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 doctor.json, login.json, maternity.json, medicalRecord.json, pharmacy.json, radiology.json, settings.json and subStore.json.

Path	File	Description
src\data	doctor.json login.json maternity.json medicalRecord.json pharmacy.json radiology.json settings.json subStore.json	1. Contains data to read from json file.

Mymedic automation using playwright

src\ pages	● DoctorsPage ● LoginPage ● MaternityPage ● MedicalRecordsPage ● PharmacyPage ● RadiologyPage ● SettingsPage ● SubStorePage	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 need to share data/status between different methods. 4. Do not modify the signature of methods declared here.

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	1.The application should read the data from login.json file and fetch the user's name and password. 2. It should call the method performLogin() 3. Perform login method will perform authentication with the username and password. 4. Verify admin name is visible on the home page. **It is must to implement this login functionality at first and then implement any other test case.**	**Reference path** \src\pages**LoginPage** **methods** performLogin() **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.**	Successfully logs in with provided credentials. The user is logged in the admin page.
1	Handle Alert on Pharmacy Module	1. Navigate to "Pharmacy" module and select "Order" tab. 2. Click on "Add New Good Receipt" button. 3. Without adding any details, click on "Print Receipt" button. 4. 2 alert boxes should appear with messages as "Please select supplier" and "Please enter Invoice no.".	**Reference path** \src\ pages**PharmacyPage** **methods** handlingAlertOnRadiology() **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.**	Handles alerts during the Good Receipt print process, and ensures the modal is visible before performing further actions.
2	Verify to get the validation message when click on "Print Receipt" without filling any details	1. Navigate to "Pharmacy" module and select "Order" tab. 2. Click on "Add New Good Receipt" button. 3. Click on "Add New Item" button and fill details like "Item Name", "Batch no", "Item Qty" and "Rate" with data read from pharmacy.json file.	**Reference path** \src\ pages**PharmacyPage** **methods** handlingAlertOnRadiology()	Verify success message popup - "**Goods Receipt is Generated and Saved.**"

Mymedic automation using playwright

Test Case No.	Test Case Name	Test Steps to be performed	Path & Method Used	Expected Result
		4. Add details to "Supplier Name" and "Invoice" as data read data from pharmacy.json file and any 3-digit random number like "777" respectively. 5. Click on "Print Receipt" button. 6. Verify the success message in pop-up at bottom right side as "**Goods Receipt is Generated and Saved.**"	**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.**	
3	Verify to data range by select "Last 3 months" option from drop down	1. Use verifyDataWithinLastThreeMonths(). 2. Navigate to "Radiology" module. 3. Select "List Requests" tab. 4. Click on "-" button (present at left side of "OK" button) and select "Last 3 Months" option. 5. Click "Ok" button.	**Reference path** \src\ pages**RadiologyPage** **methods** verifyDataWithinLastThreeMonths() **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.**	Data should be present as per the selected date range using dropdown The 'Requested on' column date must fall within the "Last 3 months".
4	Verify that entering a keyword matching existing records in the search bar returns the corresponding data.	1. Navigate to "Medical Records" module. 2. Select "MR Outpatient List" tab. 3. Enter data to be read from medicalRecord.json file in "From" field. 4. Click on "Ok" button. 5. Enter gender data to be read from medicalRecord.json file in search field.	**Reference path** \src\ pages**MedicalRecordsPage** **methods** keywordMatching() **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.**	Data should be present for the search, and the 'Gender' column should contain only patients in the "Female" category.
5	Verify the presence of "Doctor filter" drop down by selecting "Dr. ALEX OKELLO ONYIEGO" option.	1. Navigate to "Medical Records" module. 2. Select "MR Outpatient List" tab. 3. Enter from data to be read from medicalRecords.json file in "From" field. 4. Click on "Ok" button. 5. Select doctor name to be read from medicalRecords.json file in doctor filter drop down.	**Reference path** \src\pages**MedicalRecordsPage** **methods** presenceOfDoctorFilter() **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.**	Data should be present as per the selected doctor from the dropdown the 'Doctor Name' column only the selected doctor name.
6	Add and Verify a New Dynamic Template in Settings Module.	1. Navigate to "Settings" module and select "Dynamic Templates" tab. 2. Click on "Add Template" button. 3. Add template modal opens and fill all the mandatory details like Template Type, Template Name, Template Code to be read from settings.json file.	**Reference path** \src\pages**SettingsPage** **methods** verifyDynamicTemplates()	A success confirmation popup with the message: "**Template Saved.**" should appear, indicating that the template has been added successfully.

