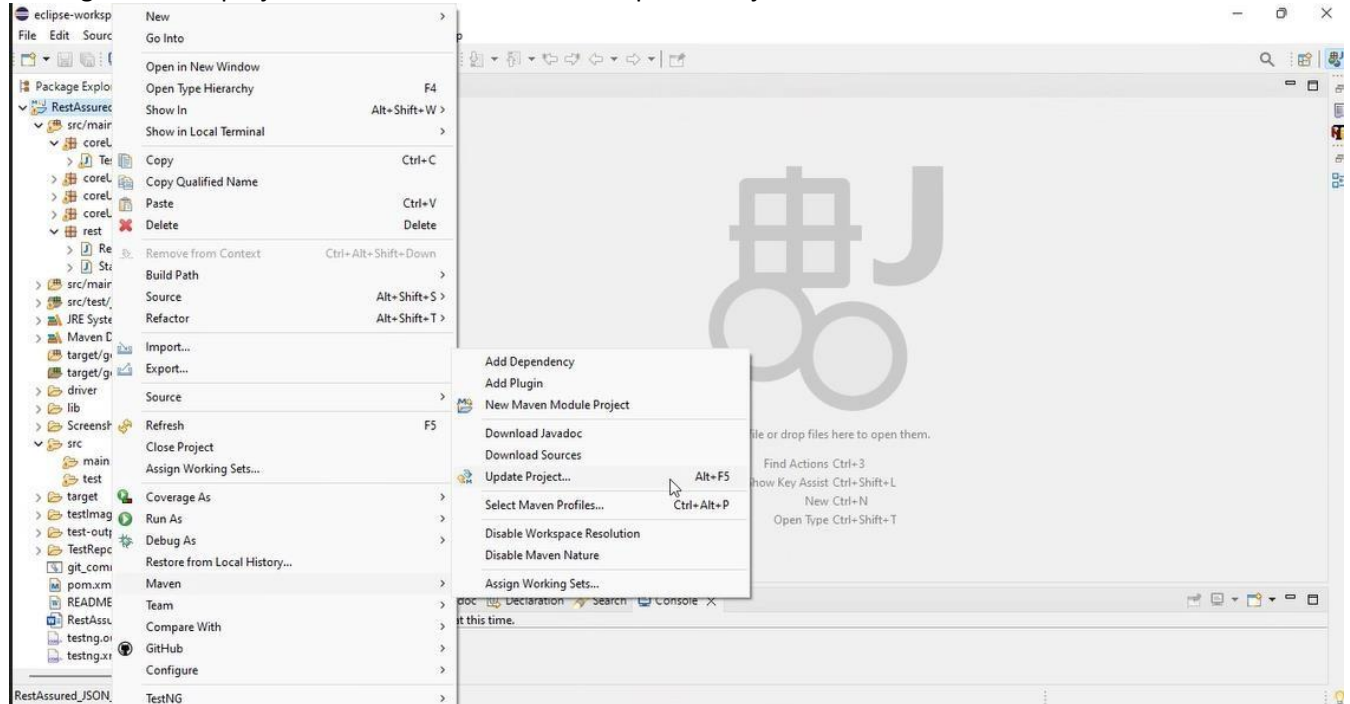


# RESTASSURED API AUTOMATION PROJECT

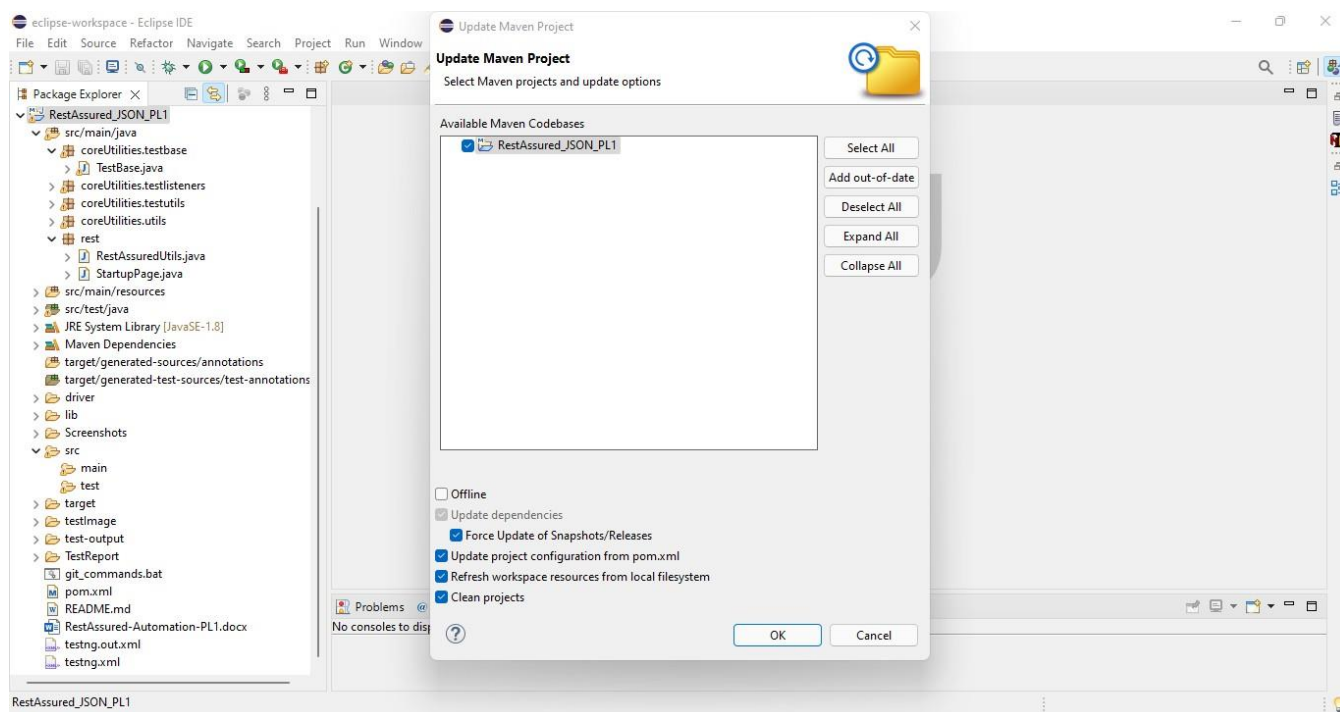
## Pre-requisite:

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



## Template Code Structure:

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

Package	Class/File	Description
src/main/java/coreUtilities/utis/	FileOperations.java	<ol style="list-style-type: none"><li>1. Contains methods to read from excel file.</li><li>2. Method is in templated form.</li><li>3. <b>You will be required to implement these methods as very first activity, because for creating post the data should be read from excel.</b></li></ol>
/src/main/java/rest	ApiUtil.java	<ol style="list-style-type: none"><li>1. All core activities to be performed here.</li><li>2. The comments associated with each templated method here describe the expectation.</li><li>3. Declare any variable/object you need to share data/status between different methods.</li><li>4. Do not modify the signature of methods declared here.</li><li>5. You can create additional supportive common methods in CommonEvents class.</li></ol>
/src/main/java/rest	AuthUtil.java	<ol style="list-style-type: none"><li>1. Class already defined to read and return bearer token from</li></ol>

		config.properties file.
/src/main/resources/	Config.xlsx	Data present to be used in Implementing functions.

/src/main/java/coreUtilities/utls	CommonEvents.java	<ol style="list-style-type: none"> <li>1. Contains all common activities.</li> <li>2. Certain templated common method declared here.</li> <li>3. You implement them as per your need.</li> <li>4. You can add any additional method for common activity here</li> </ol>
