

YAKSHA HEALTH APP WITH TYPESCRIPT AND PLAYWRIGHT

Mymedic automation using playwright

Usecase summary

Project Name: healthapp.yaksha app – Medical Record Management System

Use Case Summary: healthapp.yaksha is a healthcare application designed to manage Electronic Medical Records (EMR). It allows users to view, search, and manage patient records. It features functionality such as adding/editing patient records, filtering data by doctor and department, and exporting records. The primary use case is to automate the process of medical record management, ensuring efficient and reliable operations for healthcare providers.

Technology Stack:

- **Automation Tool:** Playwright (for testing)

Key Features:

- **Patient Record Management:** Add, edit, and delete patient records.
- **Filtering and Search:** Search medical records by date range, doctor, department, and more.
- **Export Functionality:** Export records for offline access.

Expected Outcomes:

- Automate key healthcare operations like patient record handling, filtering, and validation.
- Ensure the accurate retrieval and modification of medical records, enhancing operational efficiency.

Overview of the application

Pages/Features that are to be focused for the application

Please use the Application URL <https://healthapp.yaksha.com>

PROBLEM STATEMENT

Need to automate the following activities using playwright+typescript

You will be given few Json files in Data folder like PatientName.json and ValidLogin.json.

Path	File	Description
src\data	PatientName.json ValidLogin.json	1. Contains data to read from json file.
src\ pages	<ul style="list-style-type: none">• ADTPage• AdminPage• AppointmentPage• DashboardPage• DoctorPage• IncentivePage	<ol style="list-style-type: none">1. All core activities to be performed here.2. The comments associated with each templated method here describe the expectation.3. Declare any variable/object you

Mymedic automation using playwright

	<ul style="list-style-type: none"> • LoginPage • OperationTheaterPage • PatientPage • ProcurementPage • SettingsPage • SubstorePage • UtilitiesPage 	<p>need to share data/status between different methods.</p> <p>4. Do not modify the signature of methods declared here.</p>
--	--	---

Here's a detailed table format for the test cases to be tested

Test Case No.	Test Case Name	Test Steps to be performed	Path & Method Used	Expected Result
	Verify Login with Valid Credentials	<p>1.The application should read the data from ValidLogin.json file and fetch the user's name and password.</p> <p>2. It should call the method performLogin()</p> <p>3. Perform login method will perform authentication with the username and password.</p> <p>4. Verify admin name is visible on the home page.</p> <p>It is must to implement this login functionality at first and then implement any other test case.</p>	<p>Reference path</p> <p>\src\pages\LoginPage</p> <p>methods</p> <p>performLogin()</p> <p>You can use highlightElement method present in CommonMethods file to highlight the element before performing any action on it. It takes locator as a parameter.</p>	Successfully logs in with provided credentials. The user is logged in the admin page.
1	Verify the presence of Visit Type drop down by selecting "New patient" option	<p>1. Use verifyVisitTypeDropdown().</p> <p>2. Open "Appointment Booking List" tab inside Appointment module.</p> <p>3. Select any counter, if it's not selected already and then select "Appointment Booking List" tab.</p> <p>4. Select "New Patient" from "Visit Type" dropdown.</p> <p>5. Select "All Doctors" from "Doctor" dropdown.</p> <p>6. Pick "01-01-2024" in "From Date" field.</p> <p>7. Pick "31-03-2024" in "To Date" field.</p> <p>8. Click on "Show Patient" button.</p>	<p>Reference path</p> <p>\src\ pages\AppointmentPage</p> <p>methods</p> <p>verifyVisitTypeDropdown()</p>	The "Visit Type" column in list should contain only patients of "New" category.
2	Handle Alert for OT Booking Without Patient Selection	<p>1. Navigate to Operation theatre page.</p> <p>2. Use handleOtBookingAlert() to handle the alert.</p> <p>3. Click on "New OT Booking" button.</p> <p>4. Verify that the "Booking OT Schedule New Patient" modal is displayed.</p> <p>5. Without entering any details, within the modal, click on the "Add New OT" button.</p> <p>6. Handle the alert with message "Patient not Selected! Please Select the patient first!".</p>	<p>Reference path</p> <p>\src\ pages\OperationTheatrePage</p> <p>methods</p> <p>handleOtBookingAlert()</p>	<p>1. An alert with the message "Patient not Selected! Please Select the patient first!" is displayed.</p> <p>2. Handle and accept the alert to proceed.</p>

Mymedic automation using playwright

Test Case No.	Test Case Name	Test Steps to be performed	Path & Method Used	Expected Result
3	Verify Patient Overview Page Displays Information Correctly	<ol style="list-style-type: none"> 1. Use verifyPatientOverview(). 2. Read data from PatientName.json file for any one patient name. 3. Goto "Doctor" module, then "In Patient Department" tab. 4. In the search bar, enter the patient name read from patientName.json file and perform the search. 5. Locate the patient in the results and click on the "Preview" icon under the Actions column. 	Reference path \src\ pages\DoctorPage methods verifyPatientOverview()	Verify the same patient overview page is displayed with the same patient's name.
4	Add Progress Note for In Patient	<ol style="list-style-type: none"> 1. Use addProgressNoteForPatient() to check pop up message. 2. Go to "Doctor" module, then "In Patient Department" tab. 3. In the search bar, enter the patient's name read from patientName.json file and perform the search. 4. Locate the patient in the results and click on the "Preview" icon under the Actions column. 5. Click on "Notes" section. 6. Click on "Add Notes" button. 7. Select "Progress Note" option from "Template" dropdown. 8. Enter subjective Notes as "Test Notes" and click on save button. 	Reference path \src\ pages\DoctorPage methods addProgressNoteForPatient() You can use highlightElement method present in CommonMethods file to highlight the element before performing any action on it. It takes locator as a parameter.	The method should successfully add a Progress Note for the patient, and a success confirmation message with the text "Progress Note Template added." should be displayed.
5	Add and Verify New Currency in Settings	<ol style="list-style-type: none"> 1. Navigate to Procurement > Settings. 2. Select "Currency" sub tab. 3. Click "Add Currency" button. 4. Add any data in "Currency Code" and "Description" fields. 5. Click on "Add Currency" button. 	Reference path \src\pages\ProcurementPage methods addCurrencyAndVerify()	The new currency should be added successfully and displayed in the table with the correct currency code and description.
6	Verify Warning Popup for Mandatory Fields in Scheme Refund	<ol style="list-style-type: none"> 1. Navigate to Utilities module and select "Scheme Refund" tab. 2. If required, please select any counter value and then select "Scheme Refund" tab. 3. Click on "New scheme Refund Entry" button. 4. Now click on save without entering value in any field. 	Reference path \src\ pages\UtilitiesPage methods verifyMandatoryFieldsWarning() You can use highlightElement method present in CommonMethods file to highlight the element before performing any action on it. It takes locator as a parameter.	A warning popup should appear with the message: "Please fill all the mandatory fields."
7	Verify Navigation to User Profile Page	<ol style="list-style-type: none"> 1. Navigate to Homepage i.e https://healthapp.yaksha.com/Home/Index#/ 2. Click on the Admin dropdown. 3. Select the "My Profile" option. 	Reference path \src\ pages\AdminPage methods verifyUserProfileNavigation() You can use highlightElement method present in	Verify that the user is redirected to the "User Profile" page and the page header or title confirms this.

