HEALTHAPP AUTOMATION PATIENT MODULE – PL2

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

HealthApp Automation – Patient Module

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/utils/	FileOperations.java	1. Contains methods to read from excel file. 2. Method is in templated form. **You will be required to implement these methods as very first activity, because even URL to navigate to, is read using these methods.**
/src/main/java/pages	automation_page.java	1. All core activities (mentioned in list above) to be performed here. 2. The comments associated with each templated method here describe the expectation. 3. You can define locators and xpath here. 4. Declare any 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 CommonEvents class.
/src/main/resources/	Config.xlsx	URL to navigate to. Already URL is defined here
	expected_data.xlsx	Contains data to fill in form\

HealthApp Automation – Patient Module

/src/main/java/coreUtilities/utils	CommonEvents.java	1. Contains all common activities. 2. Certain templated common method declared here. 3. You implement them as per your need. 4. You can add any additional method for common activity here
	Testng.xml	Execution needs to kick started from TestNG xml

PROBLEM STATEMENT

Need to automate the following activities using Selenium + Java.

Key Activities to implement:

Sl No.	Summary	Action	Expected Result
1	* Navigate to the URL. * Retrieve Title and URL of the Home page. * Verify Title & URL: Check if the title & URL matches the expected title.	1. go to URL : **https://healthapp.yaksha.com/** 2. login as valid credential **(username : admin , password : pass123)** and click on "**Sign in**" Button 3. get the title and URL of the Home page, post login 4. validate the title and URL of the Home page	Title should be : **DanpheHealth** Url should be : **https://healthapp.yaksha.com/Home/Index#/**
2	Ensure the Patient module is available and functional within the health app.	**Preconditions:** User must be logged into the health app. **Steps:** 1. **Confirm Module Presence:** Check if the Patient Module is listed in the left navigation. 2. **Expand Patient Module:** Click on the expand icon next to the Patient Module. 3. **Verify Sub-Modules:** Ensure the sub-modules "Search Patient" and "Register Patient" are visible and correctly labeled.	The Patient Module should be present and expandable. Sub-modules "Search Patient" and "Register Patient" should be correctly displayed under the Patient Module.

HealthApp Automation – Patient Module

3	Ensure that the 'Search textbox correctly displays its placeholder text as "Search (Minimum 3 Character)".	**Preconditions**: • User must be logged into the health app and on the 'Search Patient' page within the Patient module. **Steps**: 1. **Check for Textbox Presence**: Verify the presence of the 'Search textbox'. 2. **Click on Textbox**: Click on the textbox to focus on it. 3. **Retrieve Placeholder Text**: Obtain the placeholder text from the textbox. 4. **Confirm Placeholder Text**: Ensure the placeholder text matches "Search (Minimum 3 Character)".	The textbox should be present and accessible. Placeholder text should accurately read "Search (Minimum 3 Character)".
4	To confirm the functionality and navigational flow within the 'Register Patient' submodule, specifically for adding and managing patient photos.	**Preconditions**: User is logged into the healthApp system and on the "Search Patient" page of the Patient module. **Steps**: 1. **Navigate to Register Patient**: Click on the "Register Patient" submodule to access the "Basic Information" page. 2. **Access Photo Upload**: Click on the camera icon to start the photo upload process. 3. **Check for '+ New Photo' Button**: Verify the presence of the "+ New Photo" button. 4. **Initiate Photo Upload**: Click on the "+ New Photo" button. 5. **Verify 'Take a Snapshot' Button**: Check if the "Take a Snapshot" button appears after initiating the photo upload. 6. **Abort Process**: Click on the "Cancel" button to exit the photo upload interface.	Navigating to "Register Patient" should access the "Basic Information" page correctly. Buttons for adding and capturing a new photo should display and function properly, and clicking "Cancel" should revert the interface without issues.
5	To ensure that the system correctly prompts an error message when attempting to register a patient without filling out necessary information in the 'Basic Information' form.	**Preconditions**: User is logged into the health system and is on the profile picture form of the "Register Patient" submodule. **Steps**: 1. **Navigate to Basic Information**: Click on the "Basic Information" page link to access the form. 2. **Attempt to Register**: Without entering any data, directly click the "Register Patient" button. 3. **Observe Error Message**: Check for an error message that should appear due to the missing required information under **Phone number** textfield. 4. **Validate Error Message**: Verify the error message for the "Primary Phone" field, ensuring it states "Primary Phone is required."	Error Message should be **"Primary Phone is required"**

