YAKSHA HEALTH APP WITH JAVACRIPT AND CYPRESS

Mymedic 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 that are to be focused for the application

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

PROBLEM STATEMENT

Need to automate the following activities using cypress+javascript

You will be given few Json files in fixtures folder like PatientName.json and ValidLogin.json.

Path	File	Description
fixtures	PatientName.json ValidLogin.json	1. Contains data to read from json file.
PageObject\Pages	● AdminPage ● AppointmentPage ● DispensaryPage ● DoctorPage ● IncentivePage ● LaboratoryPage	1. All core activities to be performed here. 2. The comments associated with each templated method here describe the expectation. 3. Declare any variable/object you

Mymedic automation using cypress

			LoginPage • OperationTheaterPage • PatientPage • ProcurementPage	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
	Verify Login with Valid Credentials	1. The application should read the data from ValidLogin.json file and fetch the user's name and password. 2. It should call the method performLogin() 3. Perform login method will perform authentication with the username and password. 4. Verify admin name is visible on the home page. **It is must to implement this login functionality at first and then implement any other test case.**	**Reference path** \PageObject\Pages **LoginPage** **methods** performLogin()	Successfully logs in with provided credentials. The user is logged in the admin page.
1	Verify the presence of Visit Type drop down by selecting "New patient" option	1. Use verifyVisitTypeDropdown(). 2. Open "Appointment Booking List" tab inside Appointment module. 3. Select any counter, if it's not selected already and then select "Appointment Booking List" tab. 4. Select "New Patient" from "Visit Type" dropdown. 5. Pick "01-01-2024" in "From Date" field. 6. Pick "31-03-2024" in "To Date" field. 7. Click on "Show Patient" button.	**Reference path** \ PageObject\Pages**AppointmentPage** **methods** verifyVisitTypeDropdown()	The "Visit Type" column in list should contain only patients of "New" category.
2	Handle Alert for OT Booking Without Patient Selection	1. Navigate to Operation theatre page. 2. Use handleOtBookingAlert() to handle the alert. 3. Click on "New OT Booking" button. 4. Verify that the "Booking OT Schedule | New Patient" modal is displayed. 5. Without entering any details, within the modal, click on the "Add New OT" button. 6. Handle the alert with message "Patient not Selected! Please Select the patient first!".	**Reference path** \ PageObject\Pages**OperationTheatrePage** **methods** handleOtBookingAlert()	1. An alert with the message "Patient not Selected! Please Select the patient first!" is displayed. 2. Handle and accept the alert to proceed.
3	Verify Patient Overview Page Displays Information Correctly	1. Use verifyPatientOverview(). 2. Read data from PatientName.json file for any one patient name. 3. Goto "Doctor" module, then "In Patient Department" tab.	**Reference path** \PageObject\Pages **DoctorPage** **methods**	

Mymedic automation using cypress

Test Case No.	Test Case Name	Test Steps to be performed	Path & Method Used	Expected Result
		4. In the search bar, enter the patient name read from patientName.json file and perform the search. 5. Locate the patient in the results and click on the "Preview" icon under the Actions column.	verifyPatientOverview()	Verify the same patient overview page is displayed with the same patient's name.
4	Add Progress Note for In Patient	1. Use addProgressNoteForPatient() to check pop up message. 2. Go to "Doctor" module, then "In Patient Department" tab. 3. In the search bar, enter the patient's name read from patientName.json file and perform the search. 4. Locate the patient in the results and click on the "Preview" icon under the Actions column. 5. Click on "Notes" section available on left side of the panel below Profile. 6. Click on "Add Notes" button. 7. Select "Progress Note" option from "Template" dropdown. 8. Enter subjective Notes as "Test Notes" and click on save button.	**Reference path** \PageObject\Pages **DoctorPage** **methods** addProgressNoteForPatient()	The method should successfully add a Progress Note for the patient, and a success confirmation message with the text "Progress Note Template added." should be displayed.
5	Add and Verify New Currency in Settings	1. Navigate to Procurement > Settings. 2. Select "Currency" sub tab. 3. Click "Add Currency" button. 4. Add any data in "Currency Code" and "Description" fields. 5. Click on "Add Currency" button.	**Reference path** \PageObject\Pages**ProcurementPage** **methods** addCurrencyAndVerify()	The new currency should be added successfully and displayed in the table with the correct currency code and description.
6	Verify export to "User collection report"	1. Navigate to "Dispensary" module. 2. Click on "Reports" tab. 3. Click on "User Collection Report". 4. Select from date as 01-01-2020. 5. Click on "Show Report" button. 6. Click on "Export" button to download the report.	**Reference path** \PageObject\Pages**DispensaryPage** **methods** verifyExportUserCollectionReport()	The exported file should download with the name "PharmacyUserwiseCollectionReport_2025".
7	Verify Navigation to User Profile Page	1. Navigate to Homepage i.e https://healthapp.yaksha.com/Home/Index#/ 2. Click on the Admin dropdown. 3. Select the "My Profile" option.	**Reference path** \PageObject\Pages**AdminPage** **methods** verifyUserProfileNavigation()	Verify that the user is redirected to the page where "User Profile" heading is present.
8	Verify Patient Profile Picture Upload	1. Navigate to Patient module. 2. Select "Register Patient" tab. 3. Select "Profile Picture" tab (camera icon). 4. Click on the "New Photo" button. 5. Upload an image present in TestImage folder and click on the "Done" button.	**Reference path** \PageObject\Pages**PatientPage** **methods** uploadProfilePicture()	Verify that the uploaded image is displayed successfully in the patient's profile.