Test Case No.	Test Case Name	Test Steps to be performed	Path & Method Used	Expected Result
			CommonMethods file to highlight the element before performing any action on it. It takes locator as a parameter.	
8	Verify Patient Profile Picture Upload	1. Navigate to Patient module. 2. Select "Register Patient" tab. 3. Select "Profile Picture" tab (camera icon). 4. Click on the "New Photo" button. 5. Upload an image present in TestImage folder and click on the "Done" button.	Reference path \src\ pages\ PatientPage methods uploadProfilePicture()	Verify that the uploaded image is displayed successfully in the patient's profile.
9	Verify TDS Percent update for an employee	1. Navigate to the "Incentive" module and then "Settings" tab. 2. Click on the "Settings" tab. 3. Locate the row corresponding to the specified employee name. 4. Click the "Edit TDS%" button within the located row. 5. In the "Edit TDS Percent" modal, enter the updated TDS% value. 6. Click on the "Update TDS" button. 7. Verify that the updated TDS% value is correctly displayed in the table.	Reference path \src\pages\ IncentivePage methods editTDSForEmployee()	The updated TDS% value is displayed correctly in the corresponding row of the table.
10	Verify Price Category Enable/Disable	1. Navigate to "Settings" module. 2. Click on more... and select "Price Category" tab. 3. Click on "Disable" button to disable any Code in the table. 4. Verify a success message appears with the message "Deactivated." 5. Activate the same code by clicking "Activate" button and verify the success message as "Activated".	Reference path \src\pages\ SettingsPage methods togglePriceCategoryStatus()	A success message is displayed for both actions: "Deactivated." for disabling and "Activated." for enabling.
11	Verify Navigation Between Different Tabs	1. Navigate to "Substore" module. 2. Click on "Accounts" button. 3. Select "Inventory" tab. 4. Click and navigate between different tabs like "Stock", "Inventory Requisition", "Consumption", "Reports", "Patient Consumption" and "Return".	Reference path \src\ pages\ SubstorePage methods verifyNavigationBetweenSubmodules()	Ensure that it should navigate to each sub tab of the "Inventory" tab.
12	Verify tooltip text on hover in Inventory tab.	1. Navigate to "Substore" module. 2. Click on "Accounts" button. 3. Hover over a black coloured icon on top right side and capture the text.	Reference path \src\ pages\ SubstorePage methods verifyTooltipText()	Tooltip text should contain: "You are currently in Accounts sub store. To change, you can always click here."
13	TS-13 Capture screenshot of Inventory Requisition section.	1. Navigate to "Substore" module. 2. Click on "Accounts" button. 3. Select "Inventory" tab and then "Inventory Requisition" sub tab. 4. Capture a screenshot of the page and save it in the src/Screenshots folder.	Reference path \src\ pages\ SubstorePage methods captureInventoryRequisitionScreenshot()	Screenshot of the page is captured and saved successfully.

Test Case No.	Test Case Name	Test Steps to be performed	Path & Method Used	Expected Result
14	Verify to navigate to each section which are present in the "Inventory" sub-module	1. Navigate to ADT module. 2. Click on "Admitted Patients" tab. 3. Search for any patient name which data to be read form PatientName.json file. 4. Click on "..." button from table and select "Change Doctor". 5. Change doctor modal will open and then click on update button without filling any value.	Reference path \src\pages\ADTPage methods verifyFieldLevelErrorMessage()	Verify a field level error message appears "Select doctor from the list."

Learners will gain experience in building strongly-typed applications using React.js and managing data flow with **TypeScript**. They'll learn how to define interfaces, use types for error prevention, and improve code maintainability.

With **Playwright**, learners will learn to write and execute automated tests for the <https://healthapp.vaksha.com>

app. Key skills include:

- **Browser Automation:** Interacting with web elements and testing multiple browsers.
- **Assertions & Validations:** Ensuring app behaviour meets expected results.
- **End-to-End Testing:** Automating real user interactions and validating overall app functionality.