Mymedic automation using playwright

Test Case No.	Test Case Name	Test Steps to be performed	Path & Method Used	Expected Result
		4. Read TextFiled data from settings.json file to place in text area field. 5. Click on "Add" button.	**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.**	
7	Create and Verify Inventory Requisition in Substore Module	1. Navigate to "Substore" module. 2. Click on the "Account" button and then "Inventory" tab. 3. Click on "Inventory Requisition" tab. 4. Click on "Create Requisition" button. 5. Fill requisition details: 6. Select "Target Inventory" and "Item name" data from substore.json file. 7. Enter required quantity as "1". 8. Click on "Request" button.	**Reference path** \src\ pages**SubstorePage** **methods** createInventoryRequisition() **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 success confirmation popup with the message: "**Requisition is Generated and Saved**" should appear.
8	Verify maternity allowance report is visible.	1. Navigate to "Maternity" module. 2. Click on "Reports" tab. 3. Click on "MaternityAllowance" button. 4. Enter date in "From'" field to be read from maternity.json file and click on "Show Report" button.	**Reference path** \src\ pages**MaternityPage** **methods** verifyMaternityAllowanceReport() **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.**	Report should be visible when click on show more button.
9	Verify imaging and lab order add successfully.	1. Navigate to the "Doctor" module and then "In Patient Department" tab. 2. Click on the search bar and search patient with filed name as patientName read from doctor.json file. 3. Click on "Imaging" icon under the action column. 4. Select "imaging" option from drop down in "New Order" section. 5. Select searchOrderItem data read from doctor.json file in the search order item. 6. Click on "Proceed" button. 7. Click on "Sign" button.	**Reference path** \src\pages**DoctorPage** **methods** performInpatientImagingOrder() **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 success confirmation popup with the message: "**Imaging and lab order add successfully**" should appear.
10	Verify to filter the records by select "X-RAY" from Filter drop down.	1. Navigate to "Radiology" module. 2. Read "filter" data from radiology.json file and select that option from "Filter" drop down. 3. Read "from" and "to" data from radiology.json file and put in "From" and add "To" fields. 4. Click on "Ok" button.	**Reference path** \src\pages**RadiologyPage** **methods** filterListRequestsByDateAndType()	Record should filter out as per status.

Mymedic automation using playwright

Test Case No.	Test Case Name	Test Steps to be performed	Path & Method Used	Expected Result
			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.	