	Testng.xml	Execution needs to kick started from TestNG xml

## PROBLEM STATEMENT

Need to automate the following activities using RestAssured.

## Key Activities to implement:

Below activities need to be implemented in ApiUtil.java file present in src/main/java/rest package.

Sl No.	Summary	Action	Expected Result
1	Create an Appointment with Authorization in Method:  createAppointmentWithAuth(String endpoint, Map<String, String> body)	<ol style="list-style-type: none"> <li>1. Call the POST on endpoint i.e <b><a href="https://healthapp.yaksha.com/api/Appointment/AddAppointment">https://healthapp.yaksha.com/api/Appointment/AddAppointment</a></b></li> <li>2. Provide the appointment details in the body parameter, including FirstName, LastName, Gender, Age, ContactNumber, AppointmentDate, AppointmentTime, PerformerName, AppointmentType, and DepartmentId.</li> <li>3. Include an authorization header for the request.</li> <li>4. Validate the response contains the appointment details in the response body.</li> </ol>	Returns a 200 OK status with the created appointment details in the response.
2	Get All Applicable Doctors with Authorization in Method:  getAllApplicableDoctorsWithAuth(String endpoint, Object body)	<ol style="list-style-type: none"> <li>1. Call the GET on endpoint i.e <b><a href="https://healthapp.yaksha.com/api/Appointment/AddAppointment/Visit/AppointmentApplicableDoctors">https://healthapp.yaksha.com/api/Appointment/AddAppointment/Visit/AppointmentApplicableDoctors</a></b></li> <li>2. If a request body is needed, include it in the body parameter.</li> </ol>	Returns a 200 OK status with the list of applicable doctors in the response.