6	To ensure that data can be accurately entered into the 'Basic Information' form fields from an Excel source and that these entries are correctly displayed.	**Preconditions**: User is logged into the healthApp system and is on the 'Basic Information' form of the 'Register Patient' submodule. **Steps**: 1. **Data Entry**: Enter values into the form's textboxes. Data for 'First Name', 'Middle Name', 'Last Name', 'Age', and 'Phone Number' should be read from an Excel file and written into the corresponding fields. 2. **Validate Entered Data**: Check the 'First Name' textbox to ensure that the data entered matches the information sourced from Excel.	Each textbox ('First Name', 'Middle Name', 'Last Name', 'Age', 'Phone Number') should contain the correct information as provided by the Excel data. The 'First Name' field, in particular, should correctly display the entered data, confirming the accuracy of the input process.
7	To confirm that the 'Blood Group' dropdown is present and accessible at the bottom of the 'Basic Information' form within the 'Register Patient' submodule.	**Preconditions**: The user is logged into the healthApp system. The user is on the 'Basic Information' form within the 'Register Patient' submodule. **Steps**: 1. **Access Basic Information Form**: Ensure the user is correctly positioned on the 'Basic Information' form of the 'Register Patient' submodule. 2. **Scroll to Bottom**: Scroll to the bottom of the form to access additional fields. 3. **Verify Presence of Dropdown**: Check for the presence of the 'Blood Group' dropdown menu.	The 'Blood Group' dropdown should be clearly visible and accessible at the bottom of the 'Basic Information' form.
8	Highlight the 'Blood Group' dropdown.	**Preconditions**: The user is logged into the healthApp system. The user is at the bottom of the 'Basic Information' form in the 'Register Patient' submodule. **Steps**: 1. **Highlight Dropdown**: If the 'Blood Group' dropdown is present, apply a visual highlight (such as a border color change) to enhance its visibility.	"Blood Group" Dropdown should be present and Highlighted.
9	To ensure that the 'B-' blood group can be selected from the 'Blood Group' dropdown and that the selection is accurately reflected on the 'Basic Information' form of the 'Register Patient' submodule.	**Preconditions**: The user is logged into the healthApp system. The user is on the 'Basic Information' form within the 'Register Patient' submodule, specifically at the section containing the 'Blood Group' dropdown. **Steps**: 1. **Select 'B-' Option**: Click on the 'Blood Group' dropdown and choose the 'O+' option from the list. 2. **Verify Selection**: Confirm that 'B-' is properly selected and displayed as the current choice in the dropdown.	User should able to select "B-" group from the "Blood Group" dropdown menu.
10	To ensure that the 'Male' radio button within the Gender section of the 'Guarantor' page can be selected and then deselected,	**Preconditions**: The user is logged into the healthApp system. The user is at the bottom of the 'Basic Information' form within the 'Register Patient' submodule. **Steps**:	The 'Male' radio button should be easily selectable in the Gender section of the Guarantor form. After validation, the user should be able to deselect the 'Male' radio button.

HealthApp Automation – Patient Module

	confirming the functionality of radio button controls.	1. **Access Guarantor Page**: Click on the 'Guarantor' link to navigate to the Guarantor details page. 2. **Select 'Male' Radio Button**: On the 'Guarantor' page, select the 'Male' radio button in the Gender section. 3. **Verify Selection**: Confirm that the 'Male' radio button is indeed selected. 4. **Deselect Radio Button**: After confirming the selection, deselect the 'Male' radio button.	
11	Ensure that selecting the 'Self' checkbox on the 'Guarantor' form triggers a notification popup.	**Preconditions**: The user is logged into the healthApp system and is on the 'Guarantor' form within the 'Register Patient' submodule. **Steps**: 1. **Select 'Self' Checkbox**: Click on the 'Self' checkbox labeled as "My Self" and verify it is selected. 2. **Observe Notification Popup**: Upon selection, a notification popup should appear at the bottom of the page. 3. **Validate Notification Message**: Check that the notification popup displays the message: **(Notice-Message** **To fill the data by self checkbox, first you have to fill your address).**	"My Self" check box should be selected and Notification popup message should be **(Notice-Message** **To fill the data by self checkbox, first you have to fill your address)**
12	To verify that JavaScript can effectively navigate to the 'Kin/Emergency Contact' form from the 'Guarantor' form and input data into the 'Comments' textbox.	**Preconditions**: The user is logged into the health system and currently on the 'Guarantor' form within the 'Register Patient' submodule. **Steps**: 1. **Navigate via JavaScript**: Execute JavaScript to simulate a click on the "Kin/Emergency Contact" link, ensuring navigation to the corresponding form. 2. **Input Data via JavaScript**: Utilize JavaScript to input a predetermined value into the 'Comments' textbox on the 'Kin/Emergency Contact' form.	User should navigate to the "Kin/Emergency Contact" form and able to send the value to the "comments" textbox using JavaScript" operation