IMPLEMENTATION/FUNCTIONAL REQUIREMENT

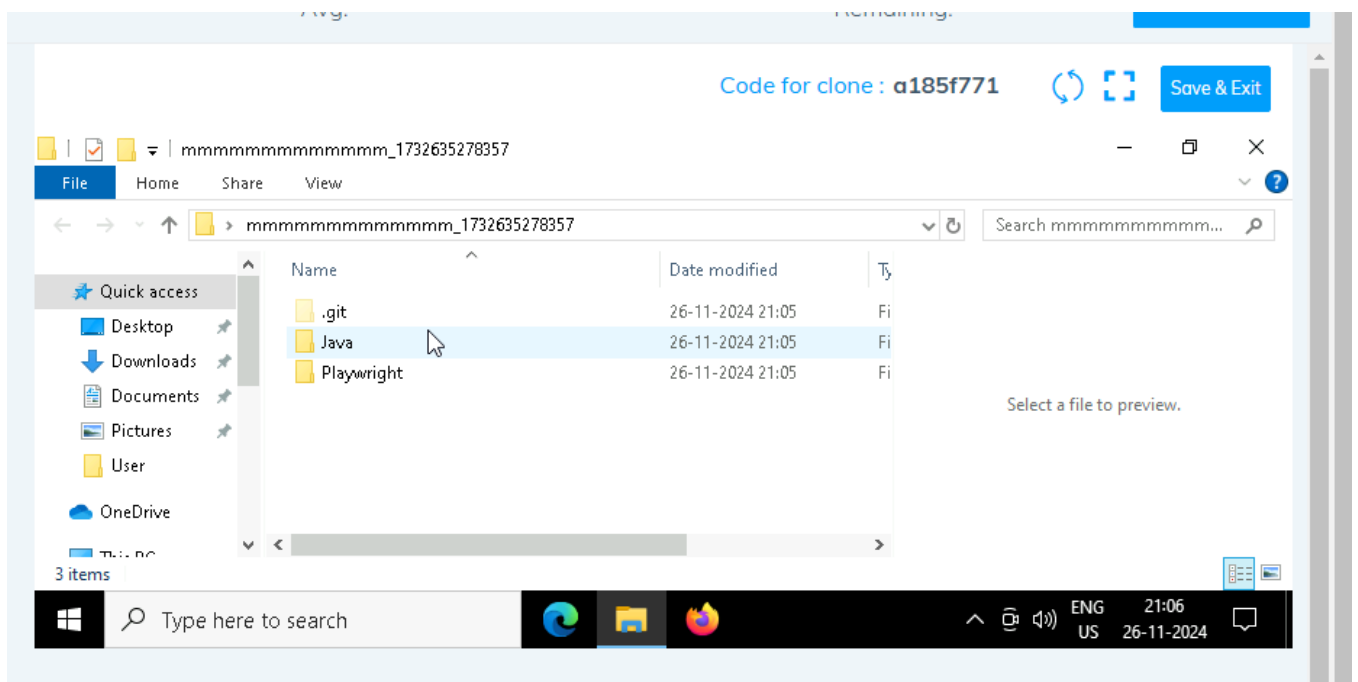
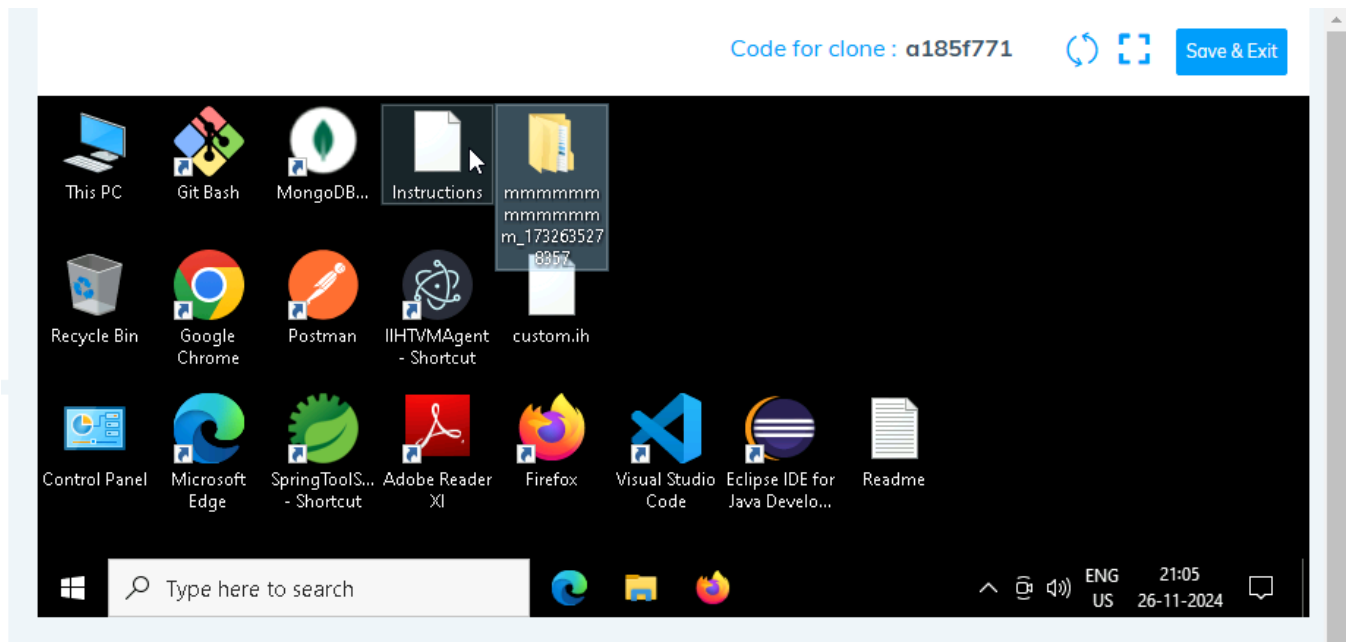
1.1 CODE QUALITY/OPTIMIZATIONS

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

Execution Steps:

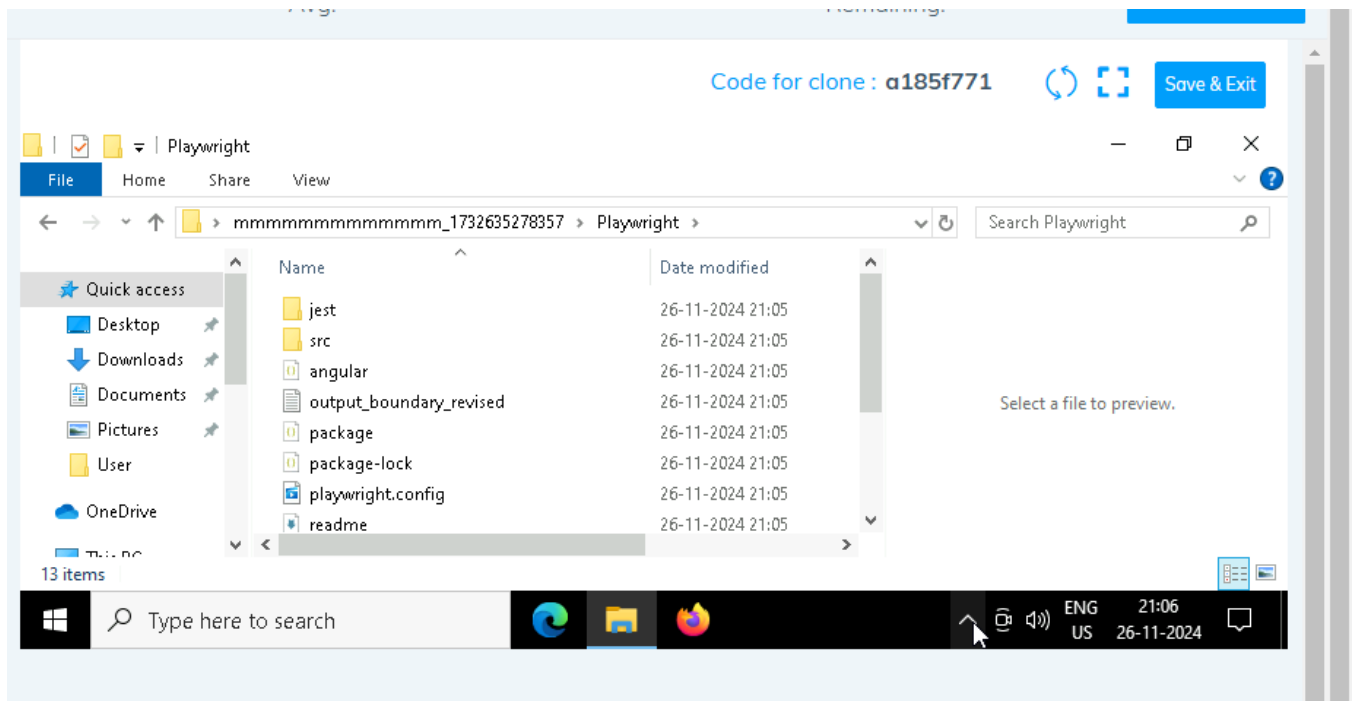
Steps for Execution:

1. Please open the folder created on desktop with the email name you used to login.



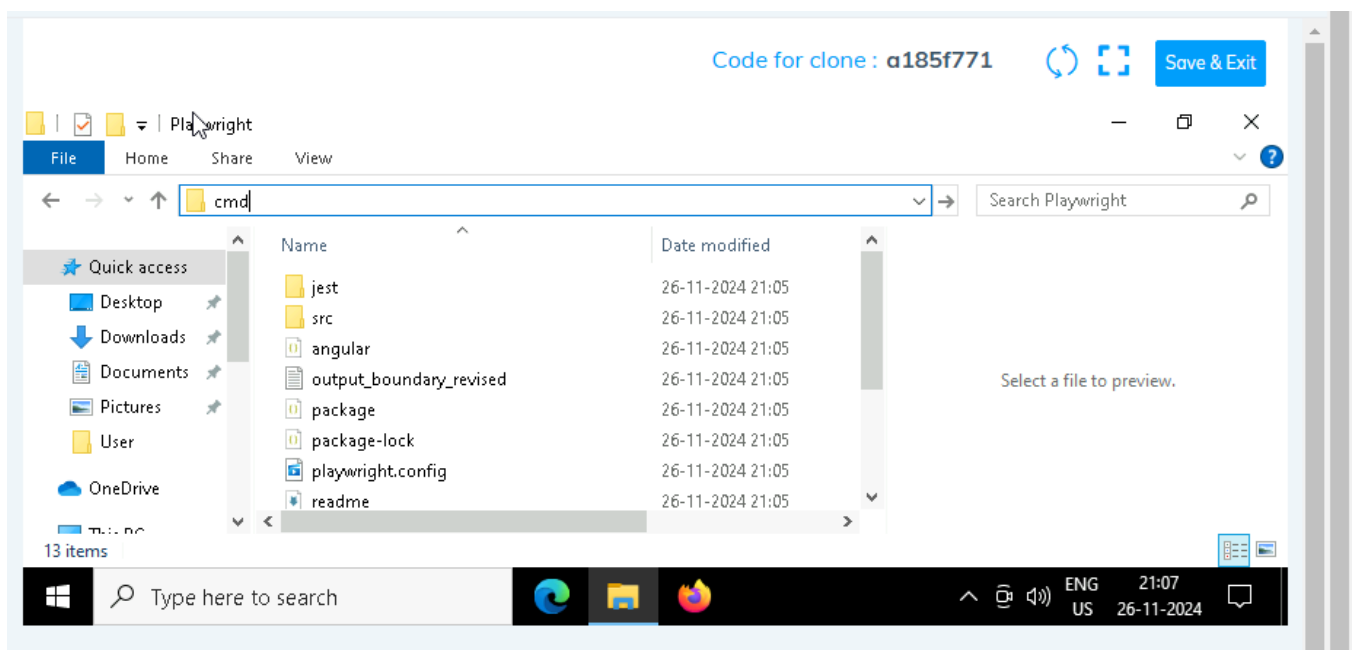
Mymedic automation using playwright

2. Go into the Playwright folder

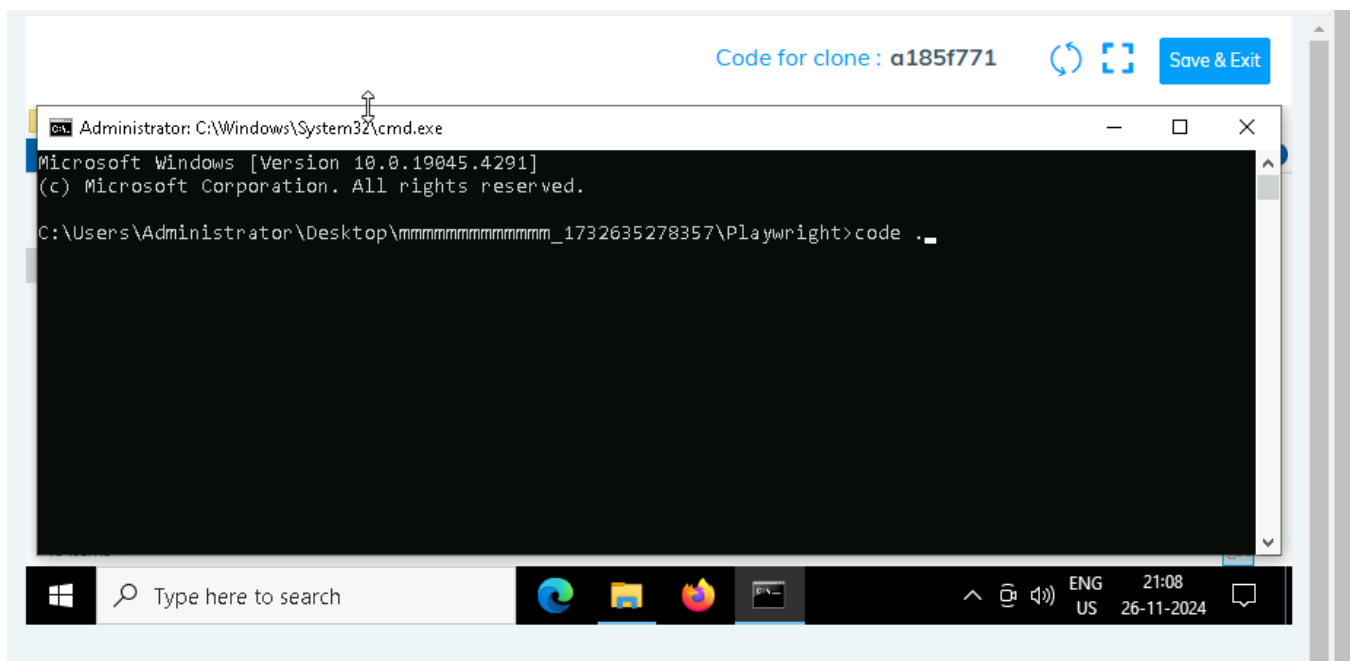
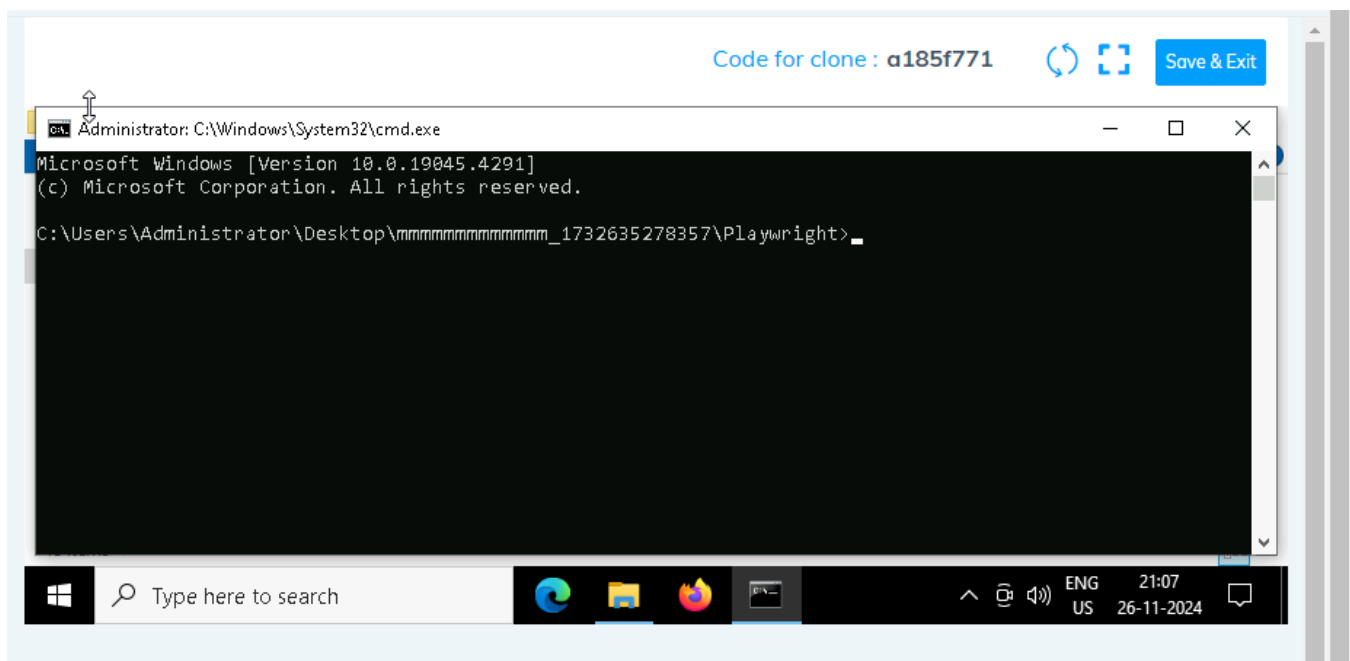


3. Open command prompt with its location and use below command:

code .