		3. Ensure an authorization header is included. 4. Validate the response contains a list of applicable doctors.	
3	Cancel an Appointment with Authorization in Method:  cancelAppointmentWithAuth (String endpoint, Object body)	1. Call the PUT on endpoint i.e <b>https://healthapp.yaksha.com/api/Appointment/AppointmentStatus?appointmentId=+ appointmentId</b>  2. If a body is required, pass it as the body parameter; otherwise, leave it null. 3. Include an authorization header. 4. Validate the response confirms the cancellation of the appointment.	Returns a 200 OK status if the appointment is successfully canceled.
4	Find Clashing Appointments with Authorization in Method:  clashAppoitnmentWithAuth (String endpoint, Object body)	1. Call the GET on endpoint i.e <b>https://healthapp.yaksha.com/api/Appointment/CheckClashingAppointment?patientId="+ patientId + "&amp;requestDate=" + requestDate + "&amp;performerId=" + performerId</b>  2. Pass query parameters in the request if required. 3. Include an authorization header. 4. Validate the response for clashing appointments, ensuring it contains a list of appointments in the Results field.	Returns a 200 OK status with a list of clashing appointments or a 404 status if no clashes are found.
5	Search for a Patient with Authorization in Method:  searchPatientWithAuth(String endpoint, Object body)	1. Call the GET on endpoint i.e <b>https://healthapp.yaksha.com/api/Patient/SearchRegisteredPatient?search=Test</b>  2. Include the necessary query parameters in the body if required. 3. Add an authorization header to the request. 4. Validate the response contains patient details matching the search criteria, including fields like PatientId, ShortName, FirstName, LastName, Age, etc.	Returns a 200 OK status with a list of patients matching the search criteria in the response.
6	Retrieve a List of Appointments for a Specified Performer in Method:  bookingListWithAuthInRange(String endpoint, Object body)	1. Call the GET on endpoint i.e <b>https://healthapp.yaksha.com/api/Appointment/Appointments?FromDate="+ dateFiveDaysBeforeStr + "&amp;ToDate=" + currentDateStr + "&amp;performerId=" + performerId + "&amp;status=new"</b>	Returns a 200 OK status with a list of appointments matching the specified performer and date range.

		<ol style="list-style-type: none"> <li>2. Include query parameters for FromDate, ToDate, and PerformerId in the request body if necessary.</li> <li>3. Add the required authorization header.</li> <li>4. Validate that the response contains appointments matching the specified criteria, including fields like AppointmentId, PatientId, FullName, AppointmentDate, AppointmentTime, AppointmentStatus, etc.</li> </ol>	
7	<p>Retrieve All Stock Details in Method:</p> <p>AllStockDetailsWithAuth(String endpoint, Object body)</p>	<ol style="list-style-type: none"> <li>1. Call the GET on endpoint i.e <b><a href="https://healthapp.yaksha.com/api/PharmacyStock/AllStockDetails">https://healthapp.yaksha.com/api/PharmacyStock/AllStockDetails</a></b></li> <li>2. Add the necessary authorization header.</li> <li>3. Validate that: <ol style="list-style-type: none"> <li>a. The response status code is 200 (OK).</li> <li>b. Each ItemId in the results is not null.</li> <li>c. The Status field in the response is "OK".</li> </ol> </li> </ol>	Returns a 200 OK status with the list of stock items, ensuring all ItemId fields are populated.
8	<p>Retrieve Main Store Details in Method:</p> <p>MainStoreDetailsWithAuth(String endpoint, Object body)</p>	<ol style="list-style-type: none"> <li>1. Call the GET on endpoint i.e <b><a href="https://healthapp.yaksha.com/api/PharmacySettings/MainStore">https://healthapp.yaksha.com/api/PharmacySettings/MainStore</a></b></li> <li>2. Add the necessary authorization header.</li> <li>3. Validate that: <ol style="list-style-type: none"> <li>a. The response status code is 200 (OK).</li> <li>b. Fields Name, StoreDescription, and StoreId are not null.</li> <li>c. The Status field in the response is "OK".</li> </ol> </li> </ol>	Returns a 200 OK status with the main store details, ensuring all essential fields (Name, StoreDescription, StoreId) are populated.
9	<p>Retrieve a List of Pharmacy Stores in Method:</p> <p>PharmacyStoresWithAuth(String endpoint, Object body)</p>	<ol style="list-style-type: none"> <li>1. Call the GET on endpoint i.e <b><a href="https://healthapp.yaksha.com/api/Dispensary/PharmacyStores">https://healthapp.yaksha.com/api/Dispensary/PharmacyStores</a></b></li> <li>2. Add the necessary authorization header.</li> <li>3. Validate that: <ol style="list-style-type: none"> <li>a. The response status code is 200 (OK).</li> <li>b. Each store in the results has non-null StoreId and Name fields.</li> <li>c. The Status field in the response is "OK".</li> </ol> </li> </ol>	Returns a 200 OK status with the list of pharmacy stores, ensuring all StoreId and Name fields are populated for each store.

