YAKSHA HEALTH APP WITH TYPESCRIPT AND 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 and Load Time for Billing Counter	User logged in	1. Go to Sidebar Toggler -> Utilities -> Change Billing Counter. 2. Verify the URL is: https://healthapp.yaksha.com/Home/Index#/Utilities/ChangeBillingCounter 3. Measure the page load time. 4. 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.	1. Go to Sidebar Toggler -> Appointment. 2. Click on the first counter (New-1). 3. Navigate to Sidebar Toggler -> Appointment -> Book Appointment. 4. Verify the URL is: https://healthapp.yaksha.com/Home/Index#/Appointment/CreateAppointment 5. Enter valid patient data (from Excel). (read values from excel i.e **patientName1**). 6. Submit search and verify results.	The correct patient search results should be displayed.
4	Activate Counter in Dispensary	User logged in.	1. Go to Sidebar Toggler -> Dispensary. 2. Verify the URL is: https://healthapp.yaksha.com/Home/Index#/Dispensary/ActivateCounter 3. Click "Activate Counter" button. (anyone from 3 i.e Morning or Evening or Night counter). 4. Navigate to Sidebar Toggler -> Dispensary -> Counter. 5.You can view the tile with the message “**You have activated the counter and the current activated counter is” and the counter you have selected.**	The counter should be successfully activated, and a confirmation tile should appear.
5	Purchase Request List Load	User logged in	1. Go to Sidebar Toggler -> Procurement -> Purchase Request. 2. Verify the URL is: https://healthapp.yaksha.com/Home/Index#/ProcurementMain/PurchaseRequest/PurchaseRequestList 3. Select a date one year past the current date and click OK. 4. Verify the purchase request list page loads successfully.	The purchase request list should load with all the relevant data displayed.
6	Lab Dashboard Data Validation	User logged in.	1. Go to Sidebar Toggler -> Laboratory. 2. Verify URL: https://healthapp.yaksha.com/Home/Index#/Lab/Dashboard 3. Validate dashboard data (e.g., "Test Requests till Date = 4"). 4. Compare displayed values with expected data for validation.	The lab dashboard should display accurate and up- to-date data.
7	Handle Alert on Billing Counter	User logged in.	1. Go to Sidebar Toggler -> Utilities -> Change Billing Counter. 2. Verify URL: https://healthapp.yaksha.com/Home/Index#/Utilities/ChangeBillingCounter 3. Deactivate the counter. 4. Handle alert using Playwright's dialog handling. 5. Go to Index page. 5. Verify the application works as expected.	The alert should be handled successfully without causing test failure.