Mymedic automation using cypress

Test Case No.	Test Case Name	Test Steps to be performed	Path & Method Used	Expected Result
9	Verify TDS Percent update for an employee	1. Navigate to the "Incentive" module and then "Settings" tab. 2. Click on the "Settings" tab. 3. Click the "Edit TDS%" button within the located row. 4. In the "Edit TDS Percent" modal, enter the updated TDS% value. 5. Click on the "Update TDS" button. 6. Verify that the updated TDS% value is correctly displayed in the table.	**Reference path** \PageObject\Pages**IncentivePage** **methods** editTDSForEmployee()	The updated TDS% value is displayed correctly in the corresponding row of the table.
10	Verify the tooltip text on hover of Star icon in Laboratory	1. Navigate to Laboratory module. 2. Hovers over star icon and waits for tooltip to appear. 3. Verifies the visibility of the star icon and retrieves the tooltip text.	**Reference path** \PageObjects\Pages \ LaboratoryPage **methods** verifyStarTooltip()	Verifies the visibility of the star icon and retrieves the tooltip text. Result should be "Remember this date".

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

With **Cypress**, 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 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.
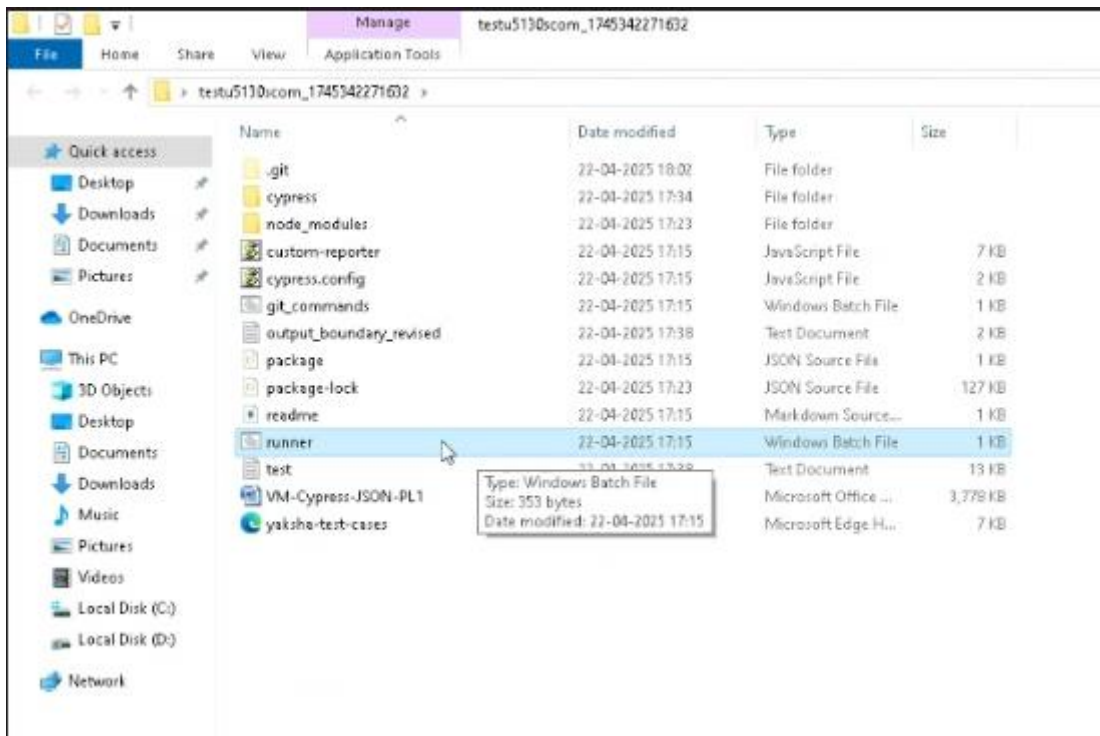
Execution Steps:
Steps for Execution:
Mymedic automation using cypress