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.yaksha.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
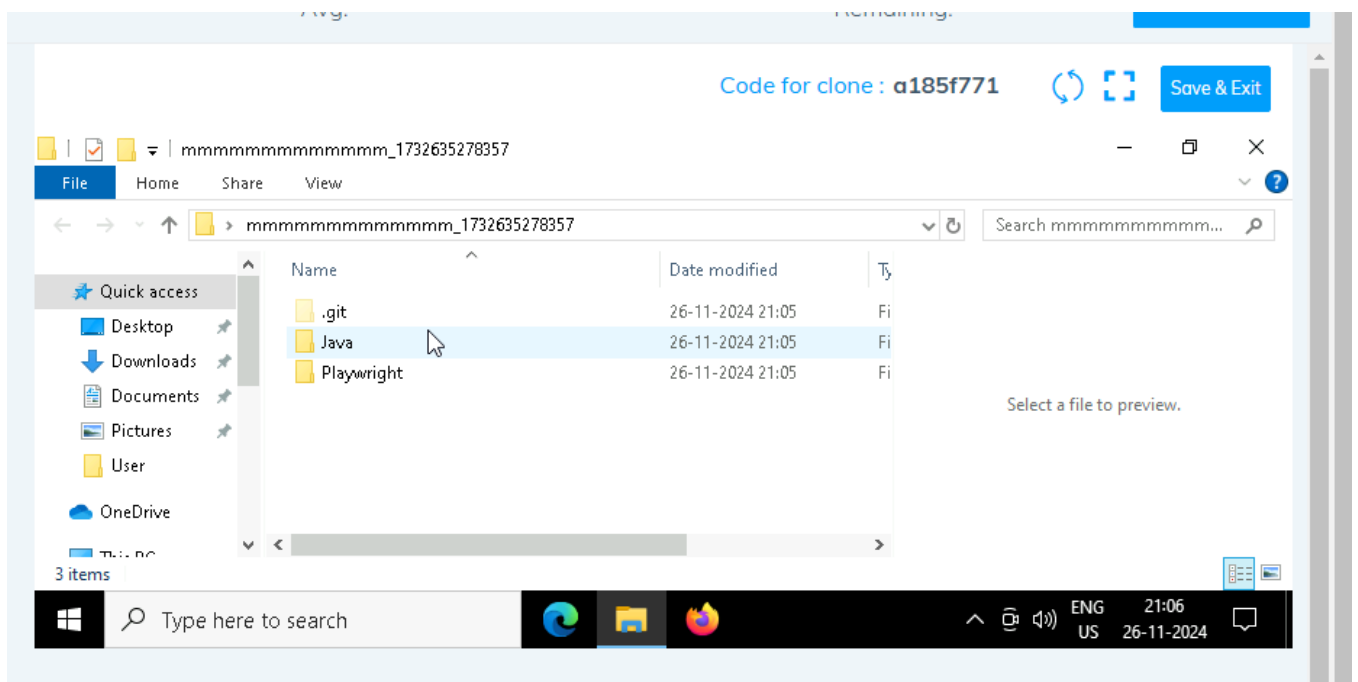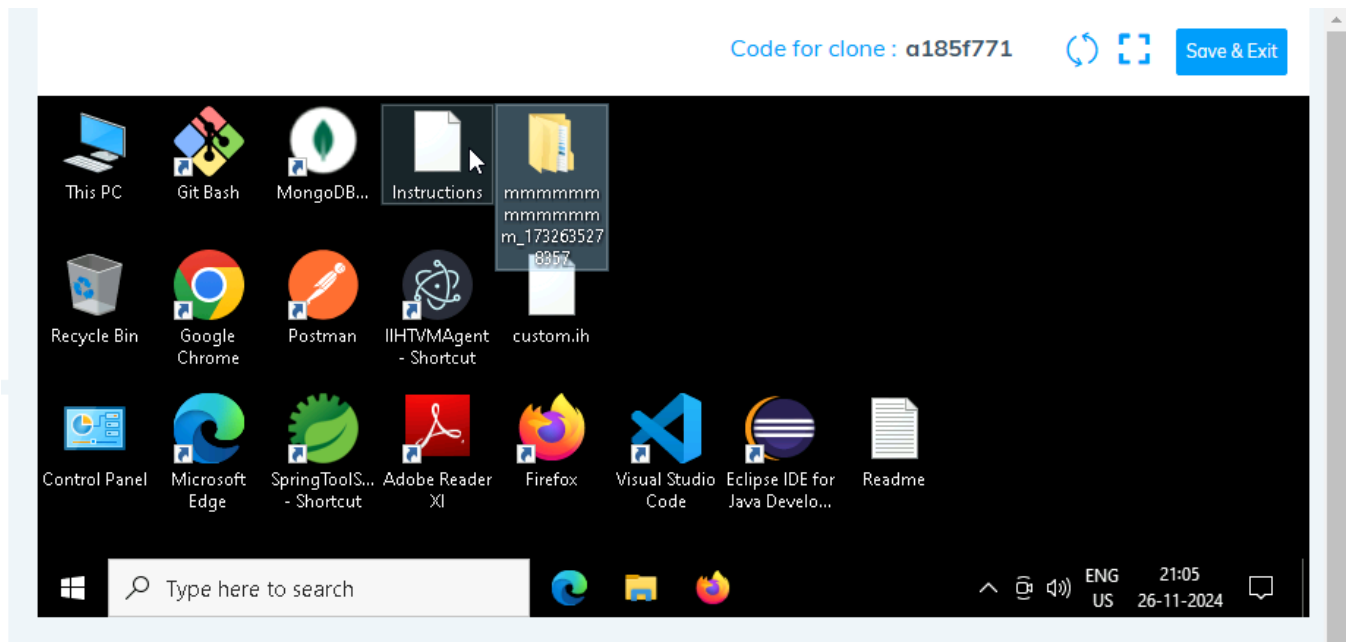
Mymedic automation using playwright