10	<p>Retrieve Patient Consumption Details in Method:</p> <p>PatientConsumption(String endpoint, Object body)</p>	<ol style="list-style-type: none"> <li>1. Call the GET on endpoint i.e <b><a href="https://healthapp.yaksha.com/api/PatientConsumption/PatientConsumptions">https://healthapp.yaksha.com/api/PatientConsumption/PatientConsumptions</a></b></li> <li>2. Add the required authorization header.</li> <li>3. Validate that: <ol style="list-style-type: none"> <li>a. The response status code is 200 (OK).</li> <li>b. Each record in the results contains non-null PatientId and PatientName.</li> <li>c. The Status field in the response is "OK".</li> </ol> </li> </ol>	Returns a 200 OK status with the list of patients' consumption details, ensuring all PatientId and PatientName fields are populated for each record.
11	<p>Activate a Pharmacy Counter Using Details in Method:</p> <p>ActivatePharmCount(String endpoint, Object body)</p>	<ol style="list-style-type: none"> <li>1. Call the PUT on endpoint i.e <b><a +counterid+"&amp;countername="+counterName" href="https://healthapp.yaksha.com/api/Security/ActivatePharmacyCounter?counterId=">https://healthapp.yaksha.com/api/Security/ActivatePharmacyCounter?counterId=" + counterId + "&amp;counterName=" + counterName</a></b></li> <li>2. Include counterId and counterName in the query parameters or body.</li> <li>3. Add the required authorization header.</li> <li>4. Validate the response to ensure: <ol style="list-style-type: none"> <li>a. The status code is 200 (OK).</li> <li>b. The Results field contains non-null CounterName and CounterId.</li> <li>c. The Status field is "OK".</li> </ol> </li> </ol>	Returns a 200 OK status with details of the activated pharmacy counter in the Results field.
12	<p>Deactivate a Pharmacy Counter in Method:</p> <p>DeactivatePharmCount(String endpoint, Object body)</p>	<ol style="list-style-type: none"> <li>1. Call the PUT on endpoint i.e <b><a href="https://healthapp.yaksha.com/api/Security/DeactivatePharmacyCounter">https://healthapp.yaksha.com/api/Security/DeactivatePharmacyCounter</a></b></li> <li>2. Include the necessary authorization header.</li> <li>3. Validate the response to ensure: <ol style="list-style-type: none"> <li>a. The status code is 200 (OK).</li> <li>b. The Results field contains a StatusCode of "200".</li> <li>c. The Status field is "OK".</li> </ol> </li> </ol>	Returns a 200 OK status with details confirming the pharmacy counter has been deactivated.
13	<p>Retrieve Appointment Applicable Departments in Method:</p> <p>AppointApplicDept(String endpoint, Object body)</p>	<ol style="list-style-type: none"> <li>1. Call the GET on endpoint i.e <b><a href="https://healthapp.yaksha.com/api/Master/AppointmentApplicableDepartments">https://healthapp.yaksha.com/api/Master/AppointmentApplicableDepartments</a></b></li> <li>2. Add the required authorization header.</li> <li>3. Validate the response to ensure: <ol style="list-style-type: none"> <li>a. The status code is 200 (OK).</li> <li>b. Each item in the Results list</li> </ol> </li> </ol>	Returns a 200 OK status with a list of applicable departments, ensuring DepartmentId and DepartmentName fields are populated.



