**Resolve India   
Test Strategy**

**Table of Contents**

1. **Introduction**

## Purpose

* The purpose of this test strategy document is to outline the overall approach to testing for the "**Resolve India**".
  1. Scope
* This document covers the testing activities and strategies for both manual and automated testing throughout the project lifecycle.

**2. Testing Approach**

2.1 Manual Testing

* Manual testing will be conducted for specific scenarios that require human observation and judgment.

2.2 Automated Testing (Cypress)

* Cypress with **TypeScript** and Cypress , Mocha will be the primary framework for automated testing, covering functional and regression test cases.
* **2.1 Test Scripting**: Test scripts will be written in **TypeScript** using the **Mocha framework**. The Page Object Model will be implemented to encapsulate web elements in cypress/helper files.
* **2.2 Test Execution**: Automated tests will be executed using Cypress in the CI/CD pipeline managed by Jenkins. The following command will be used for test execution: **npx cypress run**

This command will run all Cypress tests and generate test reports.

* **2.3 Reporting**: Test reports and screenshots will be stored in the ‘cypress/screenshots’ and ‘cypress/videos’ folder for easy access and analysis.

**3. Testing Types**

3.1 Functional Testing

* **Smoke Testing:** Basic functionality will be validated in a quick test to ensure the stability of the build**.**
* **Sanity Testing**: Core functionalities will be checked after specific changes to ensure no adverse impacts.
* **System Testing**: The entire system will be tested to ensure it meets specified requirements.

3.2 Regression Testing

* Regression testing will be conducted after each code change to ensure existing functionalities are not affected

**4. Testing Tools**

4.1 Cypress

* Test Scripting: Cypress will be used for scripting automated tests. Test scripts will be written in TypeScript using the Mocha framework, following the **Page Object Model (POM)** design pattern.
* Test Execution: Automated tests will be executed using Cypress in the CI/CD pipeline managed by Jenkins. The following command will be used for test execution: **npx cypress run**.
* Reporting: Test reports generated by Cypress will be stored in the `cypress/reports` folder and will be analysed for test results. The reports will be integrated into the Jenkins pipeline for visibility and monitoring.
* Web Elements in Helper Files: Web elements will be encapsulated in Cypress helper files following the Page Object Model (POM) design pattern. This approach enhances maintainability and reusability of web element locators across test scripts.

**5. Test Environment**

5.1 Browser Compatibility

* Testing will be performed on the latest versions of Chrome, Firefox, Electron.

5.2 Mobile Testing

* Mobile testing will be conducted on iOS and Android devices

(Other Team )

**6. Test Cases**

6.1 Test Case Design

* **Positive Test Cases**: Test cases covering expected behaviours
* **Negative Test Cases**: Test cases to validate error handling and unexpected scenarios

6.2 Test Case Execution

* **Headless Mode**: Automated tests will be executed in headless mode for continuous integration and faster execution. The headless mode is suitable for automated testing in non-GUI environments. Ex: npx cypress run –headless
* **Headed Mode**: Automated tests can also be executed in headed mode for debugging and development purposes. This mode provides a visual representation of the browser during test execution. Ex: npx cypress open

**7. Defect Tracking**

7.1 Defect Logging

* Defects discovered during testing will be logged in Azure DevOps

7.2 Defect Life Cycle

* The defect life cycle will be managed within Azure DevOps, encompassing stages such as New, Assigned, In Progress, Resolved, Verified, and Closed.

**8.** **Dynamic User Data**

* + To facilitate a variety of test scenarios and ensure that the application is resilient to different inputs, dynamic user data will be provided separately from a JSON file. This approach enables easy maintenance, updates, and scalability of test data.
  + Cypress/fixture /\*. JSON

**9.Regression Testing**

* npx cypress run --spec "cypress/e2e/Company Setup For Customer/Company\_Setup.cy.ts" --env ENV='dev' **,**
* npx cypress run --spec " cypress\e2e\Company Setup For Customer\Payroll\_Change.cy.ts " --env ENV='dev'
* If user want to give different environment should be change ENV=’uat’ or ENV=’qa’ or ENV=’dev’
* If want to change username and password, then  **“cypress\support\Utility.ts “ .**

**10. Test Execution Reports**

8.1 Test Execution Reports

* Test execution reports, including screenshots and videos, will be available in the following directories:

- Screenshots: ‘cypress/Screenshot’s

- Videos: ‘cypress/videos’

8.2 Mocha reports – html Reports

“ cypress\reports\html\index.html”