2. Go into the Playwright folder
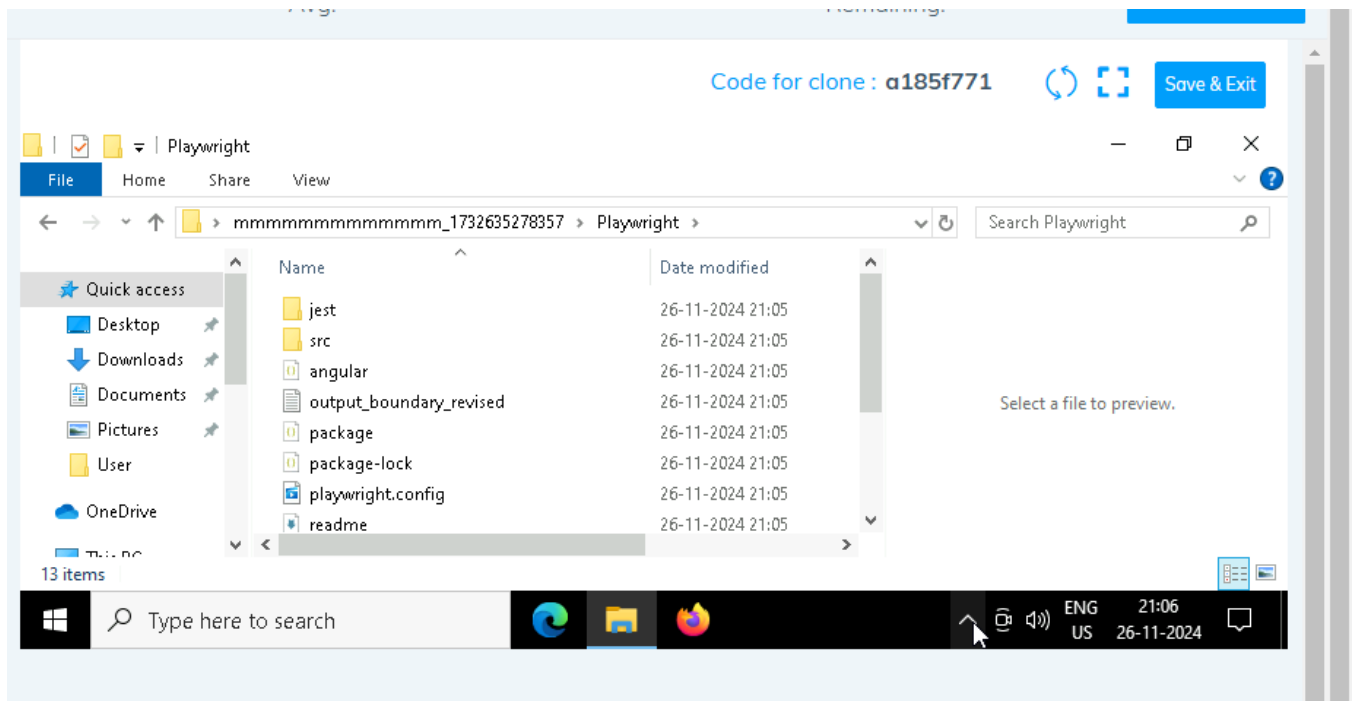
3. Open command prompt with its location and use below command:

    code .
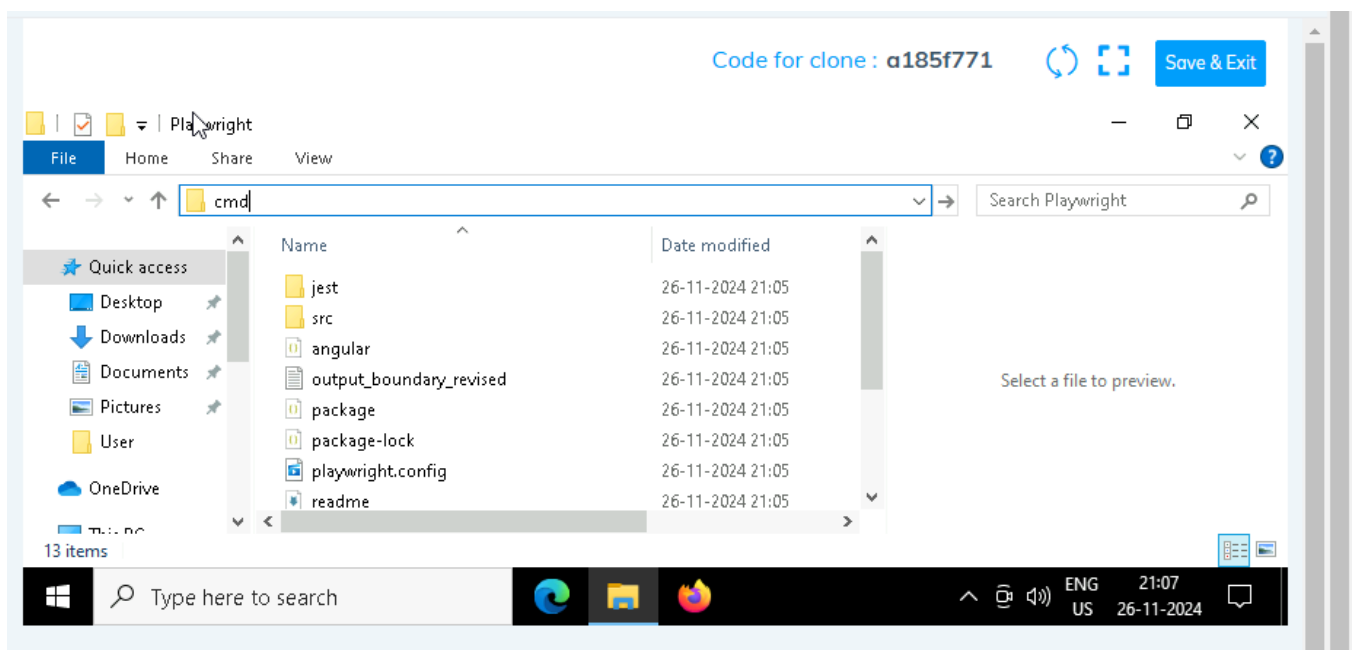
Mymedic automation using playwright
