Test Automation Report

1. Executive Summary

This solution demonstrates a comprehensive, full-cycle UI and API test automation framework built using Robot Framework. The solution is modular, scalable, and maintainable, with a well-structured directory layout(tests/, resources/, variable/). It supports end-to-end test scenarios for OrangeHRM, API and Salesforce Lightning UI, ensuring robust validation.

2. Test Strategy

Framework Design

- Modular Robot Framework structure with clear separation of:
    - tests/: Robot test suites
    - resources/: Reusable keywords
    - variables/: Centralized test data and configurations

- Structure
    - Follows keyword-driven approach with layered design.
    - Promotes reusability and maintainability of code .
- UI Automation
    - Built using SeleniumLibrary.
    - Utilizes dynamic waits, XPath, and CSS selectors for handling Salesforce Lightning and OrangeHRM DOMs.

- API Automation

- Built using RequestsLibrary
- Focuses on:
  - Status code validation (e.g., 200, 201)
  - JSON schema validation (manual or via Postman)
  - Response body field assertionsApproach

3. Tools & Versions

Tool	Version	Purpose
Python	3.12.7	Core scripting language.
Robot Framework	7.3.1	Test automation framework.
Selenium Library	6.7.1	Web UI automation.
Requests Library	0.9.7	API test automation.
Chrome Driver	135+	Browser automation (headless support).
Postman	v11	Manual API validation & benchmarking.

4. Scenario Coverage

Scenario 1: OrangeHRM Automation with API + UI Integration

Implement end-to-end HR process automation demonstrating robust keyword design with integrated API validation.

UI Test Flow

1. Login
    - Authenticate using admin credentials (Admin/admin123).
    - Verify successful dashboard navigation.
2. Add Employee
    - Navigate to PIM - Add Employee.
    - Fill in mandatory details (First Name,Middle Name Last Name, Employee ID).
    - Save and confirm record creation.
3. Search & Validate
    - Use the employee list table to search for the newly added employee.
    - Validate presence in the system via dynamic table checks.
4. Edit Employee
    - Update the employee's middle name via the edit panel.
    - Re-validate the updated record in the employee list.
5. Delete Employee
    - Initiate deletion and confirm removal.
    - Verify absence from the employee list post-deletion.

API Test Flow (reqres.in Mock API)

1. GET User List
    - Endpoint:GET /user?page=2
    - Validations:
        - Status code.
        - Response schema.
2. POST Create User
    - Endpoint: POST /user
    - Payload:
    - { "name": "Rajkumar", "job": "QA Engineer" }
    - Validations:
        - Status code.
        - Response body matches the input payload (name, job).
        - Presence of system-generated fields (id, created).

Scenario 2: Salesforce Trailhead Playground (UI Testing)

Demonstrate ability to handle dynamic Lightning UI components, custom objects, and end-to-end validations in Salesforce Trailhead Playground.

Test Case Flow

1. Login to Salesforce Trailhead Playground
    - Authenticate using valid credentials.
    - Verify successful landing on the Home page.
2. Navigate to Custom Object ("Accounts")
    - Navigate to the "Accounts" tab.

3. Add a New Record
    - Click the "New" button.
    - Fill in required fields (e.g.Account Name).
    - Save the record and confirm successful creation.
4. Validate Record via Search/List View
    - Search for the newly created record using global search or list view filters.
    - Verify the record appears in search results with correct field values.
5. Edit & Re-validate Record
    - Open the record in Edit Mode.
    - Modify a field (e.g.update Account Name).
    - Save changes and confirm the update is reflected in the UI.

5. Logs & Artifacts

- results/screenshots - Key test steps (pass/fail states).
- results - Robot Framework outputs (log.html,report.html,output.xml).
- Postman Collection - Exported mock API tests for reference.

6. Known Issues & Mitigations

Salesforce Lightning Web Components (LWC)

Issue:

- Dynamic LWC tags with custom attributes resist traditional Selenium locators .
- Click actions often fail due to shadow DOM.

Mitigation:

- Leveraged JavaScript injection via Execute JavaScript (e.g.,document.evaluate) to directly interact with obscured elements.

OrangeHRM Volatility

Issue:

- Unpredictable changes in UI (e.g., language auto-switching, downtime of public demo instances).

Mitigation:

- Pre-Execution Checks: Added suite-level validation to confirm site availability and default language (English).

7. Key Learnings

Framework Fundamentals

- Building modular tests: Structured test suites using resources/ for reusable keywords.
- Creating Custom Keywords:Developed maintainable keywords.

Real-World Salesforce Testing

- Working with Lightning Components:Gained hands-on experience with LWC interactions.

API Test Automation

- First Steps in API Validation:
    - Automated status code verification (200, 201, 404).
    - Implemented basic JSON schema checks.
- From Postman to Automation:
    - Translated manual API tests from Postman into Robot scripts
    - Built initial experience with API testing best practices