8	Data-Driven Testing for Patient Search	User logged in.	1. Go to Sidebar Toggler -> Patient -> Search Patient. 2. Verify the URL is: https://healthapp.yaksha.com/Home/Index#/Patient/SearchPatient 3. Read patient data from Excel (e.g., **patientName1**, **patientName2**). 4. Validate search results for all patients.	Patient search should work correctly for all data sets from the excel file.
9	Error Handling and Logging in Purchase Request List	User logged in.	1. Go to Sidebar Toggler -> Procurement -> Purchase Request. 2. Verify the URL is: https://healthapp.yaksha.com/Home/Index#/ProcurementMain/PurchaseRequest/PurchaseRequestList 3. Enter an invalid date range (FromDate > ToDate). 4. Capture and log error message in Excel. 5. Validate that the error message is displayed correctly and logged successfully.	Error message displayed and logged successfully: "FromDate cannot be more than ToDate."
10	Verify Sorting by Hospital Number in Patient Search	User logged in.	1. Go to Sidebar Toggler -> Patient -> Search Patient. 2. Verify the URL is: https://healthapp.yaksha.com/Home/Index#/Patient/SearchPatient 3. Click on the "Hospital Number" column header. 4. Capture the list of hospital numbers displayed. 5. Validate the numbers are sorted numerically. 6. Log the results.	Sorting worked correctly; hospital numbers appeared in the correct order.
11	Verify Locator Strategy for Appointment Search	User logged in and **counter must be activated**.	1. Go to Sidebar Toggler -> Appointment -> New Visit. 2. Verify the URL is: https://healthapp.yaksha.com/Home/Index#/Appointment/PatientSearch 3. Use CSS selectors to locate the search bar. 4. Input search criteria into the located element by reading data from external excel file (**patientName1**). 5. Perform the search. 6. Assert that displayed patient list matches search criteria. 7. Log the results in Excel file.	The correct locators should be used, and the patient list should be accurately displayed.
12	Validate Element Inspection and Tooltip Visibility in Inventory Stock List	User logged in.	1. Go to Sidebar Toggler -> Inventory -> Stock. 2. Verify the URL is: https://healthapp.yaksha.com/Home/Index#/Inventory/StockMain/StockList 3.Use Playwright Inspector to locate the search bar and button elements (export and print button) 4.Hover over a button to display its tooltip. 5.Verify that the tooltip text is visible and matches the expected description by reading the data from external excel file (**exportButtonTooltip** and **printButtonTooltip**) 6.Log the results of the validation into an excel file.	Elements should be correctly identified, and the alert should be handled without issues.
13	Handle Navigation Exception on Closing Account	User logged in.	1. Go to Sidebar Toggler -> Accounting -> Transaction. 2. Click on Account Closure tab. 3. Verify the URL is: https://healthapp.yaksha.com/Home/Index#/Accounting/Transaction/AccountClosure 4. Click on “Close Account” button. 5. Then click on “Yes”. 6.Capture any exceptions in or errors.	Exceptions should be handled gracefully.

14	Web Element Handling for Dropdowns in Purchase Request	User logged in.	1. Go to Sidebar Toggler -> Procurement -> Reports. 2. Click on "Current Stock Level". 2. Verify the URL is: https://healthapp.yaksha.com/Home/Index#/ProcurementMain/Reports/Stock/StockLevel 3. Select a specific item (e.g., "Accounts") from the item filter dropdown. 4. Apply the filter. 5. Verify that the displayed stock level report includes only items from the selected store name. 6. 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.
15	Form and Error Messages	User is logged in.	1. Go to Sidebar Toggler -> Procurement -> Purchase Order -> Create Purchase Order. 2. Verify the URL is: https://healthapp.yaksha.com/Home/Index#/ProcurementMain/PurchaseOrder/PurchaseOrderAdd 3. Attempt to submit the Purchase Order form with missing or invalid values in required fields (e.g give invalid currency code). 4. Capture and verify the error messages displayed. 5. Repeat the test for different invalid input scenarios. 6. Log the results of the validation in an excel file.	Appropriate error messages are displayed for missing or invalid inputs.
16	Verify Handling of Frames on Patient List Page	User is logged in.	1.Go to Sidebar Toggler -> Social Service. 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.
17	Verify Handling of Tabs for Billing Reports and Other Reports	User logged in.	1. Go to Sidebar Toggler -> Reports. 2.Verify the URL is : https://healthapp.yaksha.com/Home/Index#/Reports 2.Click on different tabs like "Admission", "Billing Reports", "Appointments" etc. 3.And 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.
18	Verify Handling of Popups for Changing BillingCounter	User is logged in.	1. Go to Sidebar Toggler -> Laboratory -> Settings. 2. Verify the URL is : 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 test should either accept or dismiss the popup as required.

EXPECTATIONS:

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

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

1. **Create a folder on desktop with name as of your email id to put all your deliverables.**
2. **Open command prompt and navigate to folder you created on desktop and execute below commands:**
    a. **To initialize current directory with playwright configurations**

    ```
    npm init playwright@latest
    ```
3. **Please configure your test cases to run on Chrome browser.**
4. **Create test file for each test case or combine them into one file for easier execution e.g healthapp-test.spec.ts**
5. **Run all test cases:**

    ```
    npx playwright test
    ```
6. **Run single test file:**

    ```
    npx playwright test file_name
    ```