		contains non-null DepartmentId and DepartmentName. c. The Status field is "OK".	
14	Retrieve Admitted Patients Data in Method:  admittedPatientData(String endpoint, Object body)	<ol style="list-style-type: none"> <li>1. Call the GET on endpoint i.e <b><a href="https://healthapp.yaksha.com/api/Admission/AdmittedPatientsData?admissionStatus=admitted">https://healthapp.yaksha.com/api/Admission/AdmittedPatientsData?admissionStatus=admitted</a></b></li> <li>2. Add the necessary authorization header.</li> <li>3. Validate the response to ensure: <ol style="list-style-type: none"> <li>a. The status code is 200 (OK).</li> <li>b. Each item in the Results list contains non-null PatientId and AdmittedDate, and DischargedDate is null.</li> <li>c. The Status field is "OK".</li> </ol> </li> </ol>	Returns a 200 OK status with the list of admitted patients, ensuring proper data is returned for each patient.
15	Add a New Currency in Method:  addCurrencyWithAuth(String endpoint, Map<String, String> body)	<ol style="list-style-type: none"> <li>1. Call the POST on endpoint i.e <b><a href="https://healthapp.yaksha.com/api/InventorySettings/Currency">https://healthapp.yaksha.com/api/InventorySettings/Currency</a></b></li> <li>2. Provide CurrencyCode, Description, CreatedBy, CreatedOn, and IsActive in the request body.</li> <li>3. Include the authorization header.</li> <li>4. Validate the response to ensure the currency is added successfully.</li> </ol>	Returns a 200 OK status with details of the added currency in the response.
16	Find Matching Patient by Phone Number in Method:  findMatchingPatientWithAuth(String endpoint, Object body)	<ol style="list-style-type: none"> <li>1. Call the GET on endpoint i.e <b><a &amp;age="+age+" &amp;gender="+gender+" &amp;imiscode="+imisCode" &amp;isinsurance="+isInsurance+" &amp;phonenumber="+phoneNumber+" +firstname+"&amp;lastname="+lastName+" href="https://healthapp.yaksha.com/api/Patient/MatchingPatients?FirstName=">https://healthapp.yaksha.com/api/Patient/MatchingPatients?FirstName="+ firstName + "&amp;LastName=" + lastName + "&amp;PhoneNumber=" + phoneNumber + "&amp;Age=" + age + "&amp;Gender=" + gender + "&amp;IsInsurance=" + isInsurance + "&amp;IMISCode=" + imisCode</a></b></li> <li>2. Provide the phone number in the request body.</li> <li>3. Add the required authorization header.</li> <li>4. Validate the response to ensure: <ol style="list-style-type: none"> <li>a. The status code is 200 (OK).</li> <li>b. The response contains matching patient details in the Results field.</li> </ol> </li> </ol>	Returns a 200 OK status with the list of matching patients in the Results field.

17	<p>Retrieve and Verify Registered Patients in Method:</p> <p>getRegisteredPatientsWithAuth(String endpoint, Object body)</p>	<ol style="list-style-type: none"> <li>1. Call the GET on endpoint i.e <b><a href="https://healthapp.yaksha.com/api/Patient/SearchRegisteredPatient?search=">https://healthapp.yaksha.com/api/Patient/SearchRegisteredPatient?search=</a></b></li> <li>2. Add the necessary authorization header.</li> <li>3. Validate the response to ensure: <ol style="list-style-type: none"> <li>a. The status code is 200 (OK).</li> <li>b. Each patient in the Results list has a unique PatientId.</li> </ol> </li> </ol>	<p>Returns a 200 OK status with the list of registered patients, ensuring all PatientId fields are unique.</p>
18	<p>Retrieve Billing Counters Data in Method:</p> <p>getBillingCountersWithAuth(String endpoint, Object body)</p>	<ol style="list-style-type: none"> <li>1. Call the GET on endpoint i.e <b><a href="https://healthapp.yaksha.com/api/billing/BillingCounters">https://healthapp.yaksha.com/api/billing/BillingCounters</a></b></li> <li>2. Add the necessary authorization header.</li> <li>3. Validate the response to ensure: <ol style="list-style-type: none"> <li>a. The status code is 200 (OK).</li> <li>b. The Results field contains details of all billing counters.</li> </ol> </li> </ol>	<p>Returns a 200 OK status with the list of billing counters in the Results field.</p>