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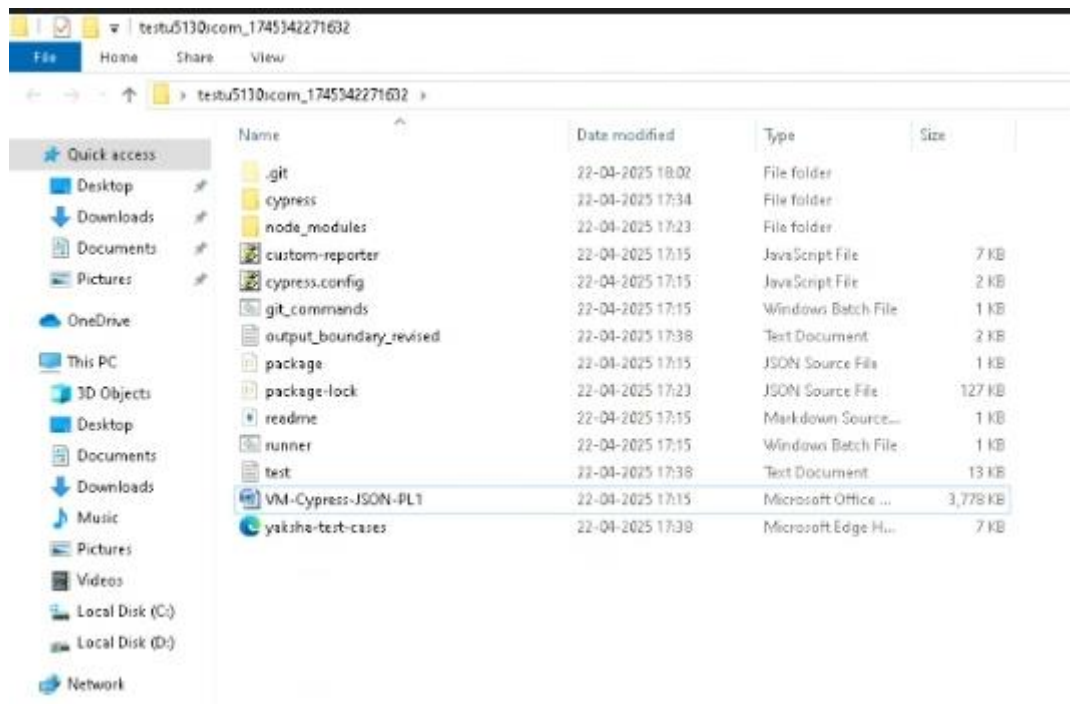