Automation Strategy :

QA Craft Growth
Automation challenge activity

The document outlines an "Automation challenge activity" for QAs at Huge, aiming to enhance their automation skills. The objective is to equip QAs to effectively contribute to projects by resolving a practical automation testing challenge. This is a mandatory activity for all QAs at Huge and is designed to be a realistic QA scenario emphasizing critical thinking, automation scripting, research, and ownership.

E-commerce Site — Cypress Approach

Automation Strategy for

laboratoriodetesting.com

Project Summary

I am the sole QA on an e-commerce project (laboratoriodetesting.com), currently in Sprint 6 of 18, with 3 developers. Key functionalities include:

- Login with existence user
- View Order History
- Add Product

Given the reported bugs in Registration and Checkout , and the upcoming changes in payment methods, checkout, and homepage design, the automation strategy should prioritize stabilizing existing critical flows while preparing for future enhancements.

2. Critical Paths (to automate first)

Scenario 1: Login with Existing User

Why it matters:

- Entry point to personalized user actions like order history, checkout, saved preferences.
- Essential for session validation, role-based access, and gating authenticated areas.

What to validate:

- User can successfully log in with valid credentials.
- Token/session is created and persisted.
- Error messaging works for invalid credentials.
- Redirect behavior (e.g., to `/dashboard` or `/products`).

Technical Focus:

- Backend auth API (token, cookies).
- Frontend session handling (localStorage, JWT, etc.).
- Form validation.

Business Risk if Broken:

- Returning users are locked out → lost sales.
- Negative perception of reliability.
- Increased customer support volume.

Scenario 2: View Order History

Why it matters:

- It proves that the user's purchases were stored correctly and are retrievable.
- A key point of user trust, accountability, and post-purchase confidence.

What to validate:

- Orders made in previous sessions are displayed.
- Details like product name, date, total, and payment method are correct.
- Orders persist even after user logs out and returns later.
- UI handles multiple orders, no orders, or failed orders gracefully.

Technical Focus:

- API call to retrieve orders.
- Mapping of backend data to UI.
- Pagination or load-more logic (if present).

Business Risk if Broken:

- Users may assume the order was not placed.
- Duplicate purchases or refund claims.
- Undermines trust in the brand.

Scenario 3: Add Product(s) to the Shopping Cart

Why it matters:

- It is the first step in the sales funnel. If users can't add products to cart, no purchase is possible.
- Acts as a bridge between product exploration and revenue conversion.

What to validate:

- Adding one or multiple products reflects accurately in the cart.
- Cart shows correct item names, prices, and quantities.

Technical Focus:

- Cart storage mechanism (localStorage, cookies, session).
- Sync with backend (if cart is stored in DB).
- Dynamic DOM rendering on the cart page or mini-cart widget.

Business Risk if Broken:

- Abandoned sessions and direct revenue loss.
- Users may abandon if cart feels unreliable or inconsistent.
- Cart mismatch issues may lead to customer complaints or incorrect charges.

3. Tools & Frameworks

Tool	Purpose
Cypress	E2E testing framework
Cypress Studio / POM	Scalable test structure
Cypress.env.json	Secure test data
GitHub	Code hosting and version control
Faker.js	Generate unique user data
CI Option (Bonus)	GitHub Actions for CI runs
Browser	Chrome (desktop only)
Language	JavaScript
Editor	Visual Studio Code

4. Test Scenarios to Implement

Scenario 1: Login with existence user

Reasoning: Login is a fundamental entry point to the e-commerce site, and ensuring it works correctly is crucial for any user interaction beyond Browse. Basic navigation after login confirms the user can access key areas of the site.

Steps:

1. Visit `/Login with good and incorrect credentials`
2. Fill the form with data.
3. Submit logging
4. Verify success via redirect/message.

Expected Result: User successfully logs in; session/token is active.

Scenario 2: Check order History

Reasoning: Order History is the **final link** in the e-commerce transaction chain. Once a user completes a purchase, they expect to:

- See a **record of what they bought**
- Track shipping or re-download digital goods
- Verify the Order Detail

Failure here directly impacts **user trust**, support ticket volume, and potential **chargebacks** or refund disputes.

Steps:

1. Login with val credentials.
2. Visit `/Mi Cuenta`
3. Select any number order ex. (Order **# 68ea6, Order # 7fdc0, Order # e2e68)**
4. Go to `/Ver detalles`, verify item or items and price.
5. Click "Checkout" and input test card data. (working progress)...
6. Complete payment. (working progress).. THIS DOESN'T WORK BECAUSE THE ORDER IS ALREADY PLACED

Expected Result: Checkout is successful. Check order History

Scenario 3: Add products to shopping cart

Reasoning:
Business Value The shopping cart is the core conversion point of any e-commerce site. If users can't reliably add items to their cart, they can't buy — this directly results in:

- Lost sales
- Abandoned sessions
- Increased bounce rates

Ensuring the cart works correctly is **critical to revenue generation**.

Steps:

1. Login with a previously used account.
2. Select a product and add it to the cart
3. Validate success added item to the cart

Expected Result: Correct add item to the shopping cart

5. Suggested Folder Structure (POM)

pgsql
CopyEdit

```
cypress/
├── e2e/
|   ├── Login.cy.js
|   ├── orderHistory.cy.js
|   └── addproduct.cy.js
├── fixtures/
|   └── users.json
├── support/
|   ├── commands
            commands.js
|   └── elements/
|       ├── addelements.js
|       ├── historyements.js
|       ├── loginelements.js
```

6. Good Practices & Strategy Rationale

- **POM architecture** ensures maintainability.
- **Retry logic** (e.g., for flaky UI elements).
- **Error screenshot/video recording** for debugging.
- **Custom Cypress Commands** for login & add-to-cart for reuse.
- **Test tagging** (smoke, regression) for CI scalability.

Final Deliverables Recap

- **Automation Strategy** (this doc)
- **GitHub Repo** with at least:
    - 3 test cases
    - Page Object implementation
    - Cypress project config