QA Engineer Assignment

Technical Assessment - Aajil

1 Part 1: Manual Testing

1.1 Target Application

- **URL:** https://demoblaze.com/index.html
- **Scenario:** You are the QA for the "Checkout Functionality" of the DemoBlaze website. Your goal is to ensure a user can successfully purchase an item.

1.2 Task 1: Test Case Development

Requirements:

- Write detailed test cases for the Cart functionality
- Include at least 10 Positive and 10 Negative test cases
- End goal is to have a purchase made through the website
- Provide test cases you executed along with expected results and actual results
- Think critically and compare the checkout to large e-commerce websites like Amazon

1.3 Task 2: Bug Reporting

Requirements:

- Perform exploratory testing on the website
- If you find bugs, report them
- If you do not find actual bugs, simulate 2 bug reports for hypothetical issues

2 Part 2: Test Automation

2.1 Target Application

- **URL:** https://blazedemo.com/
- **Note:** Login/Registration not required
- **Goal:** The company has created a Flight purchase page. They want automation created where each time a deployment is done, automation will execute some tests.

2.2 Technical Requirements

- **Language:** Java, Python, or JavaScript/TypeScript
- **Framework:** Selenium WebDriver, Cypress, or Playwright
- **Pattern:** Use Page Object Model (POM) design pattern

2.3 Automation Scenario: "Purchase Flight"

Create a function called `purchaseEndToEnd`. The function should take 3 optional parameters:

- `deptCity` (String): Departure city (e.g., "Paris"). If no value is provided, take random value each time.
- `desCity` (String): Destination city (e.g., "Buenos Aires"). If no value is provided, take random value each time.
- `flightSeq` (int): Preferred flight number (e.g., 3 means 3rd flight from Paris to Buenos Aires). If no value is provided, take random value each time.

2.3.1 Function Requirements

1. Input correct flight data and select the correct flight sequence
2. Make the purchase with dummy user data generated randomly each time
3. Once purchase is made, validate:
    - Status is "PendingCapture"
    - Price is greater than $100.00
4. If any validation fails, the function should halt and show reason in logs
5. Keep inputs sanitized and throw an exception if any input by user is wrong

2.3.2 Test Execution Requirements

Execute the function with the following inputs:

1. Boston, Berlin, 2
2. No inputs (all parameters random)
3. Boston, Boston, 1
4. Paris, Berlin, 0
5. Choose any inputs of your choice

2.4 Technical Expectations

- **Wait Handling:** Use explicit waits (no `Thread.sleep` or hard-coded pauses unless necessary)
- **Exception Handling:** Exception handling is mandatory
- **Code Quality:** Clean, readable code with comments explaining complex logic
- **Assertions:** Use proper assertions (e.g., TestNG, JUnit, Chai, Pytest, or built-in asserts) to validate UI states
- **Reporting:** Generate a simple execution report (HTML report or Console output)
- **Assumptions:** If you are making an assumption, mention them as comments in code

3 Part 3: Deliverables

Please submit the following via email or a shared folder link:

3.1 Manual Testing Artifacts

- Test Cases (Excel/Google Sheet format)
- Bug Reports (PDF/Excel format)

3.2 Automation Project

- A link to a public GitHub repository containing your code
- A `README.md` file with clear instructions on:
  - How to set up the environment
  - How to run the tests

4 Evaluation Criteria

- **Manual Testing:**
  - Clarity of test steps
  - Coverage of edge cases
  - Quality of bug reports
- **Automation:**
  - Proper use of POM (Page Object Model)
  - Effective locator strategies (XPath/CSS)
  - Handling of synchronization (waits)
  - Code maintainability
- **General:**
  - Attention to detail
  - Documentation skills

Note: This assessment is designed to evaluate both manual and automation testing skills. Please ensure all deliverables are well-documented and follow best practices.