

YAKSHA HEALTH APP WITH TYPESCRIPT AND PLAYWRIGHT PL1-8

Usecase summary

Project Name:opensource-demo.orangehrmlive.com app – HR management system.

Use Case Summary: opensource-demo.orangehrmlive.com is designed to manage Employee record.

Records (HRMS). It allows users to view, search, and manage employee records. It features functionality such as adding/editing employee details, filtering data by department and role, and exporting records. The primary use case is to automate the process of employee data management, ensuring efficient and reliable operations for human resources teams in organizations

Technology Stack:

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

Key Features:

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

Expected Outcomes:

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

Overview of the application

Pages/Features that are to be focused for the application

Please use the Application URL
<https://opensource-demo.orangehrmlive.com>

PROBLEM STATEMENT

Need to automate the following activities using playwright+typescript

You will be given few Json files in Data folder like Login.json and TestData.Json

Path	File	Description
src\data	login.json TestData.ts	1. Contains data to read from a json file.

src\ pages	<ul style="list-style-type: none"> • DependentsPage • LoginPage • MyInfoPage 	<ol style="list-style-type: none"> 1. All core activities are 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
1	Verify the 'My Info' tab Loads Successfully	<ol style="list-style-type: none"> 1. Login using valid credentials 2. Click on "My Info" tab 	Reference path \src\pages\MyInfoPage METHODS: clickMyInfoTab()	Verify the My Info tab loads successfully
2	Verify the 'Required' field error Message is displayed on leaving name field blank	<ol style="list-style-type: none"> 1. Login using valid credentials 2. Click on "My Info" tab 3. Clear the name present in the name inputbar 	Reference path \src\pages\MyInfoPage METHODS: clearAndEnterName()	Required' field error message should be displayed
3	Verify the name gets edited successfully	<ol style="list-style-type: none"> 1. Login using valid credentials 2. Click on "My Info" tab 3. Clear the name present in the name inputbar 4. Enter the new name and hit save button 5. Hit refresh 	Reference path \src\pages\MyInfoPage METHODS: updateUniqueNAmeAndVerifyName() CommonMethods file to highlight the element before performing any action on it. It takes locator as a parameter.	Verify the name gets updated in the top right corner

4	Verify the 'My Info' tab's subtabs have unique URL	1. Login using valid credentials 2. Click on "My Info" tab 3. Click on top 3 items from the sidebar('personal details',contact details,emergency contact')	Reference path \src\pages\DepedentsPage METHODS: areMyInfoSubTabHrefsUnique()	Verify the url from the three is unique
5	Verify new Dependant could be added to the list	1. Login using valid credentials 2. Click on "My Info" tab 3. Navigate to 'Dependents' subtab 4. Fill name ,select 'other' from relationship 5. enter the name of other relationship	Reference path \src\pages\DepedentsPage METHODS: addDependent()	Verify the presence of the child in the list
6	Verify new Dependant 'Specify' inputbar only displays when 'Other' option is selected from the relationship dropdown	1. Login using valid credentials 2. Click on "My Info" tab 3. Navigate to 'Dependents' subtab 4. Fill name ,select 'other' from relationship 5. enter the name of other relationship	Reference path \src\pages\DependedntsPage METHODS: selectOtherAndCheckSpecifyField()	Verify the specify inputbar is not prsent before entering the other in relationship inputbar
7	Verify the dependants could be edited from the list	1. Login using valid credentials 2. Click on "My Info" tab 3. Navigate to 'Dependents' subtab 4. Create a new dependant using porevious steps 5. click the edit icon and change the name of the dependant	Reference path \src\pages\DepedentsPage METHODS: editDependentNameFlow()	Verify the dependant details get updated

8	Verify the dependants could be deleted from the list	1. Login using valid credentials 2. Click on "My Info" tab 3. Navigate to 'Dependents' subtab 4. Create a new dependant using previous steps 5. click the delete icon and verify the candidate gets deleted from the list	Reference path \src\pages\DependentsPage METHOD: deleteDependentFlow()	Verify the dependant details get deleted

9	Verify the sample pdf file could be uploaded in the attachments	1. Login using valid credentials 2. Click on "My Info" tab 3. Navigate to 'Dependents' subtab 4. Click on 'Add' Attachments button 5. Browse the pdf from the device 6. Click save button	Reference path \src\pages\DependentPage METHODS: UploadAttachmentInDependent()	Verify the pdf gets uploaded
10	Verify the Comment inputbar has limit on length	1. Login using valid credentials 2. Click on "My Info" tab 3. Navigate to 'Dependents' subtab 4. Click on 'Add' Attachments button 5. Click Comments inputbar and add really long tex	Reference path \src\pages\DependentPage method CommentBarInputLimit ()	verify the 'Should not exceed 200 characters' is displayed

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. URL <https://opensource-demo.orangehrmlive.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

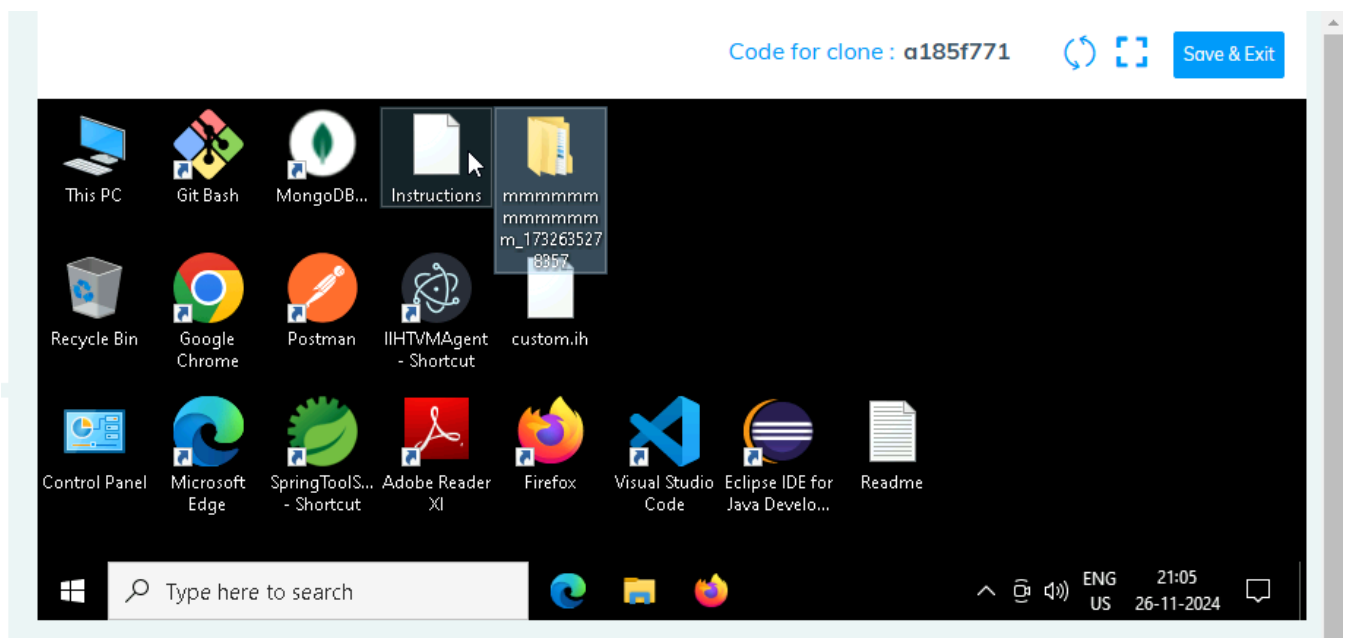
1.1 CODE QUALITY/OPTIMIZATIONS

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

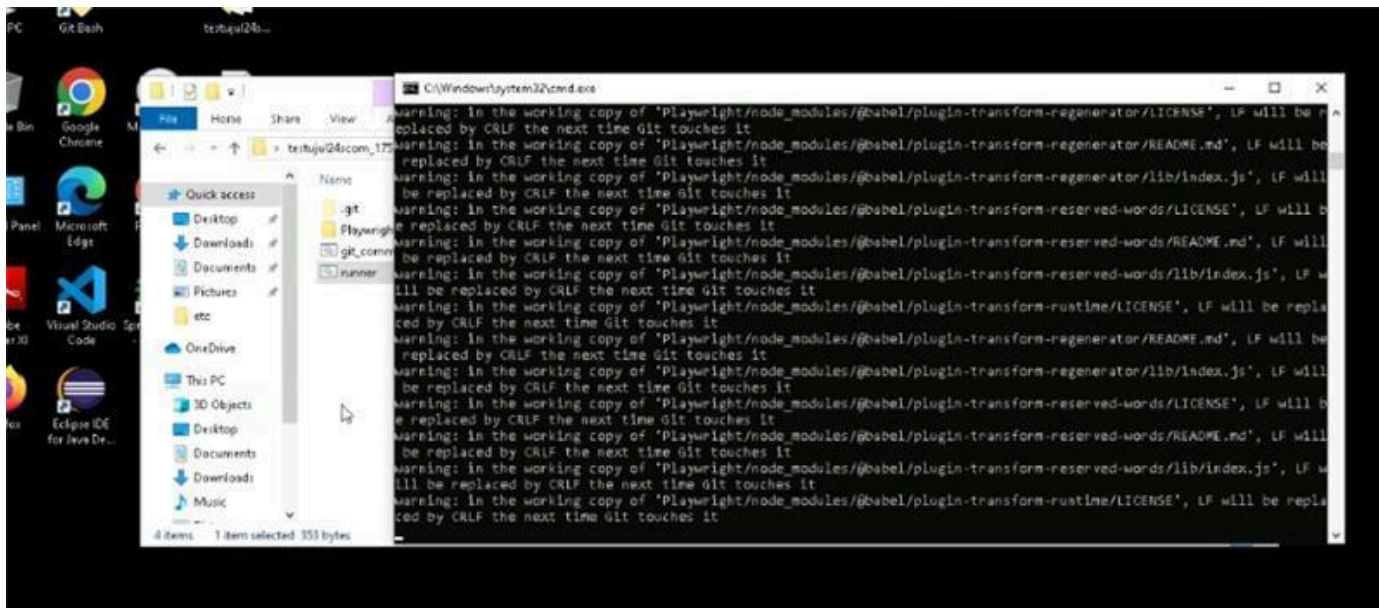
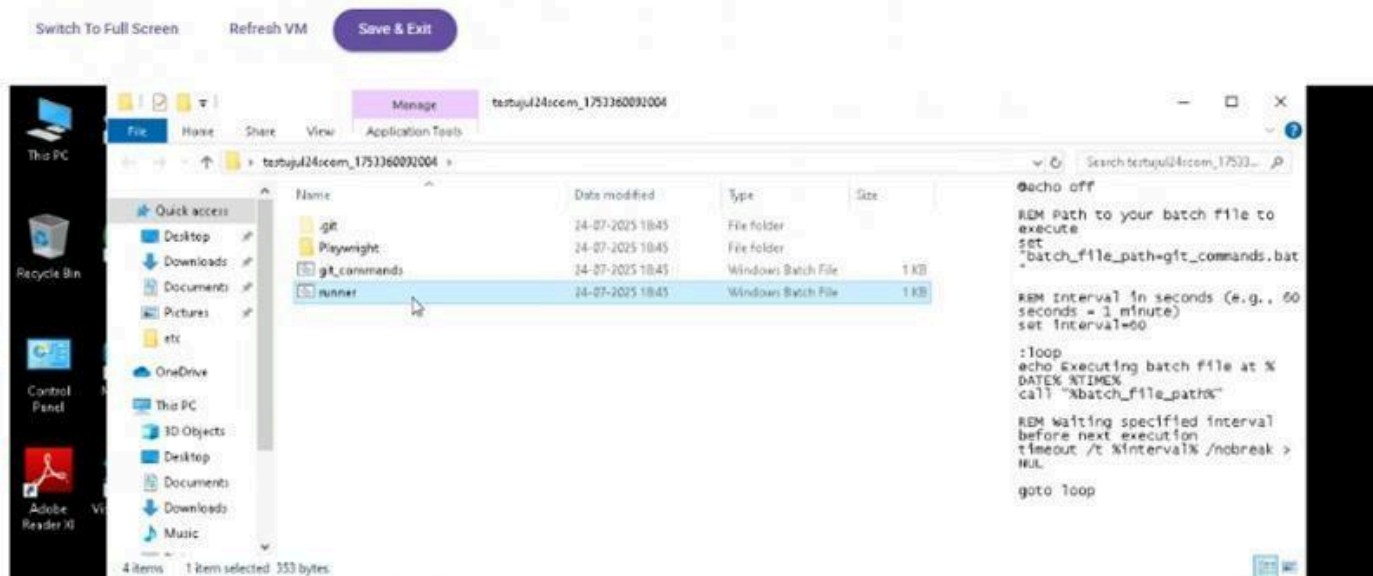
Execution Steps:

Steps for Execution:

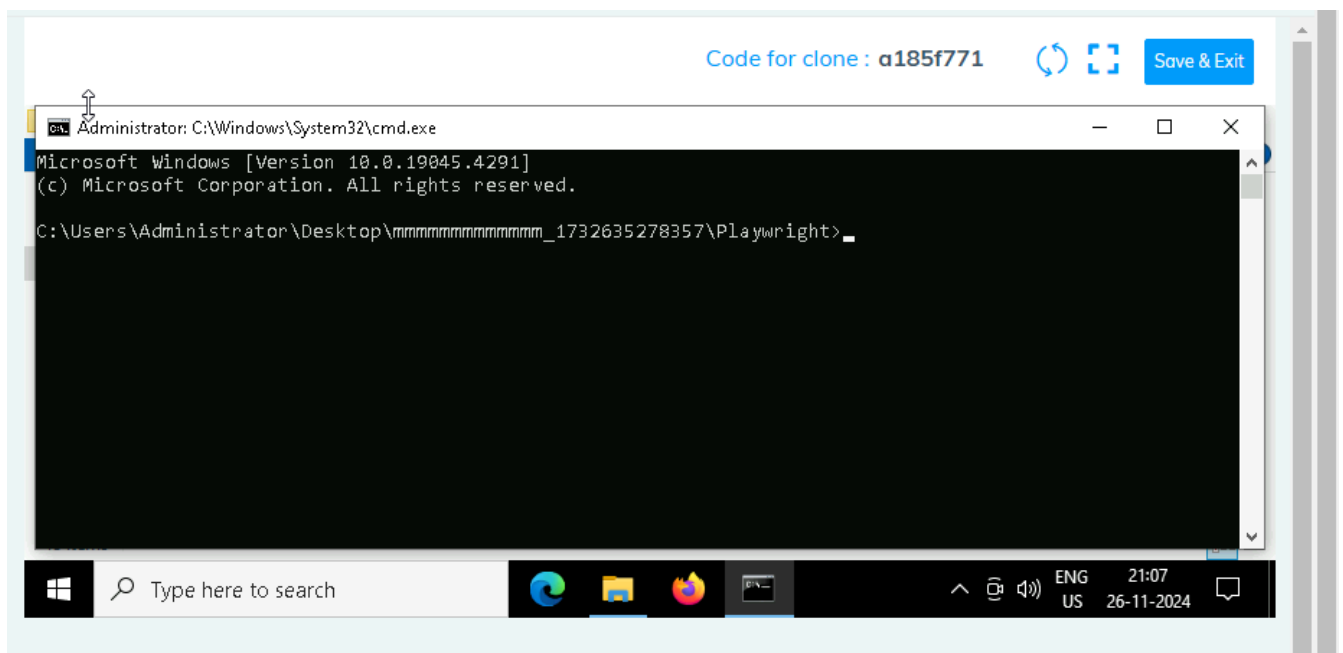
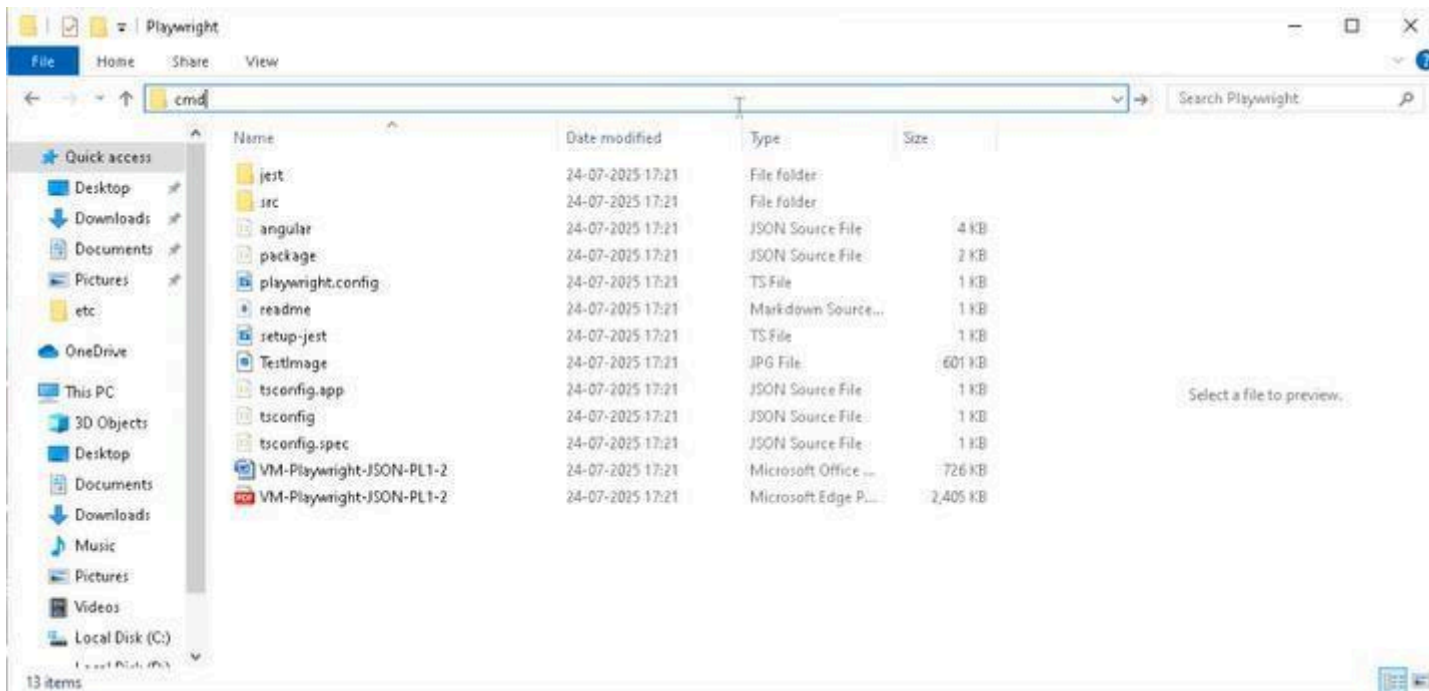
1. Please open the folder created on desktop with the email name you used to login.

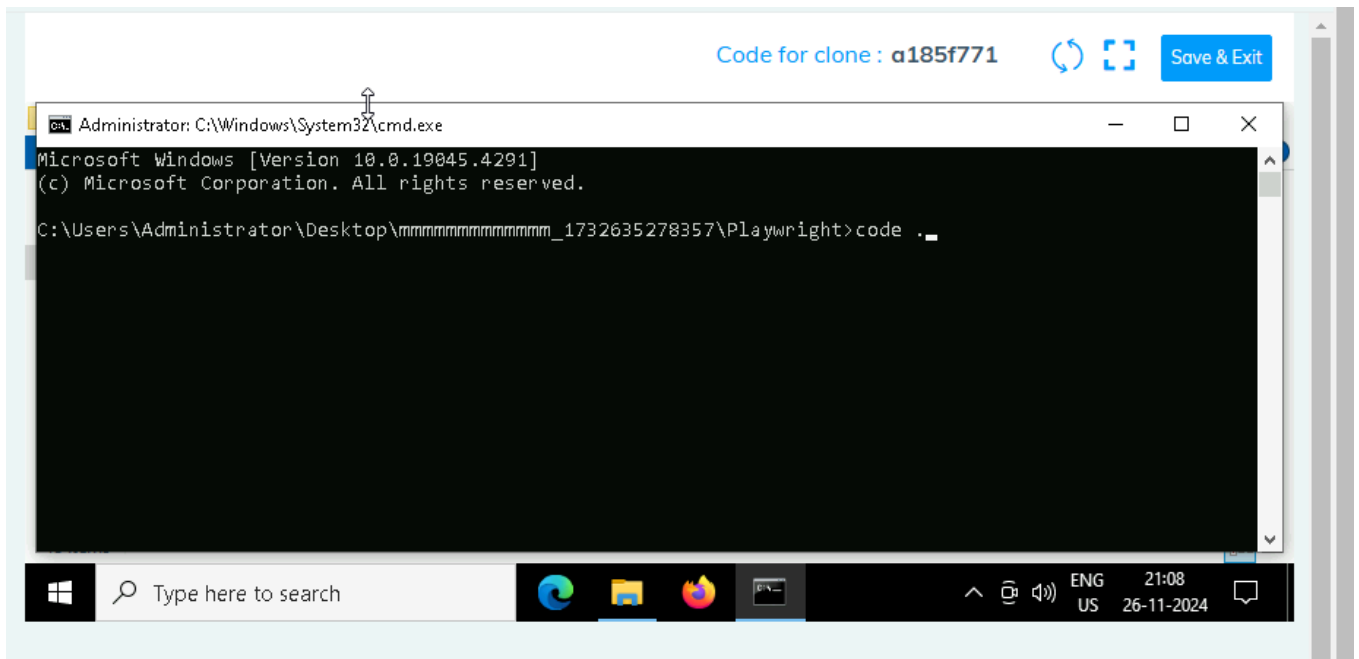


2. Execute this “runner” file. This will keep pushing the code at regular intervals

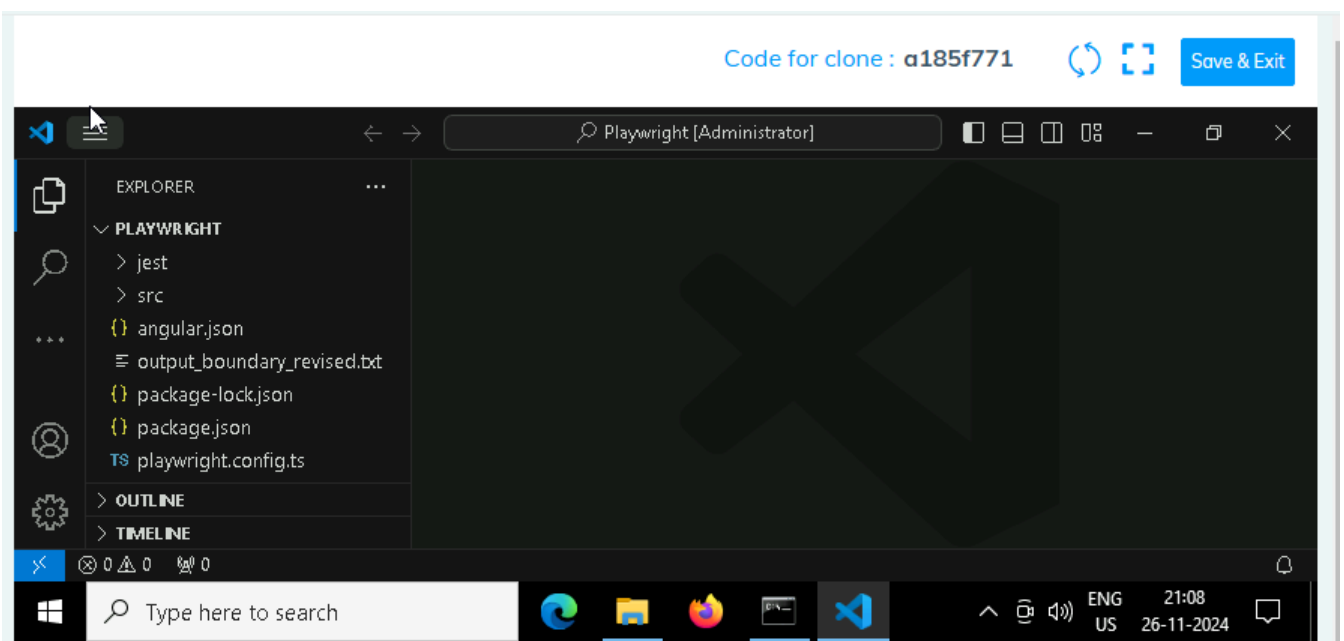


3. Go into the Playwright folder and Open command prompt with its location and use below command: code .





4. Once VS Code 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`