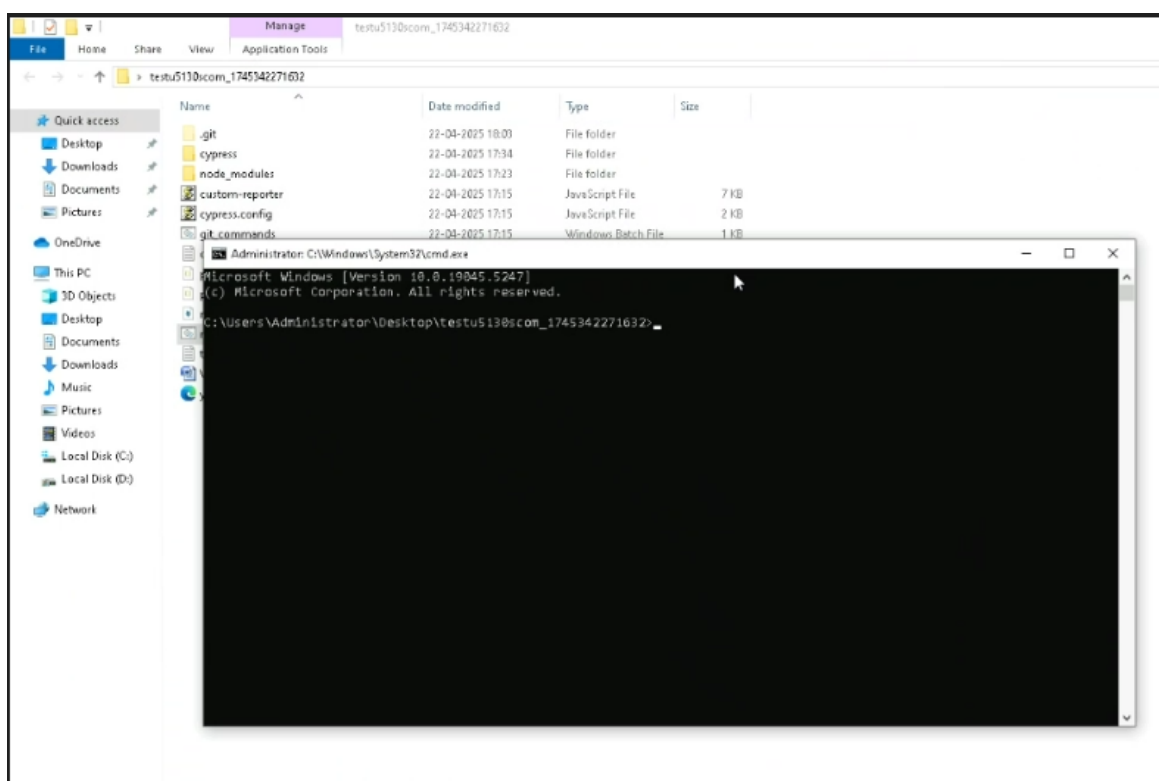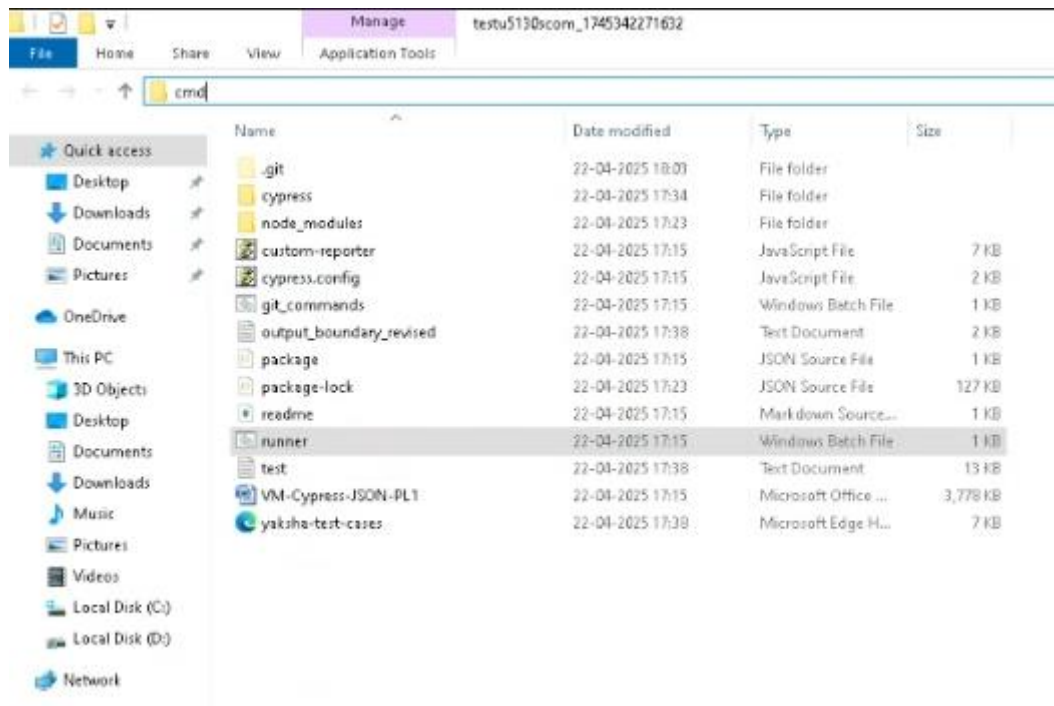

