YAKSHA HEALTH APP WITH JAVASCRIPT AND CYPRESS

Yaksha health app test automation using cypress

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:** Cypress (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

Cypress Automation Test Cases

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

Test Case ID	Test Case Title	Test Steps	Expected Result
1	Verify Login with Valid Credentials	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	Apply Assertions for Validation	1. Navigate to the login page at https://healthapp.yaksha.com/Home/Index#/ 2. Use the username as admin and password as pass123. 3. Use Remember Me checkbox. 4. Submit the login form with valid credentials and validate navigation.	Assertions confirm the presence of mandatory fields, and optional fields like Remember Me toggle correctly.
3	Lab Dashboard Data Validation	1. Navigate to https://healthapp.yaksha.com/Home/Index#/Lab/Dashboard 2. Verify that dashboard data (e.g., Test Requests till Date is 4) is displayed correctly.	The lab dashboard should display accurate result value of 4.
4	Modular Approach for Business Logic	1. Navigate to the Patient Registration page. https://healthapp.yaksha.com/Home/Index#/Patient/RegisterPatient/BasicInfo 2. Read excel file to fetch patient details for adding patient (**firstName, middleName, lastName** and all other required fields). 3. Validate new patient data in the list.	Patient record is successfully added and visible in the patient list.
5	Data-Driven Testing for Patient Search	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 all data sets from the excel file.
6	Verify Page Navigation and Load Time for Billing Counter	1. Navigate to https://healthapp.yaksha.com/Home/Index#/Utilities/ChangeBillingCounter 2. Measure page load time. Example (2.5 seconds below load time). 3. Assert that page loads within acceptable time.	The billing counter page should load within the acceptable time frame under 2.5 seconds).
7	Activate Counter in Dispensary	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 a confirmation message should appear.
8	Purchase Request List Load	1. Navigate https://healthapp.yaksha.com/Home/Index#/ProcurementMain/PurchaseRequest/PurchaseRequestList 2. Select a date which is one year past the current date.	The purchase request list should load with all the relevant data displayed.

Yaksha health app test automation using cypress

		3. Click ok. 4. Verify the purchase request list page loads successfully.	
9	Keyword-Driven Framework for Appointment Search	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.	The keyword- driven test should successfully search for a patient and verify results.

10	Manage Web Element Interactions	1.Navigate to the BillReturnRequest page https://healthapp.yaksha.com/Home/Index#/Billing/BillRReturnRequest 2.Locate and click the Fiscal Year dropdown. 3.Select 2023 and check if you are able to select the value.	You are able to select the data from the dropdown function.
11	Implement Wait Strategies	1. Navigate to https://healthapp.yaksha.com/ClaimManagement/BillReview 2. Apply explicit waits for dynamic elements to fully load. 3. Verify required elements are visible and functional after loading.	Wait strategies ensure no timeouts, and all elements load successfully.
12	Switching Between Pages & Windows	1.Navigate to https://healthapp.yaksha.com/Home/Index#/Billing/BillReturnRequest 2.Validate that the page loads successfully by checking the title and content. Try to navigate from one page to another page by clicking https://healthapp.yaksha.com/Home/Index#/ProcurementMain/GoodsReceipt/GoodsReceiptList page 3. Click on the **print button** link and validate redirection to the print window. 4. Switch to the print window and validate its content. 5. Navigate back to the Bill Return Request page and validate the redirection. 6. Log the results in the excel file.	Navigation between pages works seamlessly, with no errors during transitions. We can see that print window and good receipt list open navigated successfully.
13	Automate Complex Data Input	1. Navigate to https://healthapp.yaksha.com/Home/Index#/Appointment/CreateAppointment 2. Search for an existing patient (e.g., Jane Doe) in the patient search bar. 3. Fill in the appointment details: - Date: 2023-11-20 - Time: 11:00 AM - Doctor: Dr. Adams. 4. Save the appointment by clicking the Create Appointment button. 5. Verify that the appointment is created and appears in the appointment list.	The appointment has been successfully registered and the appointment appears in the appointment list.
14	File Upload/Download Handling	1. Navigate to https://healthapp.yaksha.com/Home/Index#/Reports/Patient/Patient/RegistrationReport 2. Select the desired filters for the report (e.g., Date Range: 2023-11-01 to 2023-11-12). 3. Click the Generate Report button. 4. Validate that the report displays patient registration data for the selected date range. 5. Export the report (e.g., as a PDF or Excel) and verify its contents.	The Patient Registration Report is generated successfully, displaying accurate data for the selected filters. The exported report (PDF/Excel) matches the displayed data.