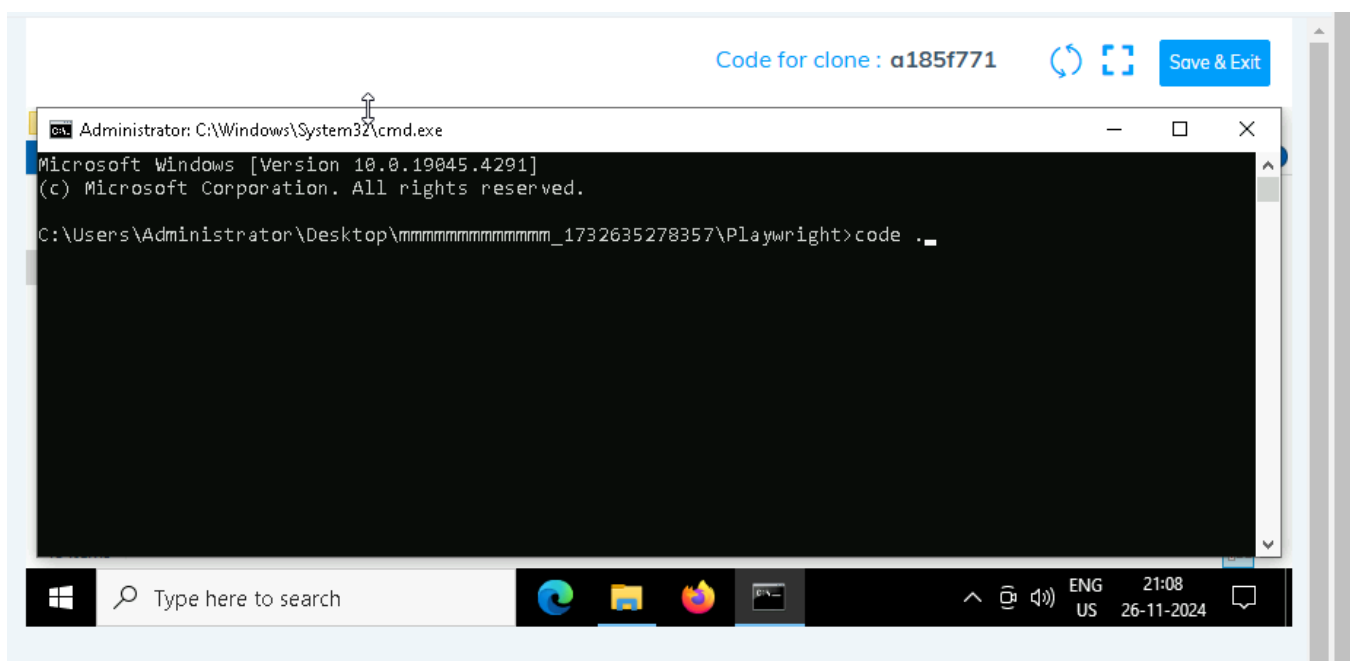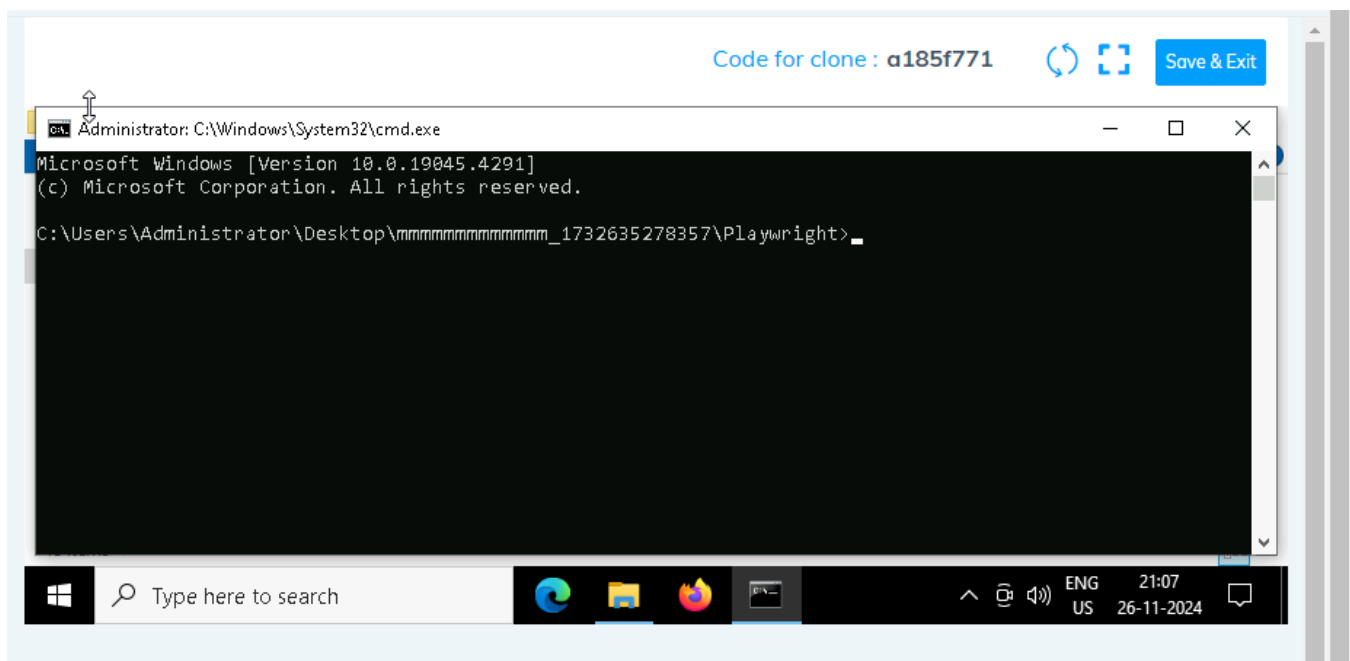
Mymedic automation using playwright