Mymedic automation using cypress

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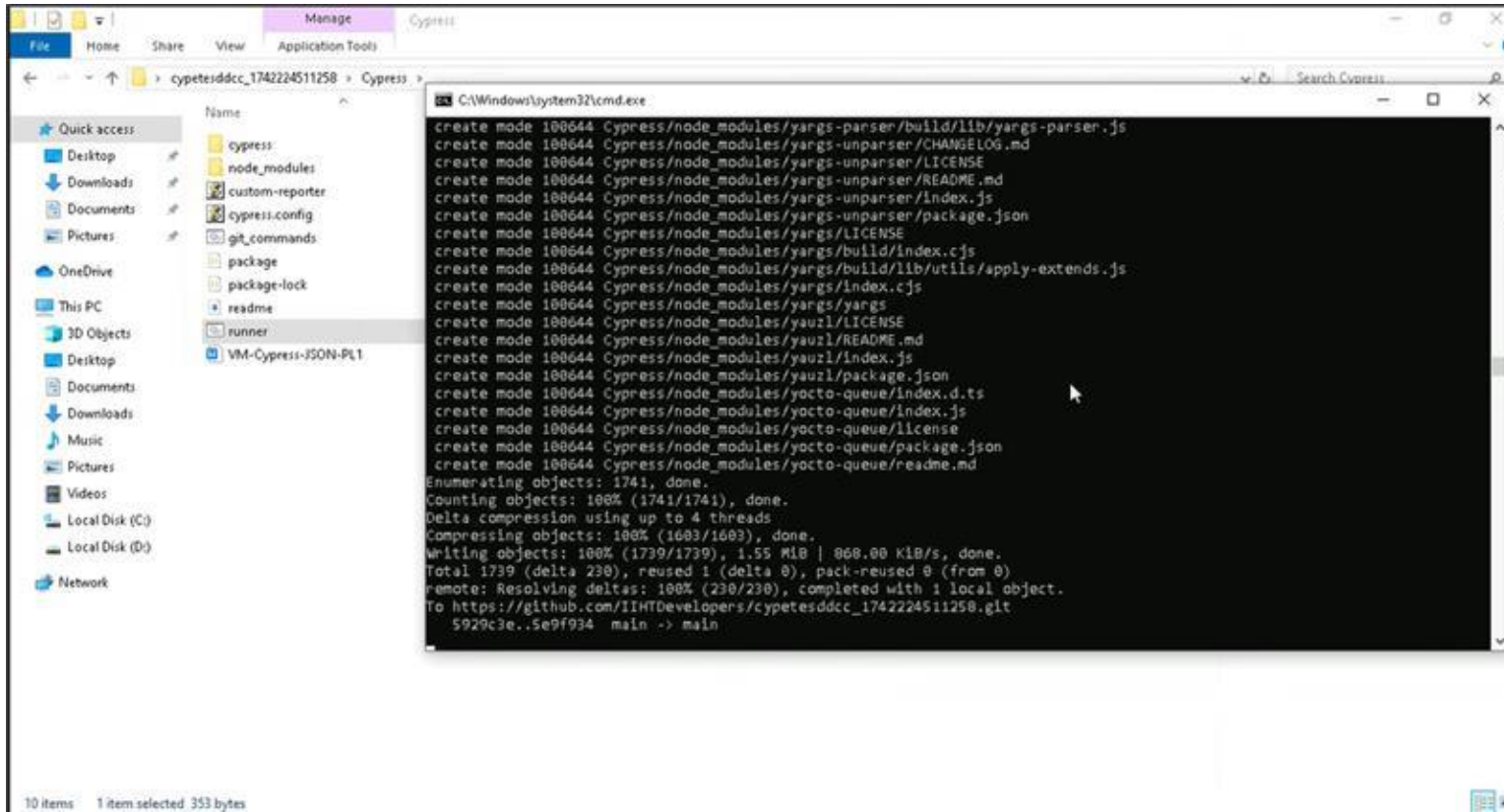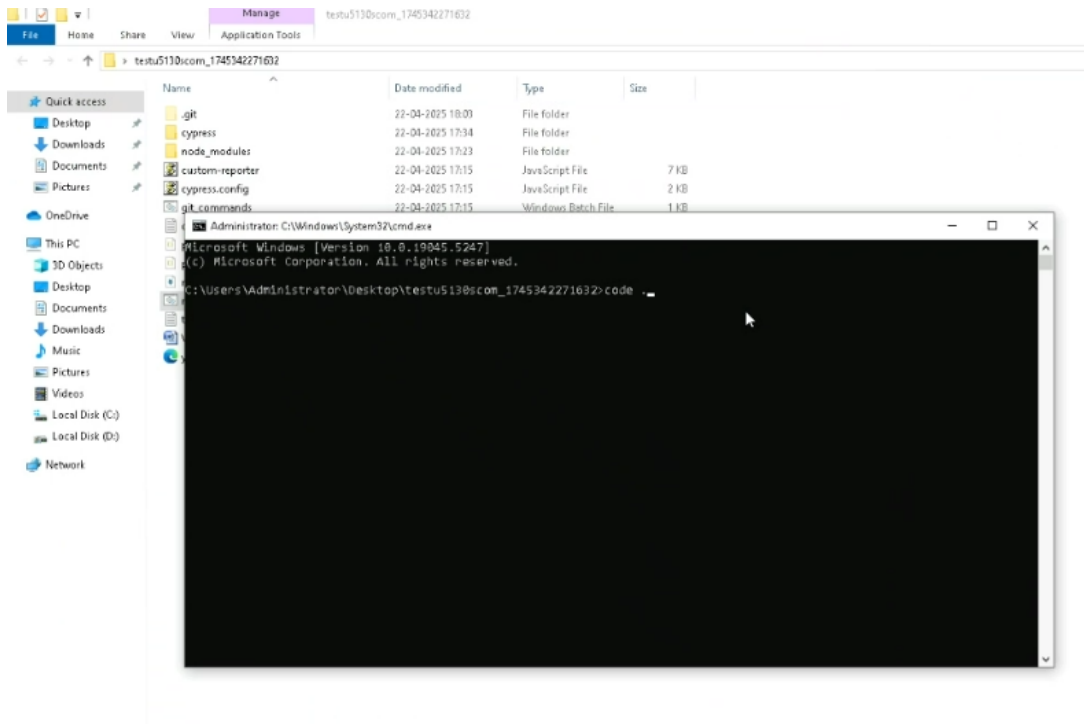