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.

HealthApp Automation – Patient Module

## 1.1 CODE QUALITY/OPTIMIZATIONS

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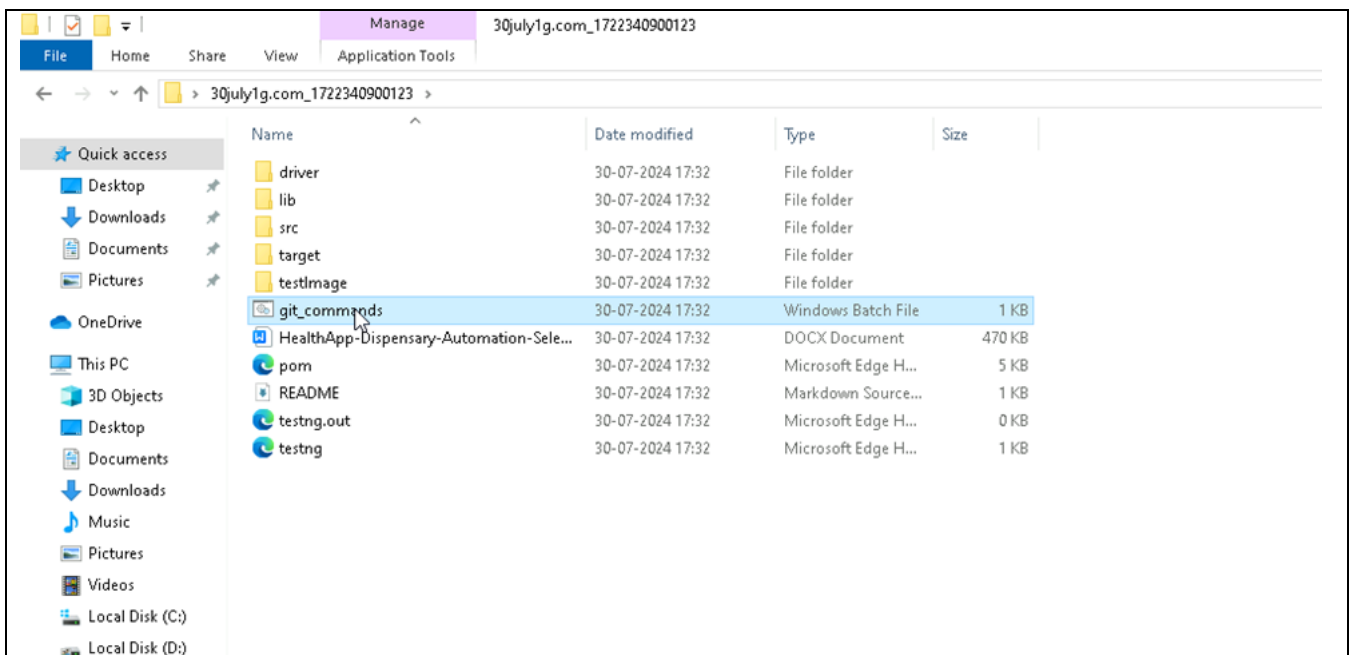2. Associates need to follow SOLID programming principles.

