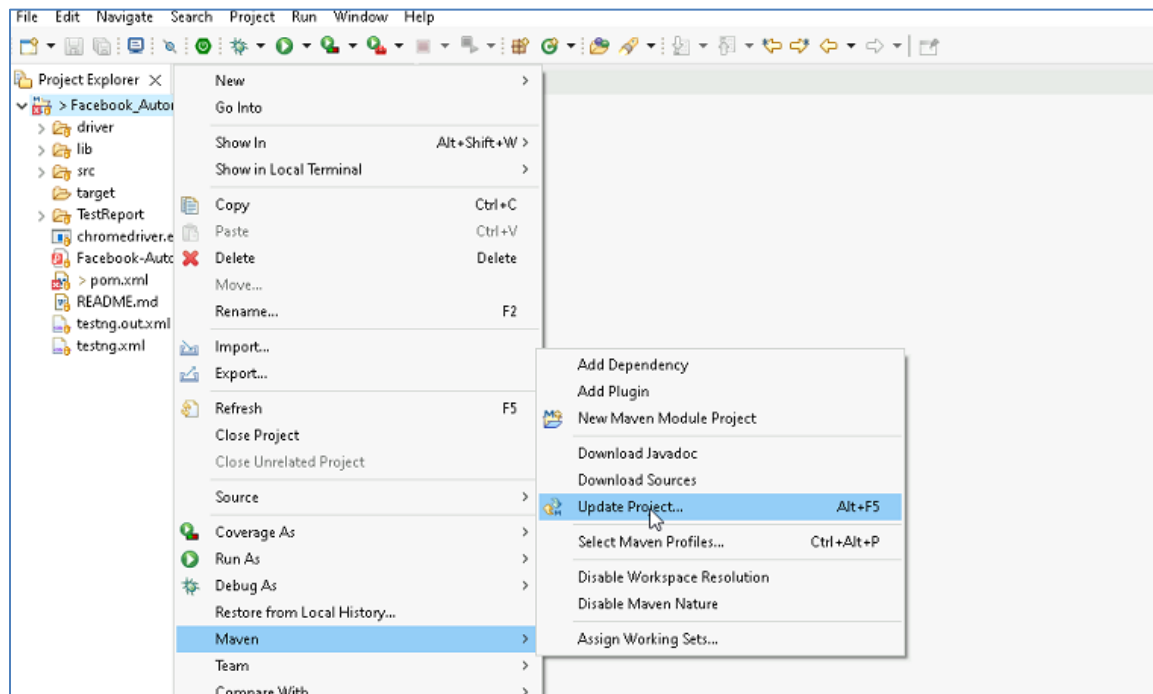
    b. In the Update Maven Project Box Select "Force Update of Snapshots/Releases" and click OK.
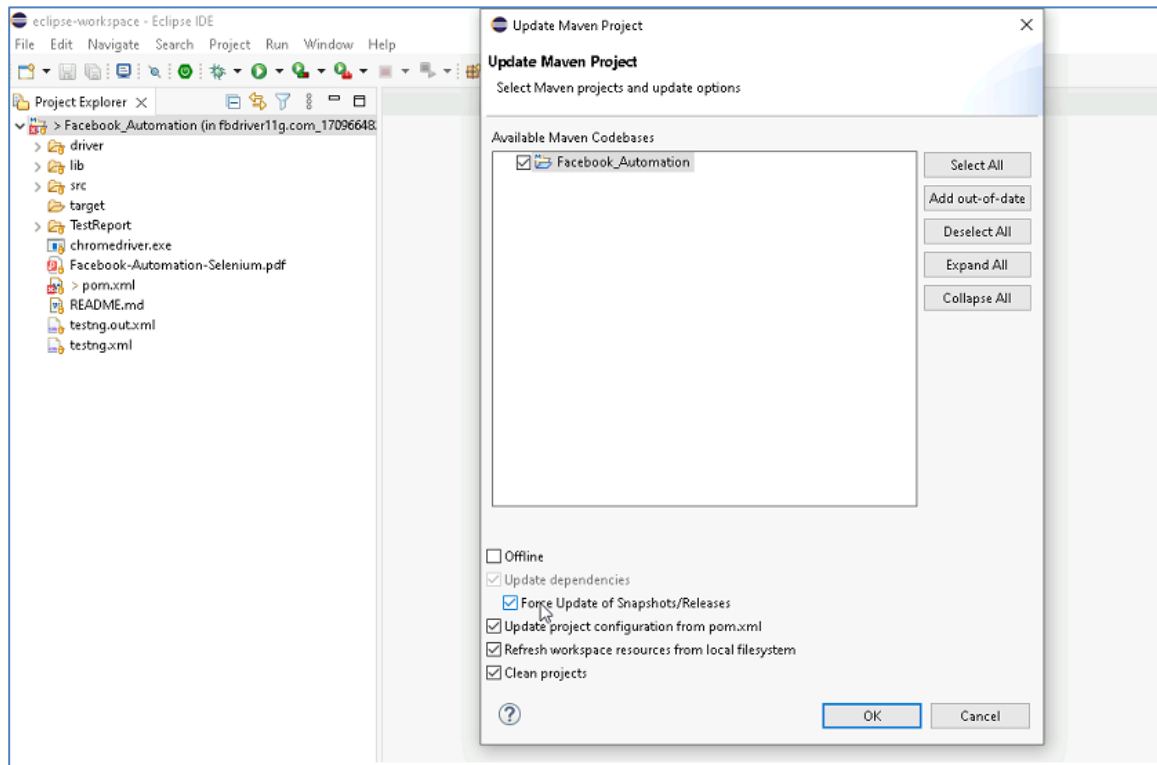
Template Code Structure:

Instructions:

1. You only need to work on the packages and files listed in the table below.
2. Ignore all other files and packages.
3. Do **not** make any changes to files outside the listed ones, as this will affect your evaluation.
4. The **"Test"** folder is non-editable. Any attempt to edit and save files there will cause errors and impact your evaluation.

Package	Class/File	Description
src/main/java/coreUtilities/utils/	FileOperations.java	• The method for reading data as input from an Excel file have already been implemented here. • This method is used to fetch the required data from excel including the URL for navigation.
src/main/java/coreUtilities/utils/	CommonEvents.java	• Create additional supportive common methods here.
/src/main/java/pages	FooterPage.java HomePage.java SearchPage.java SignupPage.java	• Perform all core activities listed under the table "Key Activities to implement" here.(This table is given below.) • Refer to the comments in each templated method to understand what's expected. • Define locators and XPaths in this file. • Declare any variables or objects needed to share data or status between methods. • Do not change the method signatures. If needed. create additional supportive common methods in the CommonEvents class.
/src/test/java/features/	01_Homepage.feature 02_SignUp.feature 03_SearchPage.feature 04_FooterLinks.feature	• The file has incomplete scenarios. • You must fill them in according to the requirement. • Steps should follow Given-When-Then structure. • The scenarios should fully validate the requirement.
src/test/java/steps/	FooterStep.java HomePageStep.java SearchPageStep.java SignUpStep.java	• It's a step definition file. • Each step in Verification_PL1.feature must be linked here.
/src/main/resources/	Config.xlsx	• URL to navigate to is already defined here
	Login.xlsx	• Contains valid credentials required for use login
	expected_data.xlsx	• Contains data to fill in form during TCs execution.
	testng_1.xml	• Execution needs to kick started from testng_1.xml

PROBLEM STATEMENT:

We need to automate the following activities using Java+Selenium+Cucumber.