**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

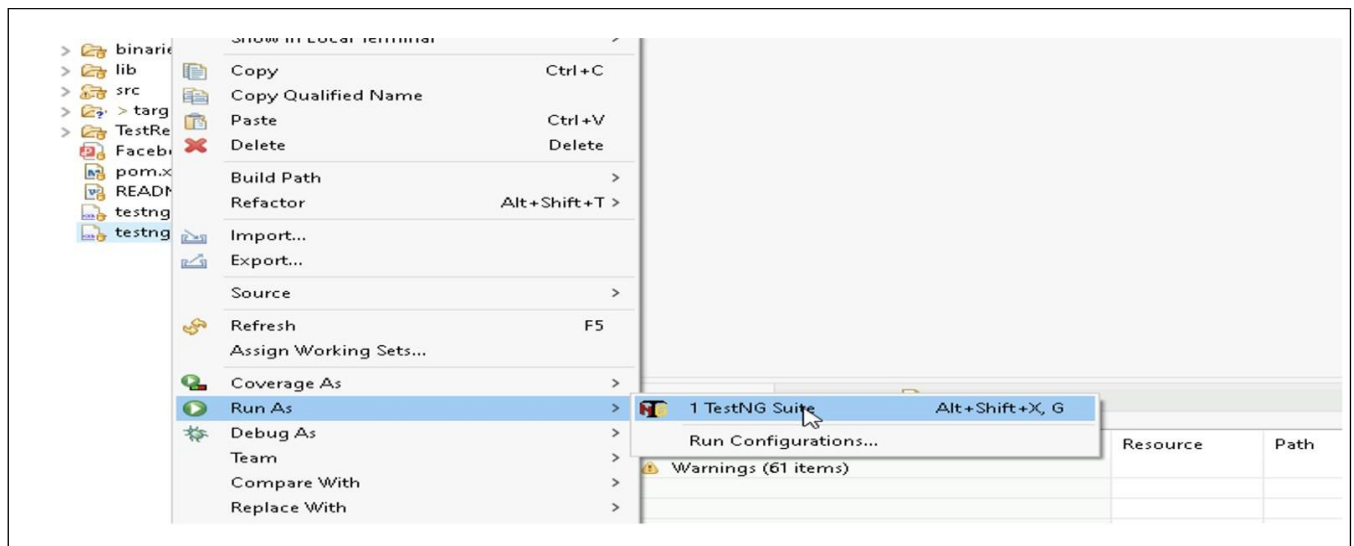
### 1.1 CODE QUALITY/OPTIMIZATIONS

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

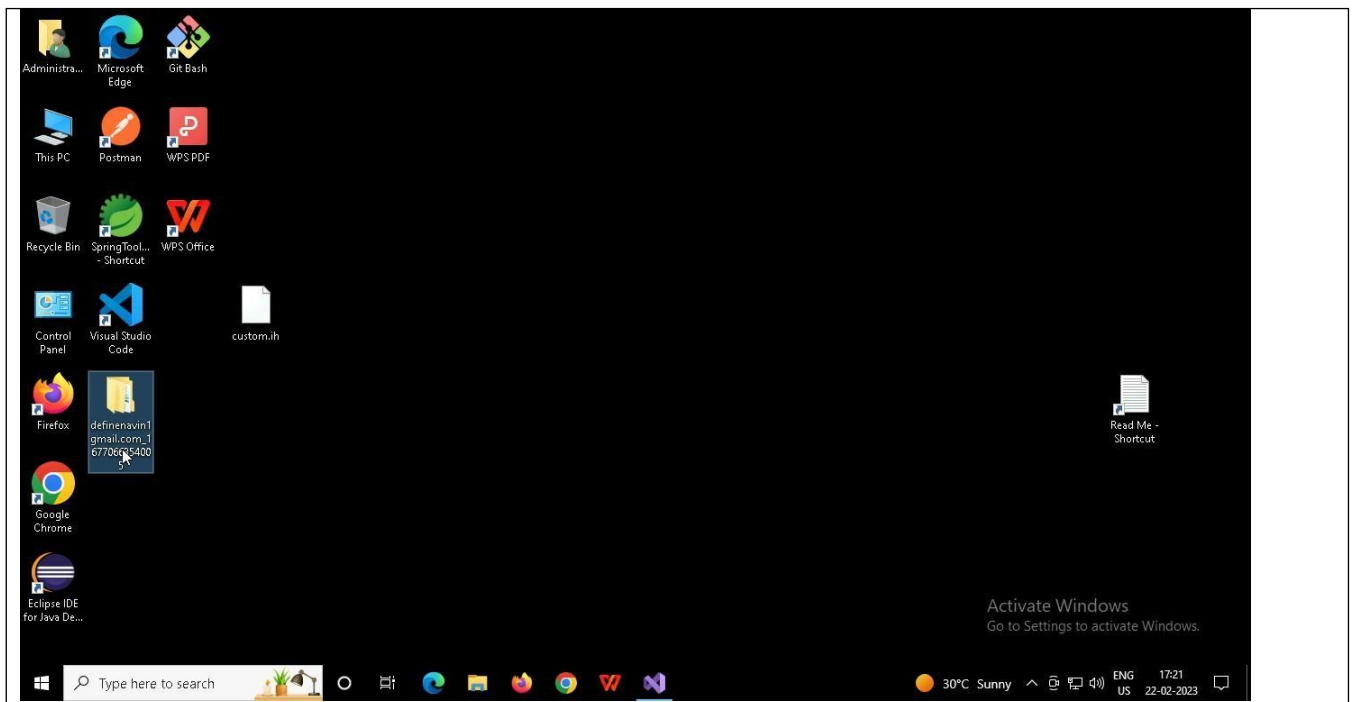
## EXECUTION STEPS TO FOLLOW

---