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

Code for clone : a185f771   Save & Exit

EXPLORER   Playwright [Administrator]

∨ **PLAYWRIGHT**
  > jest
  > src
  {} angular.json
  ≡ output_boundary_revised.txt
  {} package-lock.json
  {} package.json
  TS playwright.config.ts

> OUTLINE
> TIMELINE

⊗ 0 ⚠ 0   🔊 0

Type here to search

ENG   21:08
US   26-11-2024
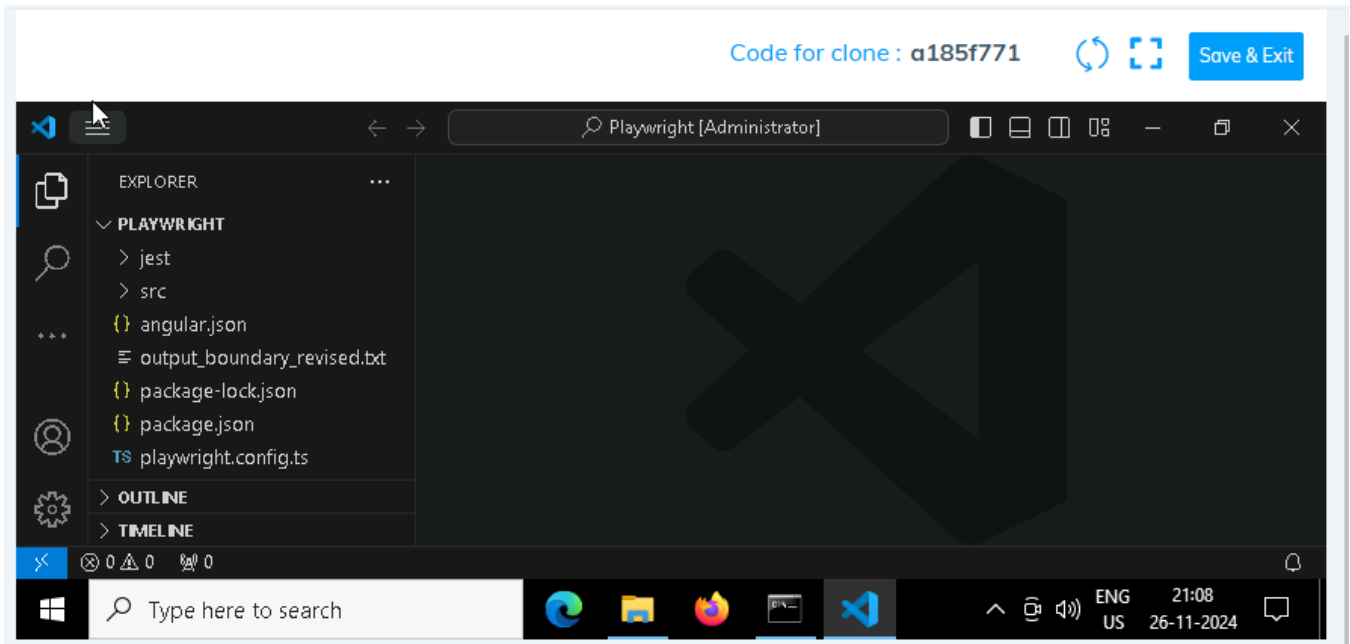
5. **Install all dependencies in the Playwright folder path using:**