Key Activities to implement:

Sl No.	Summary	Action	Expected Result
1	Verify that the homepage loads successfully.	• The user is on the homepage of Urban Ladder.	The homepage of Urban Ladder should load successfully, and the page title contains 'Urban Ladder'.
2	Verify user can sign up successfully.	• The user is on the homepage of Urban Ladder. • Hover on the profile icon • Click on the Sign Up button • Complete the signup process with random email and password as "Testing1!	Verify that "My Account" should be visible on the homepage after successfully completing the sign-up process with the specified credentials on Urban Ladder.
3	Verify user can LOG OUT successfully.	• The user is on the homepage of Urban Ladder. • Hover on the profile icon • Click on the Sign Up button • Complete the signup process with some random email and password as "Testing1!" • Click on logout button	Verify that the User successfully logs out and sees the Login Button on the Urban Ladder homepage.
4	Verify the search functionality with a valid product	• The user is on the homepage of Urban Ladder. • User search for "Sofa" from search bar. • Locate all product title elements from the search results. • Iterate through the product titles. • Verify that each product title contains the expected product name	The search results on Urban Ladder's homepage should display a list of products related to "Sofa."
5	Verify that a product can be added to the Wishlist.	• The user is on the homepage of Urban Ladder. • User search for "Table" from search bar • Capture the current wishlist count before adding any product. • Retrieve and store the name of the first product from the search results for later verification. • Convert the wishlist count into an integer and log it. • Click on the wishlist (heart) icon for the first product. • Capture the wishlist count after adding a product to the wishlist. • Convert the captured count into an integer. • Print the parsed wishlist count for debugging purposes. • Compare the current wishlist count with the previously stored count.	The first 'Table' product from the search results should be successfully added to the Wishlist.
6	Verify that the product details page opens.	• The user is on the homepage of Urban Ladder. • User search for "Chairs" from search bar • Click on view product button from the search results. • Capture the product name displayed on the product details page. • Compare it with the product name stored earlier from the search results.	Ensure that the product details page for 'Chairs' opens with accurate and complete product information displayed.
7	Verify that price filters work correctly.	• The user is on the homepage of Urban Ladder. • Search for "Sofa" from search bar • The user is on the homepage of Urban Ladder.	All sofas displayed are within the selected price range after

		Hover over the price filter option Click on the price filter element. Select the price range option. Capture the selected price range text from the filter element. Extract the minimum and maximum price values from the range. Wait for products to refresh and enable the "In Stock Only" filter for accurate results. Retrieve all product prices displayed in the search results. Parse each product's price and validate it falls within range.	applying the price filter on Urban Ladder's homepage.
8	Verify product details in the wishlist.	• The user is on the homepage of Urban Ladder. • User search for "Table" from search bar • Capture the current wishlist count before adding any product. • Retrieve and store the name of the first product from the search results for later verification. • Convert the wishlist count into an integer and log it. • Click on the wishlist (heart) icon for the first product. • Click on the wishlist icon in the header to navigate to the wishlist page. • Capture the name of the first product listed in the wishlist. • Compare the captured wishlist product name with the product name stored earlier when it was added to the wishlist.	The expected output of the scenario should confirm that the first product listed under "Table" in the search results is successfully added to the wishlist, and its details such as name on the wishlist page match those from the search results page.
9	Verify product can be added to compare section.	• The user is on the homepage of Urban Ladder. • Search for "Dining" from search bar. • Hover over the first product in the search results to make the hidden options visible. • Locate and click on the "Add to Compare" button for the first product. • Locate the comparison widget section on the page. • Check if at least one product widget is displayed inside the comparison section.	Ensure that first product from the "Dining" search results appears in the product compare section.
10	Verify the functionality of footer links.	• The user is on the homepage of Urban Ladder. • Click on the Contact Us link in the footer • The Contact Us page should open with relevant information • Locate the Contact Us heading element on the page.	The Contact Us page opens displaying relevant contact information such as phone number, email address, and physical address.

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 script using Java and selenium to automate all the steps in the above question. In other words, automation script should perform all mentioned steps.

2) Learners should not use any tools to create the xpath. They should develop the xpath/cssselector on their own.