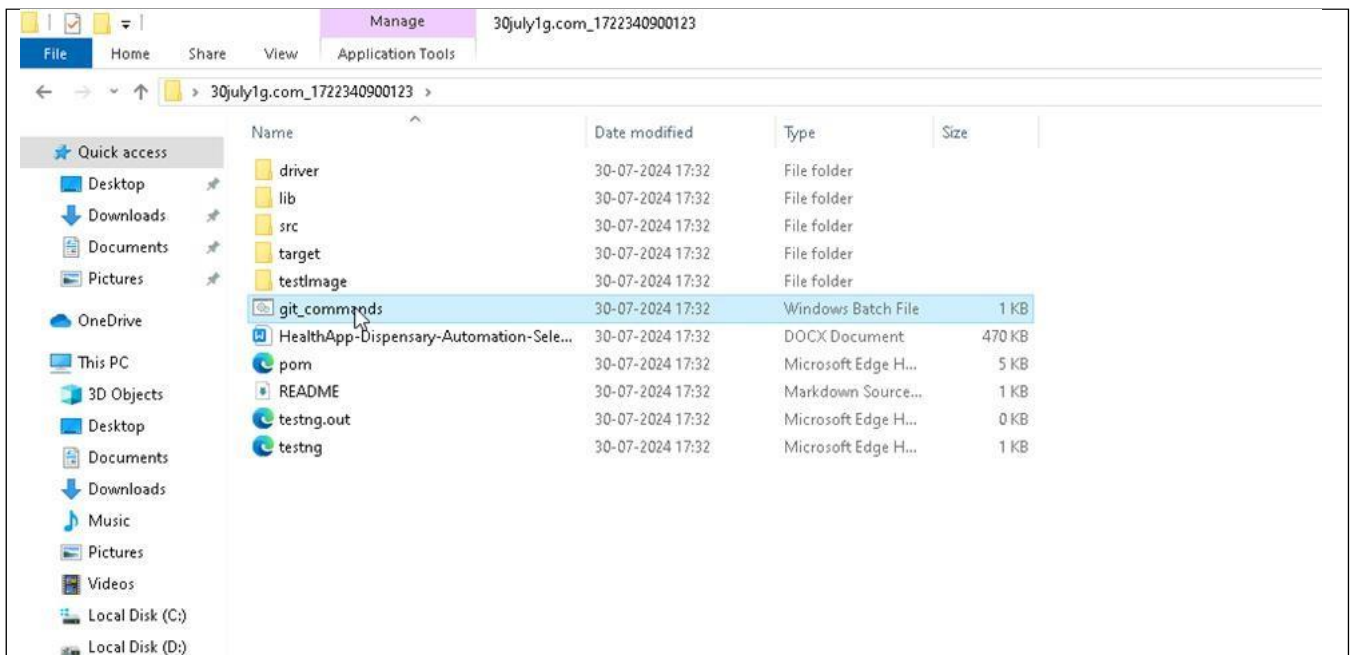
1. **You are mandatory required to run test cases for applications before final submission. Without which project evaluation will not happen.**
2. **You can launch test cases any time as follows: Right click on testng.xml and run TestNGSuite**