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.**
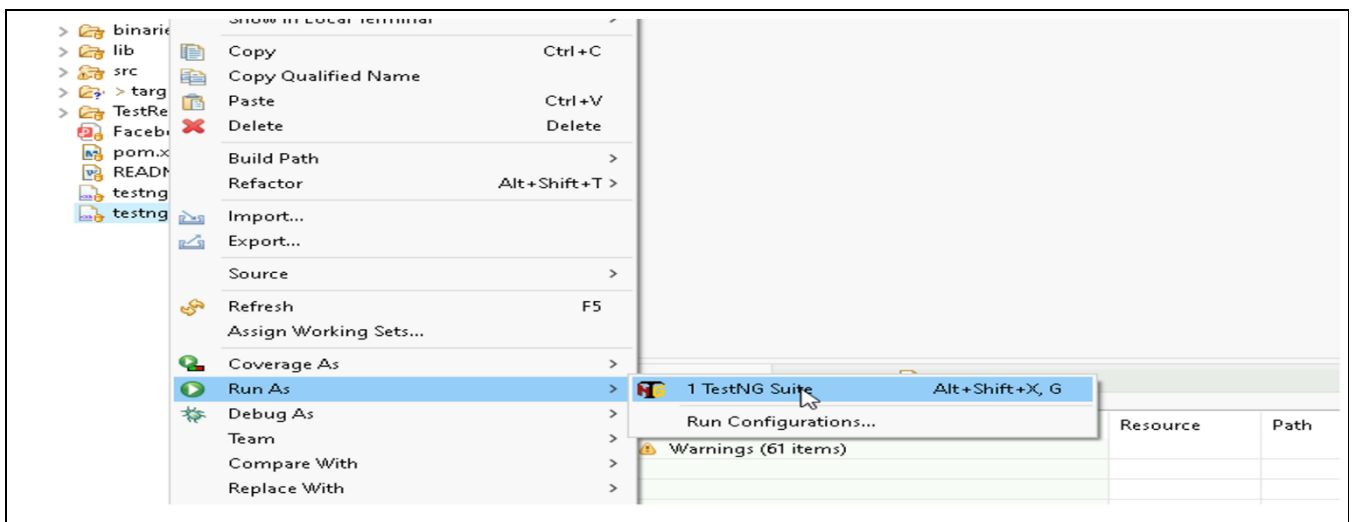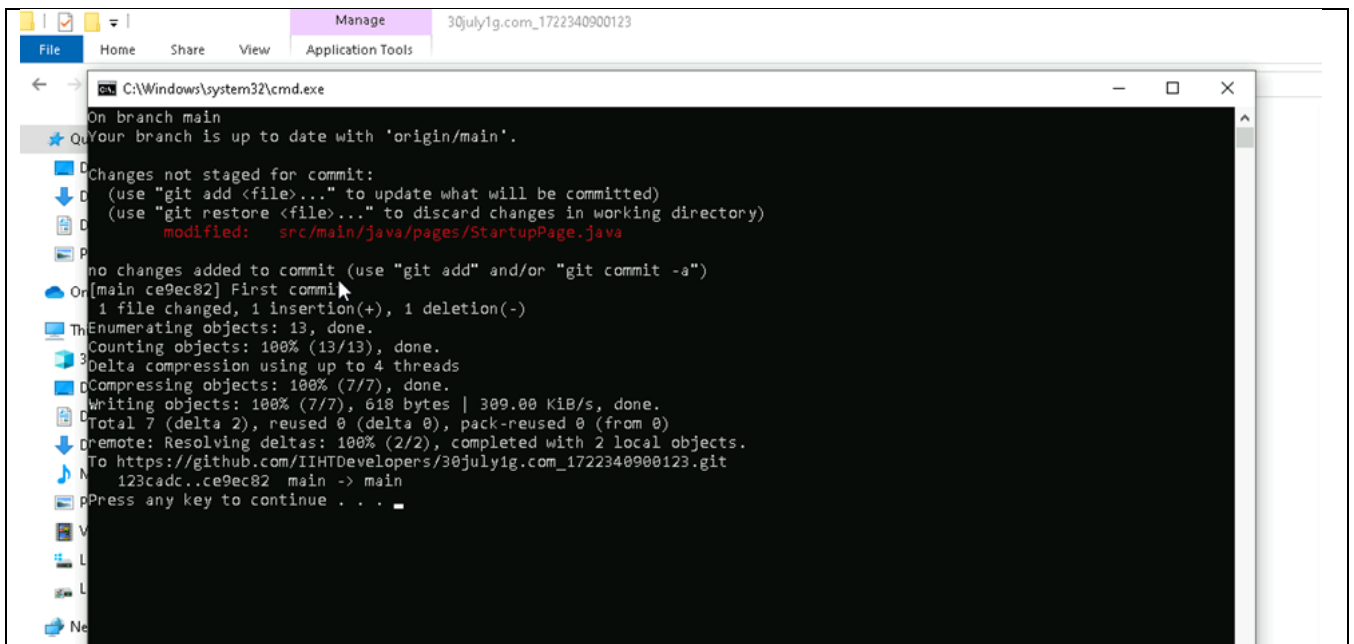