15	Secure File Download Verification	1. Navigate to https://healthapp.yaksha.com/Home/Index#/Reports 2. Download a secure file (e.g., report.pdf). 3. Verify file content and access control (e.g., login required). 4. And download the report.	Secure file downloads (e.g., report.pdf) are completed successfully with valid content and access control.
16	JavaScript Alert Handling	1. Navigate to the registration form page https://healthapp.yaksha.com/Home/Index#/OperationTheatre/OtBookingList 2. Keep the select patient empty to check if javascript is triggered 3. Click Submit. 4. Handle and verify the alert text. 5. Dismiss the alert.	Alert shows the correct error message and can be dismissed. The form remains on the same page for correction.
17	Verify Add/Remove Elements	1. Navigate to the Departments page. https://healthapp.yaksha.com/Home/Index#/Settings/DepartmentsManage/Department 2. Read department data from an external excel file to add department (**departmentName**) 3. Validate it is added. 4. Remove a department in the list 5. Validate it is removed.	Department ward is added and displayed in the list. Removed department is no longer visible.
18	Shadow DOM Element Handling	1.Navigate to the Appointment Details Report page. https://healthapp.yaksha.com/Home/Index#/Appointment/ 2.Select "New" in the Appointment Type filter and generate the report. 3.Validate the report shows only "New" appointments. 4.Change to "Follow-Up" and regenerate. 5.Validate the report shows only "Follow-Up" appointments.	Report filters and displays data correctly based on the selected appointment type. Shadow DOM elements are interactable and functional.

19	Advanced Locator Strategy	1. Navigate to the Patient Search page. https://healthapp.yaksha.com/Home/Index#/Patient/SearchPatient 2. Use locators to identify the search bar. 3. Enter a search term (e.g Baby of Joe) and validate the results.	Locators correctly identify elements, and search results are accurate and displayed as expected.
20	Infinite Scroll Verification	1. Navigate to the Patient Search page.https://healthapp.yaksha.com/Home/Index#/Patient/SearchPatient 2. Enter any patient to search (e.g Baby of Joe) 3. Scroll to the bottom to trigger loading additional records. 4. Validate records load dynamically. 5. Repeat scrolling to verify seamless loading.	Infinite scrolling loads new records without errors, and the total number of records increases progressively.
21	Dynamic Controls Verification	1. Navigate to https://healthapp.yaksha.com/Home/Index#/Maternity/PatientList 2. Select a specific date range from 12-01-2022 to 19-11-2024, then click the "OK" button. 3. Click on the "View All Maternity Patients" checkbox. 4. The data displayed will include all individuals under the maternity list	Assert that the list updates dynamically and only relevant results are shown. Disable the checkbox again and verify that the filter reset works as expected.
22	Error Handling & Logging	1. Navigate to https://healthapp.yaksha.com/Home/Index#/Dispensary/ActivateCounter 2. Activate a counter e.g Morning Counter 3. Navigate to https://healthapp.yaksha.com/Home/Index#/Dispensary/PatientConsumptionMain/PatientConsumption 4. Click on “New Consumption” button. 5. Keep the “Search Patient” textbox empty. 6. Click on “Save Consumption” button. 7. Click “Confirm” button on pop-up dialog box. 8. Validate that an error message is displayed. 9. Capture the error and log details, including the input, timestamp, and error message.	Errors like "Invalid username" are logged with sufficient details for debugging.
23	Verify Handling of Popups for Changing Billing Counter	1. Navigate to https://healthapp.yaksha.com/Home/Index#/Utilities/ChangeBillingCounter 2. Capture and handle the popup using Cypress dialog handling. 3. Close the pop-up.	The popup should be successfully captured and handled. The test should dismiss the popup as required.
24	Context Menu Interaction	1. Navigate to https://healthapp.yaksha.com/Home/Index#/Settings/DepartmentsManage/Department 2. Locate a department record (e.g Billing). 3. Select on edit button to edit the form. 4. Validate that the corresponding action is triggered successfully.	Edit Department triggers the corresponding functionality to open the department form to edit it.

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

   ```
   npm install cypress --save-dev
   ```

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

3. **Run the Tests**:

   Open Cypress Test Runner
   If you want to run tests interactively in the Cypress Test Runner, use the following command:

   ```
   npx cypress open
   ```

   This will open the Cypress Test Runner GUI. From here, you can:
   - **Select the testing environment (E2E Testing).**
   - **Click on your test file (e.g., health_app.spec.js) to run the test interactively.**