Cypress End-to-End Automation Project

Title, URL & UI Validation with Video Evidence

📌 Project Overview

This document explains how to **install, configure, and execute Cypress automation tests** used to validate **page title, URL, and UI text** on a real-world demo application (**OrangeHRM**).

The project demonstrates:

- Cypress installation & setup
- Headed and headless execution
- Video recording on successful test runs
- XPath usage
- BDD & TDD assertion styles
- Clean and maintainable test structure

🛠️ Prerequisites

- Node.js (LTS recommended)
- npm (comes with Node.js)
- Any code editor (VS Code preferred)
- Google Chrome browser

📥 Cypress Installation Steps

Step 1: Initialize npm

```
npm init -y
```

Step 2: Install Cypress

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

Step 3: Open Cypress Test Runner

```
npx cypress open
```

▶️ Running a Specific Spec File (Headed Mode)

To execute a single test file in **Chrome with UI visible**:

```
npx cypress run --spec "cypress/e2e/Spec_File.cy.js" --browser chrome --headed
```

📹 Save Test Execution as Video

To automatically **record and save videos even when tests pass**, add the following lines to **`cypress.config.js`**:

```
video: true,
videoCompression: 32,
videoUploadOnPasses: true,
```

This ensures:

- Visual proof of execution
- Better debugging

- Stronger portfolio & reporting evidence

⚡ Run in Headless Mode (Without Test UI)

Step 1: Add Spec Pattern to `cypress.config.js`

```
specPattern: "cypress/e2e/Spec_File.cy.{js,ts,jsx,tsx}"
```

Step 2: Run Cypress in Headless Mode

```
npx cypress run
```

✅ This avoids typing long commands like:

```
npx cypress run --spec "cypress/e2e/Spec_File.cy.js" --browser chrome
--headed
```

🔍 Using XPath in Cypress

Step 1: Install XPath Plugin

```
npm install cypress-xpath
```

Step 2: Update `commands.js` (last line)

```
/// <reference types="cypress" />
```

Step 3: Update `e2e.js` (last line)

```
require("cypress-xpath")
```

Step 4: Use XPath in Tests

```
cy.xpath("your-xpath-here")
```

📂 Test File Location

`cypress/e2e/Spec_File.cy.js`

🧪 Cypress Test Code

```
describe('template spec', () => {

  // Executes before each test case
  beforeEach(() => {

cy.visit('https://opensource-demo.orangehrmlive.com/web/index.php/auth
/login')
  });

  // Executes after each test case
  afterEach(() => {
    cy.wait(4000)
  });

  it('Validate Title', () => {
    cy.title().then(($tle) => {
      const tle = $tle
      cy.log(`Title is ${tle}.`)

      // BDD Assertions
      expect(tle).to.equal("OrangeHRM")
      expect(tle == "OrangeHRM").to.be.true
    })
  })

  it('Validate URL', () => {
    cy.url().then(($ul) => {
      const [ul, actual_ul] = [
        $ul,
```

```
"https://opensource-demo.orangehrmlive.com/web/index.php/auth/login"
      ]

      cy.log(`URL is ${ul}.`)

      // TDD Assertions
      assert.equal(ul, actual_ul)
      assert.notEqual(ul, actual_ul + ".")
      assert.isTrue(ul == actual_ul)
      assert.isFalse(ul != actual_ul)
    })
  });

  it('Validate Logo Text', () => {

    // XPath validation
    cy.xpath("//h5").then(($txt) => {
      const txt = $txt.text().trim()
      cy.log(txt)

      // TDD Assertion
      expect(txt == "Login").to.be.true
    })

    // BDD Assertions
    cy.xpath("//h5")
      .should("have.text", "Login")
      .should("contain", "Log")
      .should("include.text", "ogin")
      .and("be.visible")
  });

})
```

✅ Key Learnings & Highlights

- Implemented **BDD & TDD assertions** in Cypress
- Automated **UI smoke validations**
- Enabled **video evidence on test pass**
- Used **XPath for DOM-level validation**
- Executed tests in both **headed and headless modes**
- Followed **real-world automation best practices**

🔗 Project Link & Support

- 📌 **Project Repository:** Added in the media / references section
- 🙌 If this project helped you:
    - 👍 Like
    - 💬 Comment
    - 🔁 Share

Your support helps improve visibility and continuous learning 🚀