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 as TestNGSuite.
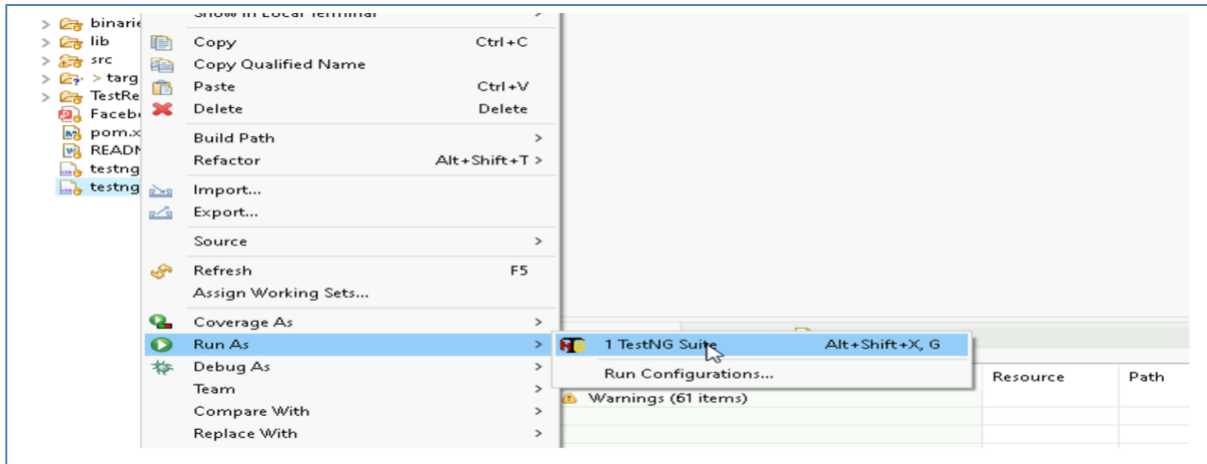
3. To do the final submission of the assessment :
   a. Press escape to come out of Fullscreen mode.
   b. Submit the assessment **(Press the button on right hand side , top corner).**
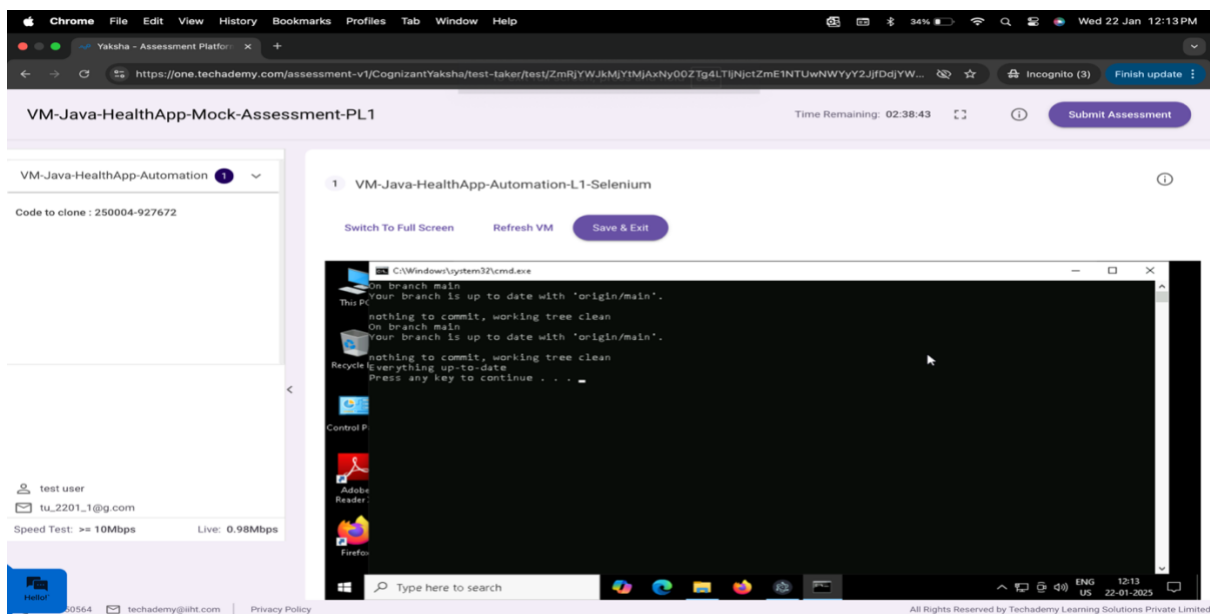
After the successful submission of the assessment, you will get a confirmation message displayed on your screen.

==

All the Best