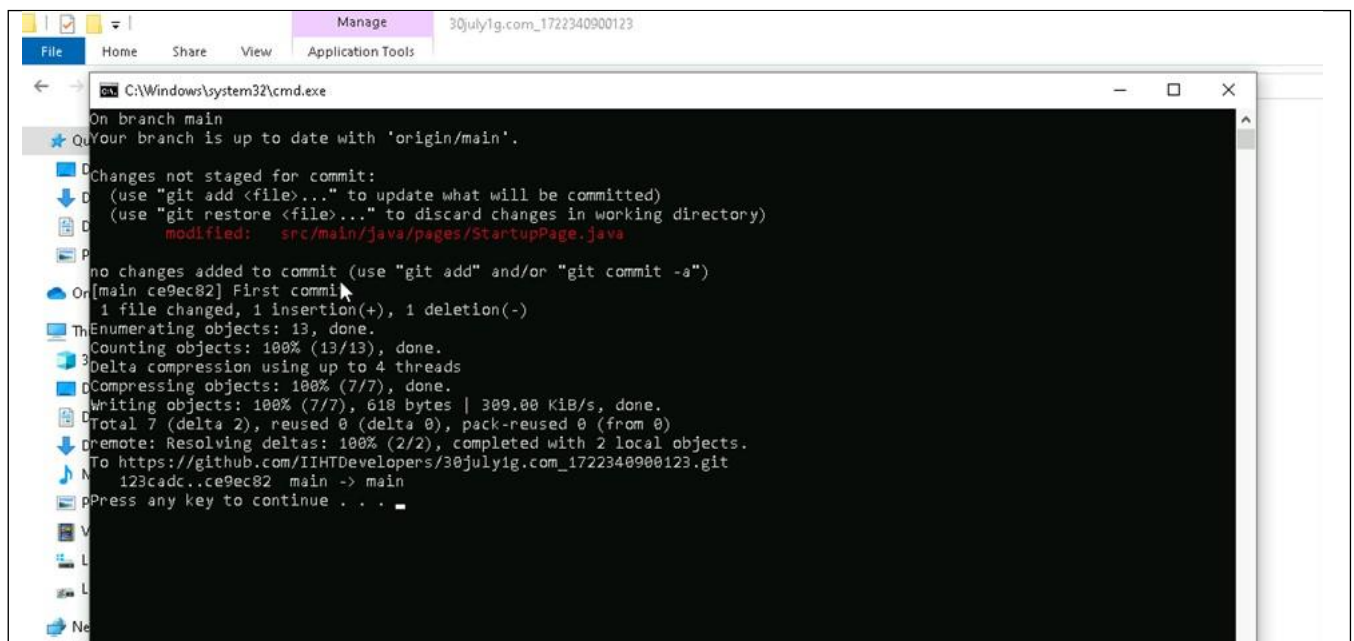
3. **Before final submission, you are also required to push your code to GIT. Following are the steps to follow:**



In your project folder, you will find a batch file named `git_commands`



Double-click the batch file to run it. It will run the commands to push your code to GIT.



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

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   src/main/java/pages/StartupPage.java

no changes added to commit (use "git add" and/or "git commit -a")
On [main ce9ec82] First commit
  1 file changed, 1 insertion(+), 1 deletion(-)
Enumerating objects: 13, done.
Counting objects: 100% (13/13), done.
Delta compression using up to 4 threads
Compressing objects: 100% (7/7), done.
Writing objects: 100% (7/7), 618 bytes | 309.00 KiB/s, done.
Total 7 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/IIHTDevelopers/30july1g.com_1722340900123.git
   123cadc..ce9ec82  main -> main
Press any key to continue . . .
```

=====

## All the Best