Mymedic automation using cypress

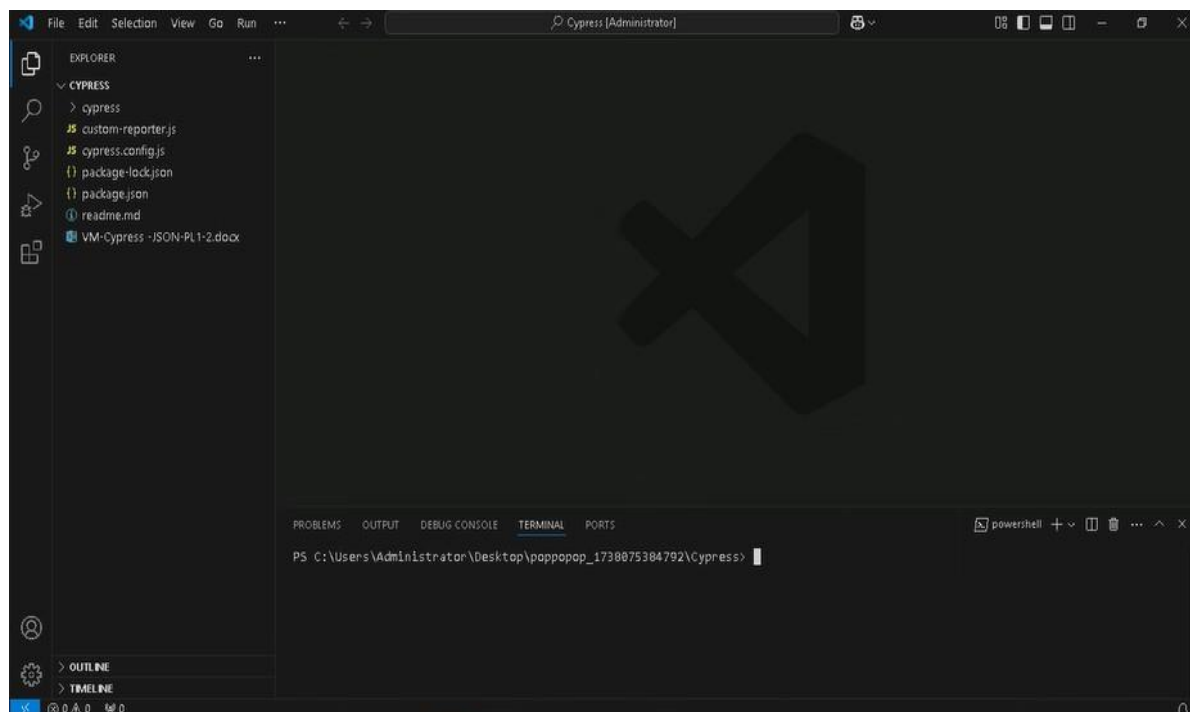**3. Open command prompt with its location using "cmd" in search bar**

Mymedic automation using cypress

4. **Open VS Code through cmd using below command:**

    code .

5. **Once VsCode is open. Please open the terminal and execute below given next commands:**

Mymedic automation using cypress

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

   npm install

2. **Run the following command to open the interactive Cypress Test Runner in the Cypress folder path:**

   npx cypress open

3. **Run the following command to run test cases in headless mode in the Cypress folder path:**

   npx cypress run

4. **Once you have executed the test cases. Now it is necessary to push your code to git. For this, please go inside the folder created on desktop with the email id you have used to login and then:**

   1. **Open gitbash**

   2. **Add all files**

3. **Commit the changes**

4. **Push the changes**

Mymedic automation using cypress

MINGW64:/c/Users/Administrator/Desktop/zxczxczxc_1732782665185

```
create mode 100644 Playwright/node_modules/zone.js/plugins/zone-patch-message-port/package.json
create mode 100644 Playwright/node_modules/zone.js/plugins/zone-patch-promise-test.min/package.json
create mode 100644 Playwright/node_modules/zone.js/plugins/zone-patch-promise-test/package.json
create mode 100644 Playwright/node_modules/zone.js/plugins/zone-patch-resize-observer.min/package.json
create mode 100644 Playwright/node_modules/zone.js/plugins/zone-patch-resize-observer/package.json
create mode 100644 Playwright/node_modules/zone.js/plugins/zone-patch-rxjs-fake-async.min/package.json
create mode 100644 Playwright/node_modules/zone.js/plugins/zone-patch-rxjs-fake-async/package.json
create mode 100644 Playwright/node_modules/zone.js/plugins/zone-patch-rxjs.min/package.json
create mode 100644 Playwright/node_modules/zone.js/plugins/zone-patch-rxjs/package.json
create mode 100644 Playwright/node_modules/zone.js/plugins/zone-patch-socket-io.min/package.json
create mode 100644 Playwright/node_modules/zone.js/plugins/zone-patch-socket-io/package.json
create mode 100644 Playwright/node_modules/zone.js/plugins/zone-patch-user-media.min/package.json
create mode 100644 Playwright/node_modules/zone.js/plugins/zone-patch-user-media/package.json
create mode 100644 Playwright/node_modules/zone.js/testing/package.json
create mode 100644 Playwright/node_modules/zone.js/zone.d.ts
create mode 100644 Playwright/node_modules/zone.js/zone_externs.js

Administrator@4f6d8dad9dbf5e2 MINGW64 ~/Desktop/zxczxczxc_1732782665185 (main)
$ git push
```

Mymedic automation using cypress