   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

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:**

Mymedic automation using playwright

1. **Open gitbash**
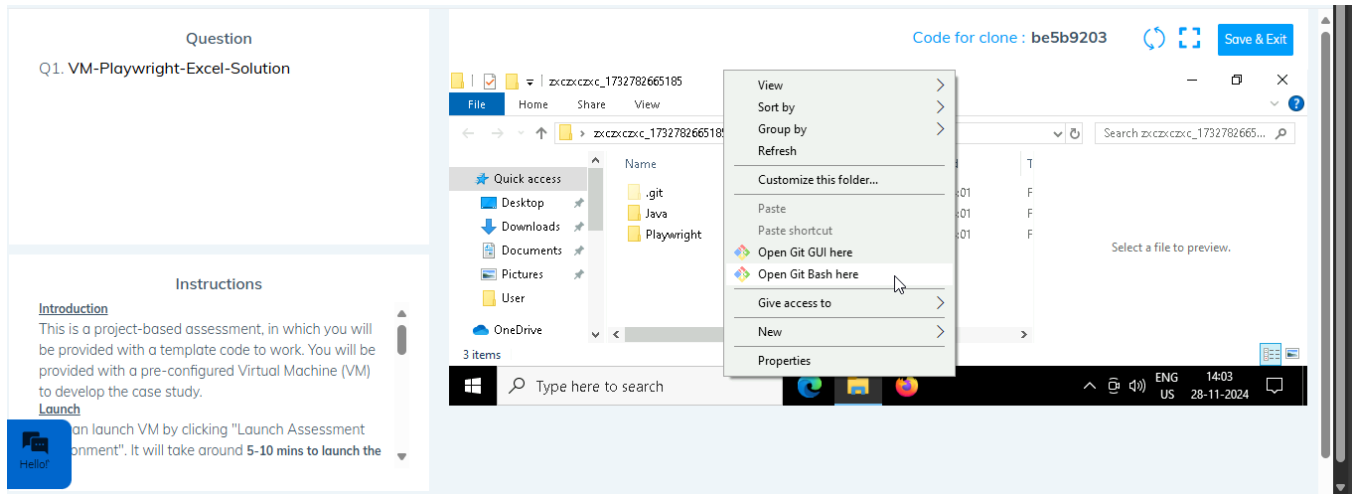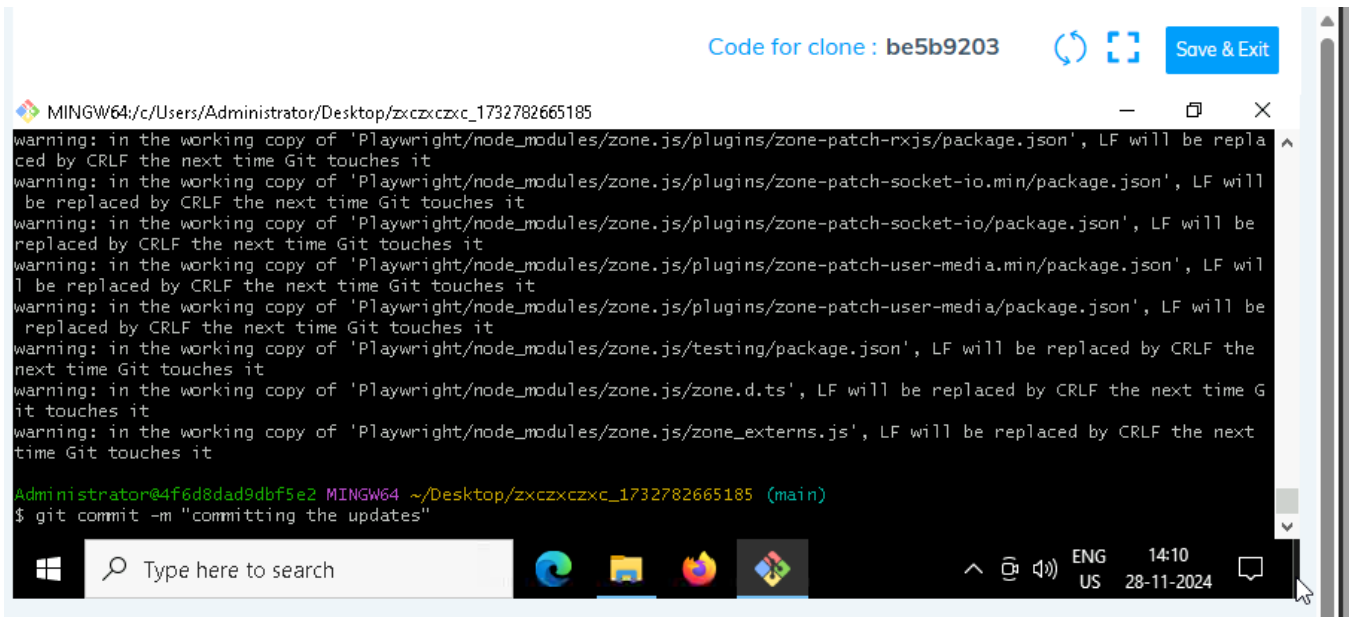
2. **Add all files**
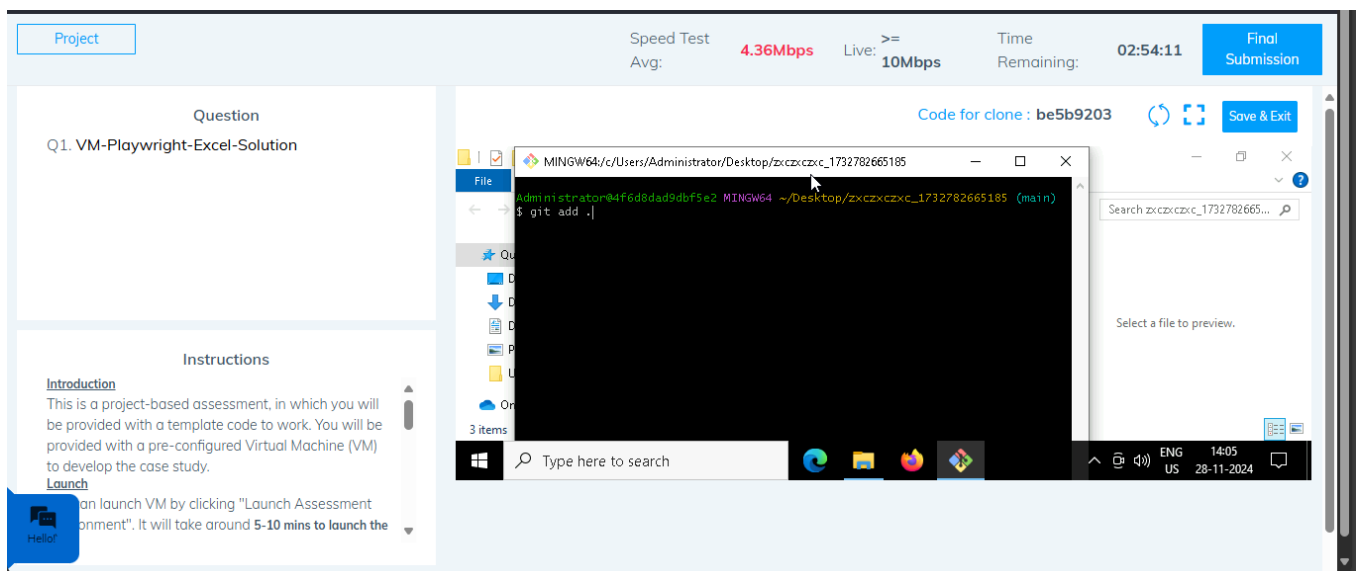
Mymedic automation using playwright

3. **Commit the changes**

4. **Push the changes**

Mymedic automation using playwright