|  |  |  |
| --- | --- | --- |
| YAKSHA HEALTH APP WITH TYPE  PLAYWRIGHT PL1-5 | SCRIPT AND |  |
| Mymedic automation using playwright |  |

**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.

|  |  |  |
| --- | --- | --- |
| **Path** | **File** | **Description** |
| src\data | login.json  TestData.json | 1. Contains data to read from json file. |

* 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](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 login.json, testData.json.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| src\ pages | ●  ●  ●  ● | AdminPage  BuzzPage  LoginPage  MyInfoPage | 1.  2. | All core activities are to be performed here.  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 Image could be  Uploaded as profile  Pic | 1. The application should read the data from login.json file and fetch the user’s name and password. 2. It should be called the method performLogin() and myinfo(). 3. Perform the myinfo method from MyInfoPage and perform the pic upload and set as profile pic. 4. Verify profile pic should be updated.     **It is must implement this login functionality at first and then implement any other test case.** | **Reference path**    \src\pages\**MyInfoPage**  **methods**  performLogin(), myInfo()  **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.** | image should get replaced |
|  |  |  | **Reference path** |  |
| 2 | Verify  Admin can edit record | 1. Login using valid credentials 2. Click on "Admin" tab in sidebar 3. Click on "edit" button | \src\ pages\**AdminPage methods** performLogin(), AdminEdit()  **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.** | Verify edit user form appears |
|  |  | 1.Login using valid credentials | **Reference path** |  |
| 3 | Verify the  Admin  can sort the records | 1. Click on "Admin" tab in sidebar 2. Click on "sort" button | \src\ pages\**AdminPage methods** performLogin(), adminSort() | List should get sorted accordingly |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | **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.** |  |
| 4 | Verify new Tab opens on clicking the "Upgrade" button | 1. Navigates to Admin section. 2. Clicks the upgrade button. 3. Waits for a new tab to open. 4. Retrieves the URL of the newly opened tab. | **Reference path**  \src\ pages\**AdminPage**  **Methods:**    performLogin(), upgrade()  **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.** | PicUpgrade to advance tab should open |
|  |  |  | **Reference path** |  |
| 5 | PicUpgr ade to  advance tab should open. | 1. Login using valid credentials 2. Click on "Admin" button in sidebar 3. Hover upon "?" button | \src\ pages\**MyinfoPage**  **methods**    PerformLogin()*,* helpHover()  **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.** | Help" should be displayed on hovering over the "?" button in top right |
| 6 | Verify admin  Seaarch  Functionality | 1. Login using valid credentials 2. Click on "Admin" button in sidebar 3. Enter the user role as admin from dropdown. 4. Click on "Search" button | **Reference path**    \src\pages\**AdminPage methods** :  performLogin(), AdminSearchVerify() | List having role "admin" should be displayed. |
| 7 | verify Image could be dragged and dropped in buzz tab | 1. Login using valid credentials 2. Click on "Buzz" tab in sidebar 3. Click on share photo button 4. Click on image area to up 5. drag & drop image> Click on   Share button | **Reference path**  \PageObjects\Pages\**BuzzPage methods**  PerformLogin(),SharePhotoPost() | Image should get uploaded |
| 8 | Verify 'Most Commented Posts' tab selection | 1. Login using valid credentials 2. Navigates to the Buzz section 3. Clicks the "Most Commented Posts" tab | **Reference path**    \src\pages\**BuzzPage**  **methods**  PerformLogin(),mostcommentTab() | **the tab is highlighted .** |
|

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 9 | Ensure comment can be successfully added to a post | 1. Navigates to the buzz section. 2. Clicks on the comment button for the first post 3. Fills in a unique comment using a timestamped message 4. Submits the comment by pressing Enter 5. Retrieves the most recent   comment text   1. Collects all comment elements on the page 2. Verifies that the posted comment is present in the list | **Reference path**  \src\ pages\**BuzzPage methods**  PerformLogin(), addCommentToPo st().  **You can use highlightElement method present in**  **CommonMethods file to highlight the element before performing any action on it. It takes locator as** | Ensures the comment submission flow is working correctly and the UI reflects the new comment as expected. |
|  |  | 1. Navigates to the Buzz section 2. Opens the options menu for the most recent post 3. Selects the edit option 4. Clears the existing post content 5. Fills in the new predefined post content (editPostText) 6. Saves the updated post 7. Verifies that the post now displays the updated content | **a parameter.**      **Reference path**    \src\pages\**BuzzPage** |  |
| 10 | Verify post content can be edited successfully |  | **methods**    PerformLogin(), editPost() | Confirms that users can update their shared posts and that those changes persist correctly. |
|  |  | **Note :- If you cannot find the locator for “Proceed” button, please try to keep the page at 75%-80%.** | **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. URL https:/opensource-demo.orangehrmlive.com

App. Key skills include:

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

# IMPLEMENTATION/FUNCTIONAL REQUIREMENT

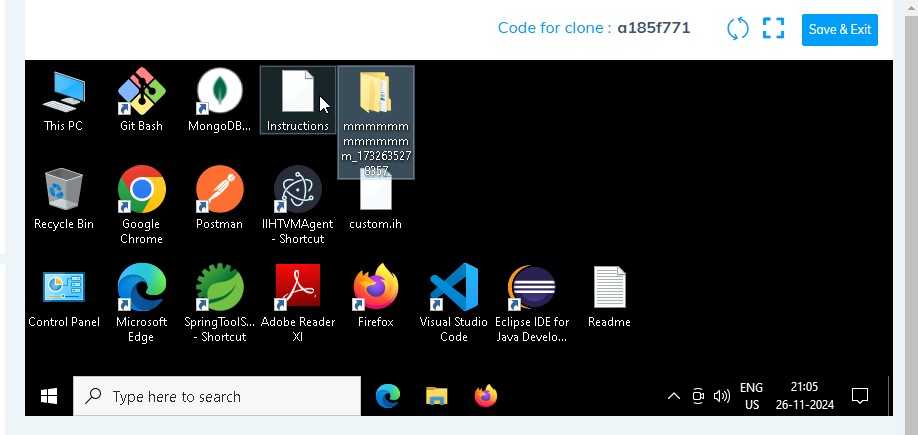
**1.1 CODE QUALITY/OPTIMIZATIONS**

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

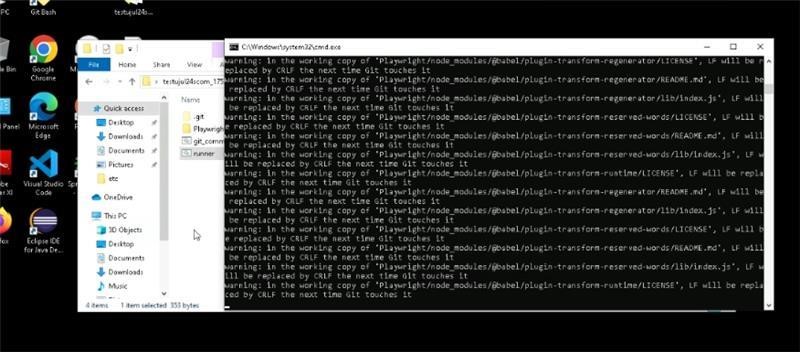
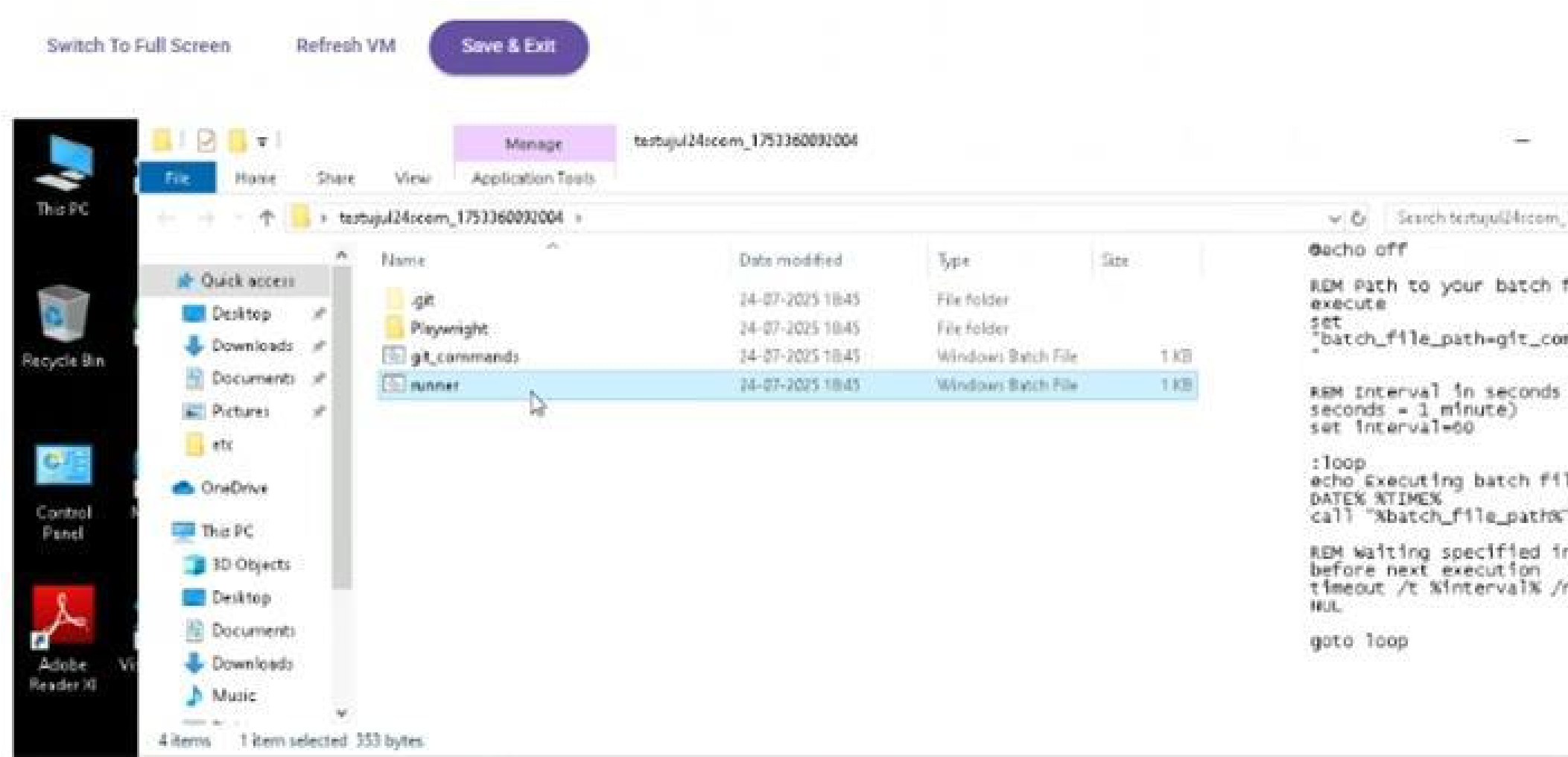
**Execution Steps:**

**Steps for Execution:**

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



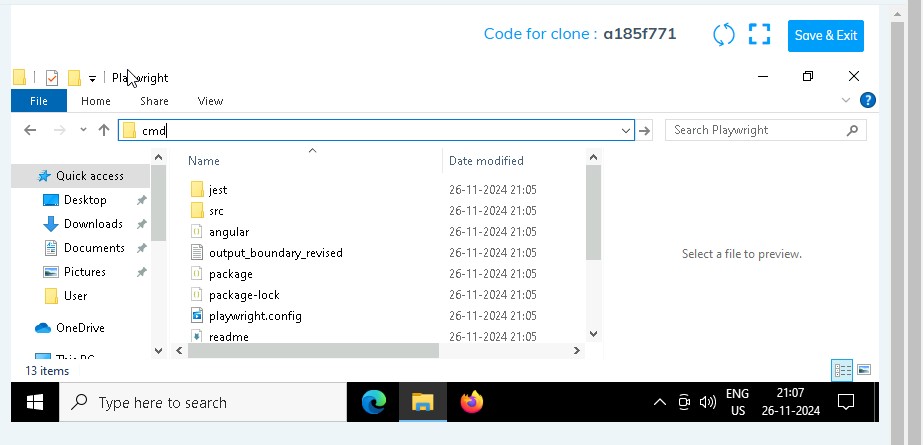
1. **Go into the Playwright folder and execute this “runner” file. This will keep pushing the code at**

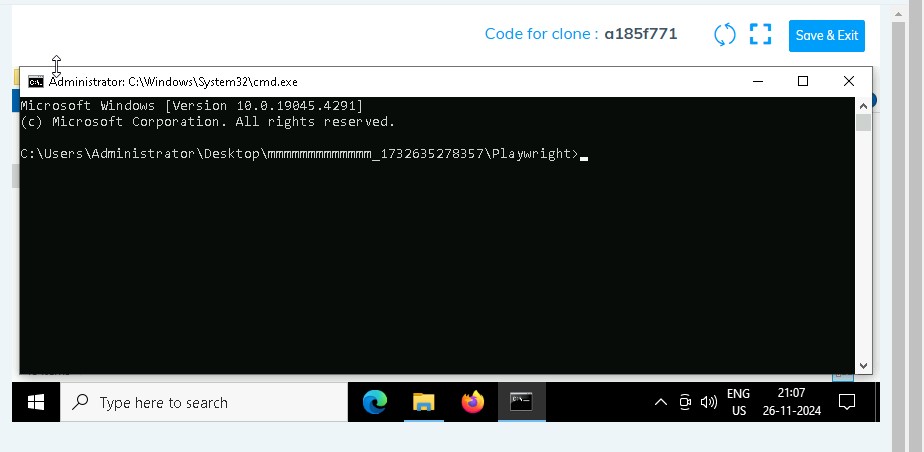


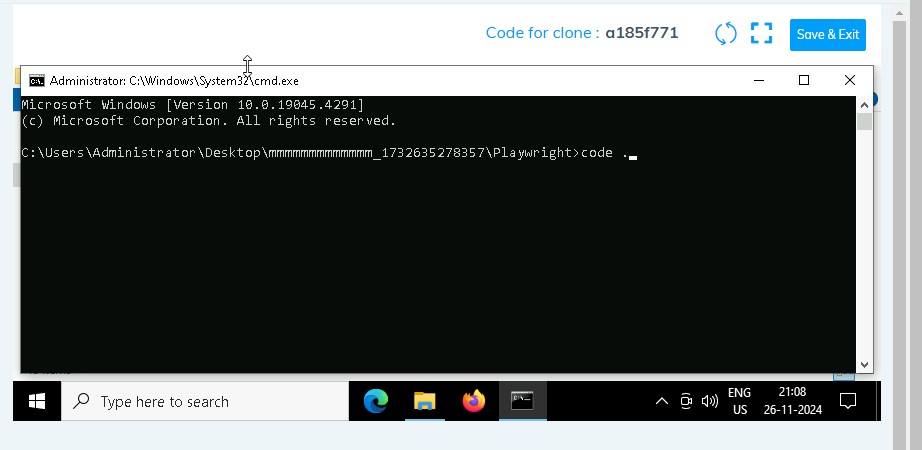
HRMS

1. **Go into the Playwright folder and Open command prompt with its**

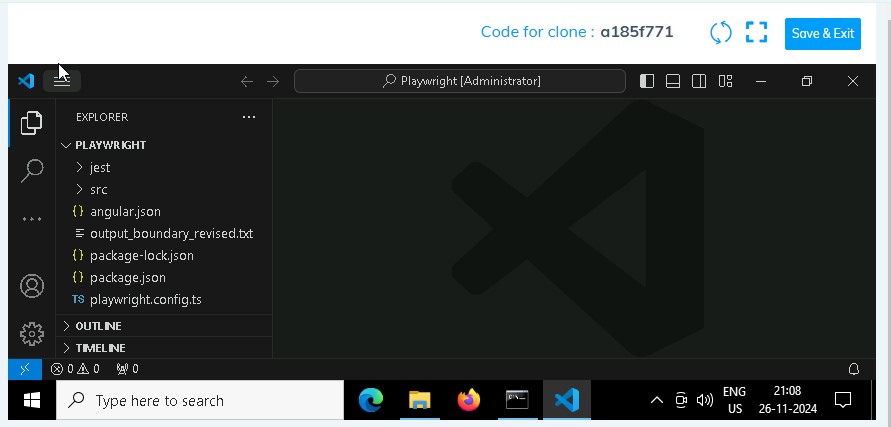
**location and use the command below: code .**







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



1. **Install all dependencies in the Playwright folder path using:**

**npm install**

1. **Install playwright in the Playwright folder path:**

**npx playwright install**

1. **Run the Tests in the Playwright folder path**:

**npx playwright test ./src/tests/PL1\_testcases/yaksha.spec.ts**