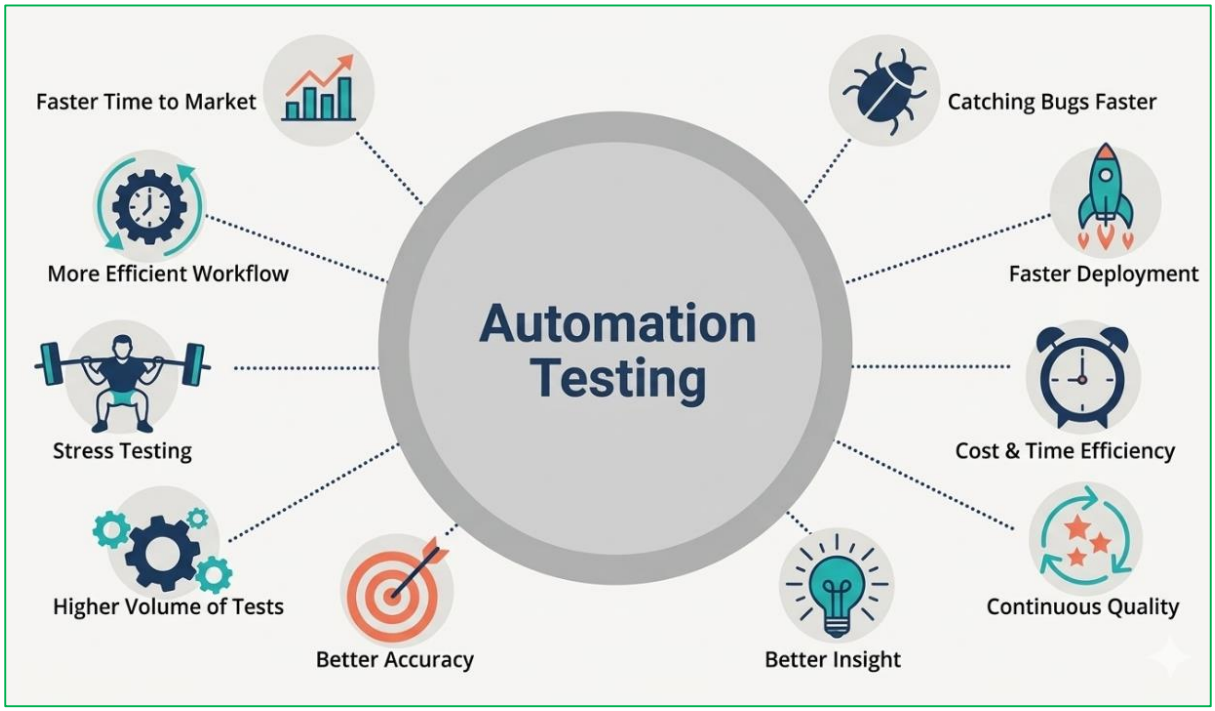
End to End Automation using Playwright can automate end-to-end UI workflows, hybrid UI-API data setups, and regression validations across Oracle Fusion Cloud—handling both classic ADF screens and modern Redwood UI pages reliably.

Because Oracle Fusion relies heavily on dynamic IDs, shadow DOM (in Redwood), and asynchronous ESS background jobs, Playwright's auto-waiting and network-interception capabilities make it particularly well-suited for ERP test automation.



1. Core Functional & Transactional Workflows

You can automate full Order-to-Cash (O2C), Procure-to-Pay (P2P), and Record-to-Report (R2R) cycles across modules:

- **Master Data Creation & Maintenance:** Automating the creation and approval of Suppliers, Customers, Items, and Employees using data-driven inputs (Excel, CSV, or JSON).
- **Financials (AP, AR, GL):** Creating and validating AP Invoices, processing Payments, generating AR Invoices/Receipts, entering GL Journals, and running period-close checklists.
- **Supply Chain & Procurement (SCM):** Raising Requisitions, converting them to Purchase Orders (POs), entering Sales Orders in Order Management (OM), and performing Receiving or Ship Confirmations.
- **Human Capital Management (HCM):** Automating "Hire an Employee" flows, terminations, transfers, and Absence Management requests.

2. ESS Job Execution & Monitoring

- **Submitting Scheduled Processes:** Navigating to *Tools > Scheduled Processes*, selecting a specific ESS job (e.g., *Import Journals*, *Load Interface File for Import*), and filling dynamic parameter popups.
- **Polling Job Status:** Using custom retry loops to refresh the process table until the status changes from Running or Wait to Succeeded, and automatically downloading the output/log .txt or .pdf files for assertion.

3. Hybrid UI + REST/SOAP API Testing

Playwright includes a built-in APIRequestContext, allowing you to combine backend speed with frontend verification:

- **Fast Test Data Setup:** Calling Oracle Fusion REST APIs to create base records (like a draft invoice or supplier site) in milliseconds before opening the UI to test approval workflows.
- **Backend Data Validation:** Querying Fusion REST endpoints after a UI action (e.g., submitting a Sales Order) to verify that database statuses and flexfield attributes updated accurately.
- **FBDI / Integration Triggers:** Triggering Oracle Integration Cloud (OIC) endpoints or uploading FBDI .zip files via the *File Import and Export* UI, then verifying the imported records on the target screen.

4. Role-Based Access Control (RBAC) & Security Testing

- **Multi-Context Session Testing:** Spinning up isolated browser contexts simultaneously to test maker-checker workflows (e.g., User A submits a Purchase Order in one context while Manager B approves it via the Bell Notification icon in another—without logging out).
- **Negative Security Assertions:** Logging in with custom read-only or restricted roles to verify that restricted tasks, menus, or action buttons (like *Post* or *Approve*) are hidden or disabled.

5. Redwood UI & Cross-Browser Regression

- **Shadow DOM Traversal:** Navigating modern Oracle Redwood pages (which use Oracle JET and web components) natively without brittle XPath workarounds.

- **Visual & Layout Regression:** Capturing baseline screenshots (expect(page).toHaveScreenshot()) after quarterly Oracle Cloud updates (26A, 26B, etc.) to catch broken layouts, overlapping fields, or missing Descriptive Flexfields (DFFs).



Practical Caveats in Oracle Fusion

- **Automated Reporting & Evidence:** Capturing step-by-step screenshots, full video recordings, and Playwright Trace Viewer logs to serve as audit-ready UAT sign-off evidence.
- **Avoid Dynamic IDs:** Oracle ADF generates dynamic element IDs (e.g., pt1:_FOr1:1:_FOSritemNode...) that break across patches. Always anchor locators using ARIA roles (getByRole), visible labels (getByLabel), or stable partial attributes.
- **MFA / SSO Handling:** For environments protected by Oracle IDCS or OCI IAM with Multi-Factor Authentication, automate login once via storageState to save session cookies and reuse them across test suites.
- If you'd like to dive deeper, let me know: