YAKSHAHEALTHAPP 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:

1. **Login/Registration**: For user authentication.
2. **Dashboard**: Main page displaying patient records.
3. **Patient Records**: Add, edit, delete, and view records.
4. **Appointment Scheduling**: Manage patient appointments.
5. **Search/Filter**: Filter records by various parameters (e.g., doctor, date).

Project Information:

- **Use Case**: Aimed at simplifying EMR management, enhancing healthcare record accessibility, and enabling easy interaction with patient data for healthcare providers
-

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

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

We have placed an excel file on desktop along with this file containing few fields which should be used while implementing.

Test Case No.	Test Case Name	Precondition	Steps	Expected Result
1	Verify Login with Valid Credentials	User is on the login page https://healthapp.yaksha.com	1. Navigate to https://healthapp.yaksha.com/Home/Index# 2. Enter username as admin. 3. Enter password as pass123. 4. Click the login button.	The user should be successfully logged in and redirected to the dashboard or homepage after login.

2	Verify Page Navigation andLoad Time for Billing Counter	User logged in	1. Navigate to https://healthapp.yaksha.com/Home/Index#/Utilities/ChangeBillingCounter 2. Measure the page load time. 3. Assert that the page load time is within the acceptable threshold (e.g., 3 seconds).	The billing counter page should load within the acceptable timeframe. Page loaded successfully
3	Patient Search with Valid Data for appointment page	User logged in and on appointment page	1. Navigate to https://healthapp.yaksha.com/Home/Index#/Appointment/CreateAppointment 2. Enter valid patient data (read values from excel i.e **patientName1**). 3. Submit search and verify results.	The correct patient search results should be displayed.
4	Activate Counter in Dispensary	User logged in and on Dispensary page	1. Navigate to https://healthapp.yaksha.com/Home/Index#/Dispensary/ActivateCounter 2. Click the "Activate Counter" button (anyone from 3 i.e Morning or Evening or Night counter). 3. Verify that the success message is displayed.	The counter should be successfully activated, and aconfirmation message shouldappear.
5	Purchase Request List Load	User logged in	1. Navigate https://healthapp.yaksha.com/Home/Index#/ProcurementMain/PurchaseRequest/PurchaseRequestList 2 Select a date which is one year past the current date. 3. Click ok. **4.** Verify the purchase request list page loads successfully.	The purchase request list should load withall the relevant data displayed.
6	Lab Dashboard Data Validation	User logged in and on Lab Dashboard page	1. Navigate to https://healthapp.yaksha.com/Home/Index#/Lab/Dashboard 2. Verify that dashboard data (e.g., Test Requests till Date having Total value as 4) is displayed correctly. 3. Compare the displayed values with expected data for validation.	The lab dashboard should display accurate and up-to-date data.
7	Handle Alert onBilling Counter	User logged inand on Billing Counter page	1. Navigate to https://healthapp.yaksha.com/Home/Index#/Utilities/ChangeBillingCounter 2. Perform an action to activate it. Incase it is already activated, first deactivate it and then activate it again. 3. Handle the alert using Playwright's dialog handling. 4. Verify the application continues to function as expected after dismissing the alert.	The alert shouldbe handled successfully without causingtest failure.
8	Data-Driven Testing for Patient Search	User logged in and search patient	1. Navigate to https://healthapp.yaksha.com/Home/Index#/Patient/SearchPatient 2. Read patient data from an external excel file to search (**patientName1, patientName2, patientName3**). 3. Use that data to search for multiple patients. 4.Validate search results for each patient.	Patient search should work correctly for alldata sets from the excel file.
9	Error Handling and Logging in Purchase Request List	User logged in	1. Navigate to https://healthapp.yaksha.com/Home/Index#/ProcurementMain/PurchaseRequest/PurchaseRequestList 2. Intentionally enter an invalid date range where FromDate > ToDate. 3. Capture the error message displayed by the application.	Error message displayed and logged successfully: "FromDate cannot be more than ToDate."

			4. Log the error message in an excel file. 5. Validate that the error message is displayed correctly and logged successfully.	
10	Keyword-Driven Framework for Appointment Search	User logged in	1. Define reusable keywords like SearchPatient and VerifyResults. 2. Use SearchPatient keyword to navigate to https://healthapp.yaksha.com/Home/Index#/Appointment/PatientSearch and perform the search by reading the data from excel file (**patientName1**). 3. Use VerifyResults keyword to validate the search results against expected values. 4. Assert that the search results are accurate based on the keyword output.	The keyword- driven test should successfully search for a patient and verify results.
11	Verify Sorting by Hospital Number in Patient Search	User logged in	1. Navigate to https://healthapp.yaksha.com/Home/Index#/Patient/SearchPatient 2. Click on the Hospital Number column header to sort patients. 3. Capture the list of hospital numbers displayed. 4. Validate that the hospital numbers are sorted numerically. 5. Log the results of the validation.	Sorting worked correctly; hospital numbers appeared in the correct order.
12	Verify Locator Strategy for Appointment Search	User logged in and on Appointment Search page	1. Navigate to https://healthapp.yaksha.com/Home/Index#/Appointment/PatientSearch 2. Use CSS selectors to locate the search bar 3. Input search criteria into the located element by reading data from external excel file (**patientName1**). 4. Perform the search. 5. Assert that the displayed patient list matches the search criteria. 6. Log the results of the validation in an excel file.	The correct locators should be used, and the patient list should be accurately displayed.
13	Validate Element Inspection and Tooltip Visibility in Inventory Stock List	User logged inand on Lab Inventory Stock List page	1. Navigate to https://healthapp.yaksha.com/Home/Index#/Inventory/StockMain/StockList 2. Use Playwright Inspector to locate the search bar and button elements (export and print button) 3. Hover over a button to display its tooltip. 4. Verify that the tooltip text is visible and matches the expected description by reading the data from external excel file (**exportButtonTooltip** and **printButtonTooltip**) 5. Log the results of the validation into an excel file.	Elements shouldbe correctly identified, and the alert should be handled without issues.
14	Handle Navigation Exception on Activate Hospital Page	User logged inand on Activate Hospital Page	1. Attempt to navigate to https://healthapp.yaksha.com/Home/Index#/Accounting/Transaction/ActivateHospital 2. Capture any exceptions or errors during navigation in an excel file. 3. Verify if the page loads correctly or displays an error. 4. Log the navigation status (success or failure) and the exception details in an excel file.	Navigation exceptions should be handled gracefully, and the page shouldload after retrying.
15	Web Element Handling for Dropdowns in Purchase Request	User logged in and on Purchase Request page	1. Navigate to https://healthapp.yaksha.com/Home/Index#/ProcurementMain/Reports/Stock/StockLevel 2. Select a specific item (e.g., "Accounts") from the item filter dropdown. 3. Apply the filter. 4. Verify that the displayed stock level report includes only items from the selected store name. 5. Log the results of the validation in an excel file.	The correct data should be displayed based on the selected filter from the dropdown menu.
	Form and	User is logged in	1. Navigate to	Appropriate error

16	Error Messages	and navigates to the Purchase Order page	https://healthapp.yaksha.com/Home/Index#/ProcurementMain/PurchaseOrder/PurchaseOrderAdd 2. Attempt to submit the Purchase Order form with missing or invalid values in required fields (e.g give invalid currency code). 3. Capture and verify the error messages displayed. 4. Repeat the test for different invalid input scenarios. 5. Log the results of the validation in an excel file.	messages are displayed for missing or invalid inputs.
17	Verify Handling of Frames on Patient List Page	User is logged in and navigates tothe Patient List page.	1.Navigate to https://healthapp.yaksha.com/Home/Index#/SSU/PatientList 2. Search for any patient name by reading data from external excel file (**patientName1**) in "edit information of" field. 3. Select the patient and validate the patient's name in the newly opened frame. 4. Write the playwright code to navigate to pervious to page. 5. Log the results of the validation in an excel file.	The test should successfully locate the frame, interact with elements inside it, Frame switching and interactions worked as expected.
18	Verify Handlingof Tabs for Billing Reports and Other Reports	navigate to the Reports section.	1. Navigate to https://healthapp.yaksha.com/Home/Index#/Reports 2. Click on opens Billing Reports. 3. Click on other tabs to check if they are switchable (e.g Appointment, Radiology, Lab, Doctors, and Patient reports.)	The test should successfully switch between tabs, interact with elements on each report tab, and verify that the correct data is displayed.
19	Verify Handlingof Popups for Changing BillingCounter	User is loggedin and navigates to the Billing Counter page.	1. Navigate to https://healthapp.yaksha.com/Home/Index#/Lab/Settings/LabTest 2. Perform an action that triggers a popup (e.g., selecting add new lab test). 3. Capture and handle the popup using Playwright's dialog handling or dismiss the popup.	The popup should be successfully captured and handled. The testshould either accept or dismiss the popup as required.

EXPECTATIONS:

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 behavior meets expected results.
- **End-to-End Testing**: Automating real user interactions and validating overall app functionality.

IMPLEMENTATION/FUNCTIONAL REQUIREMENT

Mymedic automation using playwright

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. **Set up Playwright** by installing it in the project:

   npm install --save-dev playwright

2. **Create Test File** for each test case or combine them into one file for easier execution, e.g., healthapp-tests.spec.ts.

3. **Run the Tests**:

   npx playwright test