3. **Before final submission, you are also required to push your code to GIT. Following are the steps to follow:**

HealthApp Automation – Patient Module
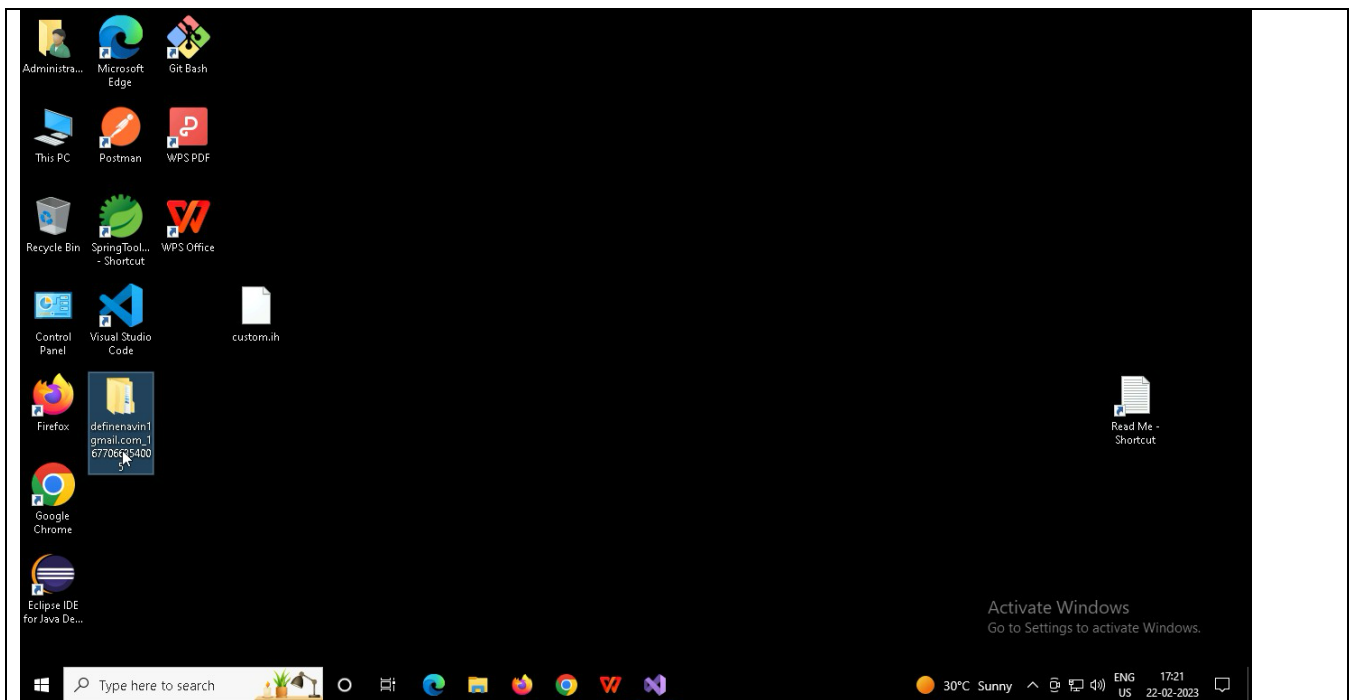
In your project folder, you will find a batch file named git_commands

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

HealthApp Automation – Patient Module

Once the code is pushed to git, you can go for the final submission of the assessment.
- Press escape to come out of Fullscreen mode.
- Submit the assessment.

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

HealthApp Automation – Patient Module

==

All the Best