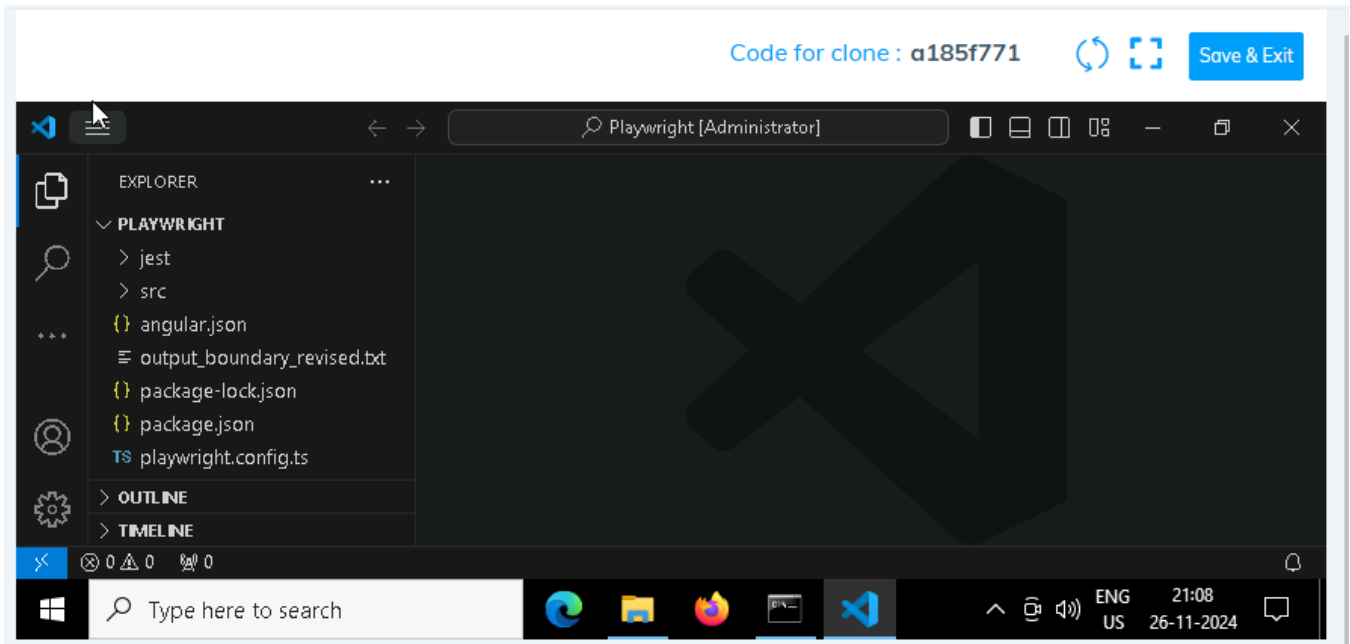


Mymedic automation using playwright



Mymedic automation using playwright

4. Once VsCode is open. Please open the terminal in Playwright folder:



5. Install all dependencies in the Playwright folder path using:

`npm install`

6. Install playwright in the Playwright folder path:

`npx playwright install`

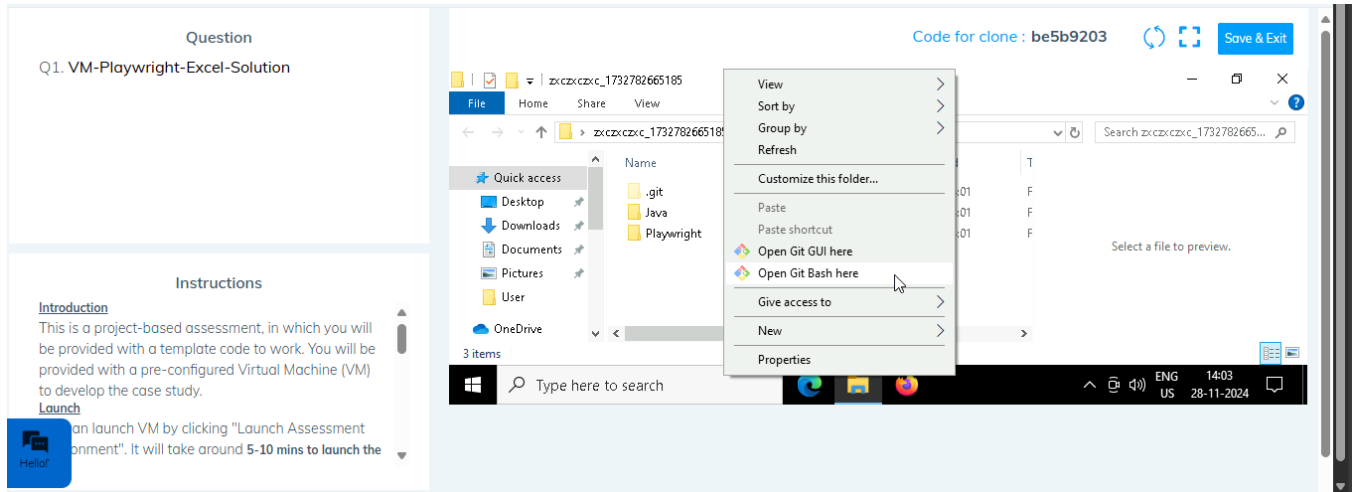
7. Run the Tests in the Playwright folder path:

`npx playwright test ./src/tests/PL1_testcases/yaksha.spec.ts`

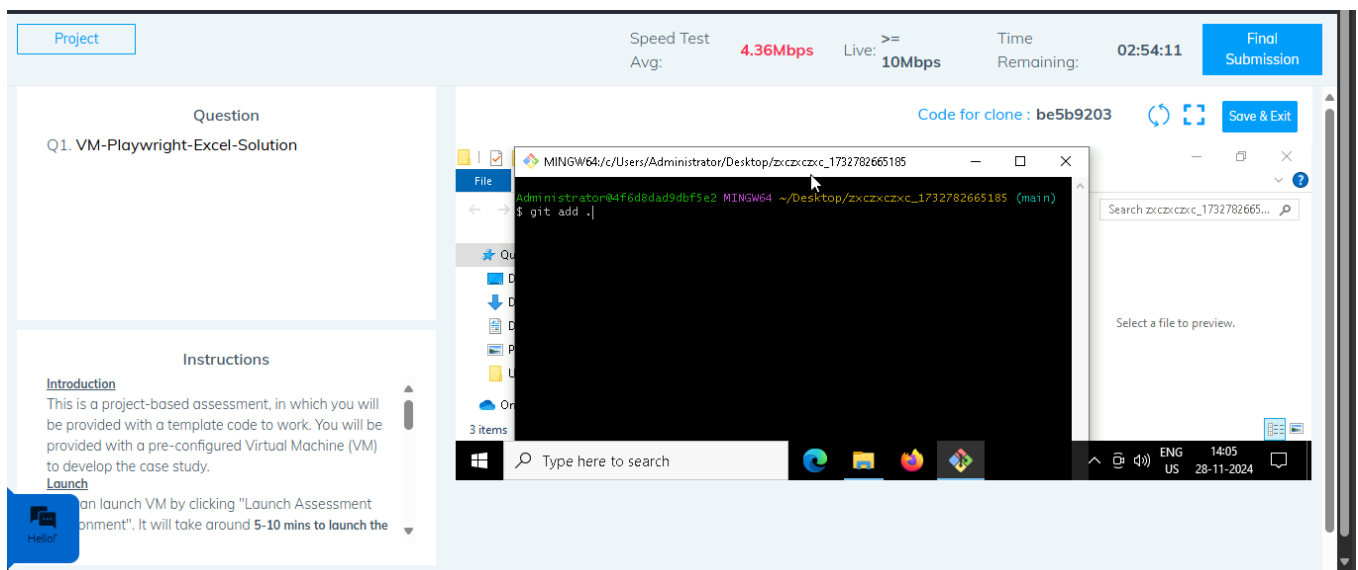
There is total 15 test cases.

8. Once you have executed the test cases. Now it is necessary to push your code to git. For this, please go inside the folder created on desktop with the email id you have used to login and then:

1. Open gitbash

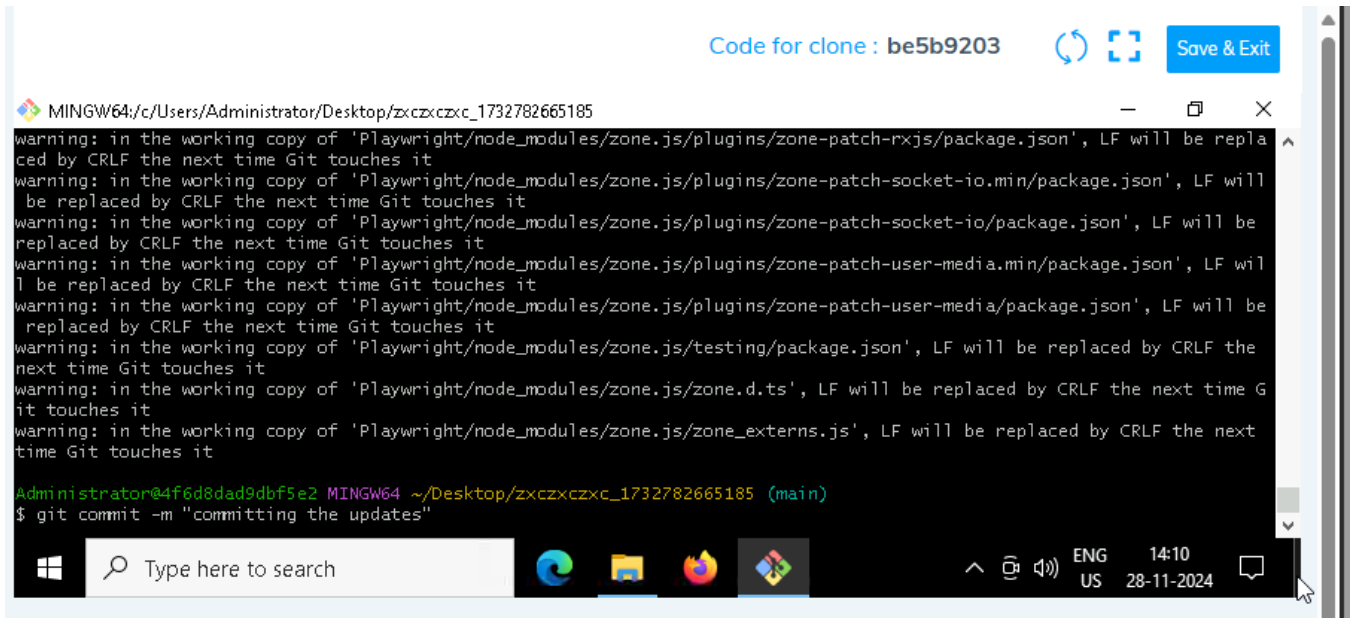


2. Add all files



Mymedic automation using playwright

3. Commit the changes



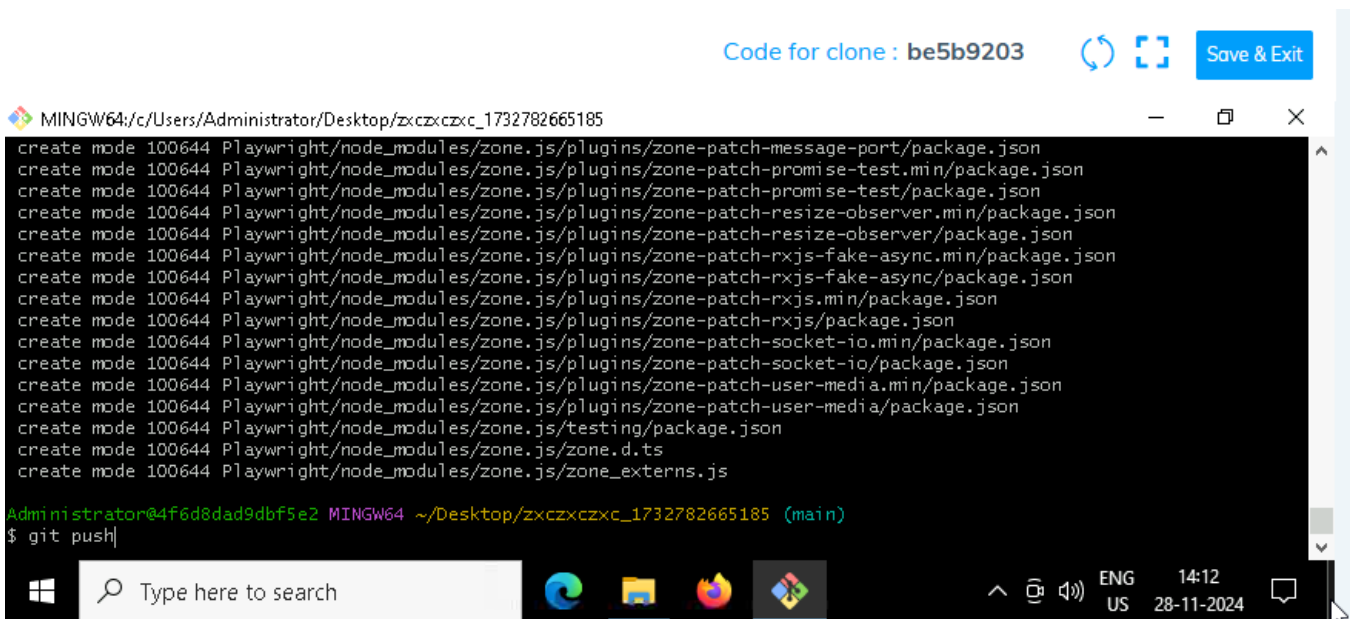
The screenshot shows a Windows terminal window with the title bar "MINGW64: c:/Users/Administrator/Desktop/zxczxczxc_1732782665185". The terminal output displays several warnings about CRLF line endings being replaced by LF in various files within the 'Playwright/node_modules/zone.js/plugins' directory. The user then enters the command `$ git commit -m "committing the updates"`. The terminal window has a taskbar at the bottom with the Windows logo, a search bar, and several application icons. The system tray shows the language as "ENG US" and the time as "14:10 28-11-2024".

```
Code for clone : be5b9203 [refresh] [copy] Save & Exit

MINGW64: c:/Users/Administrator/Desktop/zxczxczxc_1732782665185
warning: in the working copy of 'Playwright/node_modules/zone.js/plugins/zone-patch-rxjs/package.json', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'Playwright/node_modules/zone.js/plugins/zone-patch-socket-io.min/package.json', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'Playwright/node_modules/zone.js/plugins/zone-patch-socket-io/package.json', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'Playwright/node_modules/zone.js/plugins/zone-patch-user-media.min/package.json', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'Playwright/node_modules/zone.js/plugins/zone-patch-user-media/package.json', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'Playwright/node_modules/zone.js/testing/package.json', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'Playwright/node_modules/zone.js/zone.d.ts', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'Playwright/node_modules/zone.js/zone_extrns.js', LF will be replaced by CRLF the next time Git touches it

Administrator@4f6d8dad9dbf5e2 MINGW64 ~/Desktop/zxczxczxc_1732782665185 (main)
$ git commit -m "committing the updates"
```

4. Push the changes



The screenshot shows a Windows terminal window with the title bar "MINGW64: c:/Users/Administrator/Desktop/zxczxczxc_1732782665185". The terminal output displays a list of files being created, including various package.json files in the 'Playwright/node_modules/zone.js/plugins' directory. The user then enters the command `$ git push`. The terminal window has a taskbar at the bottom with the Windows logo, a search bar, and several application icons. The system tray shows the language as "ENG US" and the time as "14:12 28-11-2024".

```
Code for clone : be5b9203 [refresh] [copy] Save & Exit

MINGW64: c:/Users/Administrator/Desktop/zxczxczxc_1732782665185
create mode 100644 Playwright/node_modules/zone.js/plugins/zone-patch-message-port/package.json
create mode 100644 Playwright/node_modules/zone.js/plugins/zone-patch-promise-test.min/package.json
create mode 100644 Playwright/node_modules/zone.js/plugins/zone-patch-promise-test/package.json
create mode 100644 Playwright/node_modules/zone.js/plugins/zone-patch-resize-observer.min/package.json
create mode 100644 Playwright/node_modules/zone.js/plugins/zone-patch-resize-observer/package.json
create mode 100644 Playwright/node_modules/zone.js/plugins/zone-patch-rxjs-fake-async.min/package.json
create mode 100644 Playwright/node_modules/zone.js/plugins/zone-patch-rxjs-fake-async/package.json
create mode 100644 Playwright/node_modules/zone.js/plugins/zone-patch-rxjs.min/package.json
create mode 100644 Playwright/node_modules/zone.js/plugins/zone-patch-rxjs/package.json
create mode 100644 Playwright/node_modules/zone.js/plugins/zone-patch-socket-io.min/package.json
create mode 100644 Playwright/node_modules/zone.js/plugins/zone-patch-socket-io/package.json
create mode 100644 Playwright/node_modules/zone.js/plugins/zone-patch-user-media.min/package.json
create mode 100644 Playwright/node_modules/zone.js/plugins/zone-patch-user-media/package.json
create mode 100644 Playwright/node_modules/zone.js/testing/package.json
create mode 100644 Playwright/node_modules/zone.js/zone.d.ts
create mode 100644 Playwright/node_modules/zone.js/zone_extrns.js

Administrator@4f6d8dad9dbf5e2 MINGW64 ~/Desktop/zxczxczxc_1732782665185 (main)
$ git push
```