User

□ SDET Interview Assignment □Objective: This assignment is designed to evaluate your ability to plan and execute a comprehensive testing strategy for a web application. It will assess your analytical skills, understanding of testing principles, ability to write effective test cases, and familiarity with automation tools. □ Webpage Under Test URL: Analytics Vidhya You will test this webpage as an external black-box tester. No access to source code or internal APIs will be provided. ✓ Deliverables Please organize all submissions into a single folder (or GitHub repo) with the following components:

- 1. □ Test Strategy Document

Create a document outlining your test approach. It should include:

- Overview: Your understanding of what the webpage does (based on visual inspection and behavior).
- Scope of Testing: What is in-scope and out-of-scope.
- Assumptions: Mention any assumptions made due to lack of backend access or documentation.
- Types of Testing Covered:
- Functional Testing
- UI/UX Testing
- Compatibility Testing (browsers/devices)
- Responsive Design Testing
- Negative Testing
- Accessibility Testing
- Performance Testing (basic strategy)
- Security (basic surface-level checks)
- Test Environments: List the browsers/devices/OS you plan to test on.
- Tools You Would Use: Mention any tools you'd choose for manual or automated testing and justify your choice.
- 2. ✓ Test Cases

Document at least 10-15 detailed test cases covering different aspects of the page. Include:

- Test Case ID
- Title
- Description
- Preconditions
- Test Steps
- Expected Result
- Actual Result (if executed)
- Priority/Severity

Use a format like Excel, CSV, or Markdown.

- 3. □ Bug Report (Optional Bonus)

If you find any functional or visual bugs, include a short bug report for each:

- Bug title

- Steps to reproduce
- Actual vs. expected behavior
- Severity
- Screenshot (if applicable)

You may include this in the same test case document or separately.

- 4. □ Automation Samples (Mandatory)

You must include both frontend and backend automation test samples for the webpage using tools/frameworks of your choice. Provide 2–3 sample test scripts for each. Frontend options (e.g.):

- Selenium (Java/Python)
- Playwright (JS/TS/Python)
- Cypress (JavaScript)

Backend options (e.g.):

- REST Assured (Java)
- Postman (with Newman CLI)
- Python requests + pytest

The above are some of the options. You are free to use the framework/language/library/tools that suits you best. Include:

- A README.md with setup instructions
- Well-documented code with inline comments and function/method descriptions
- Use of assertions and best practices (e.g., Page Object Model for frontend, structured API test suites for

backend) □ Code documentation is mandatory. Your code should be clean, modular, and easy to understand, with appropriate comments and structure. ⏱ Estimated Time & Submission Deadline

- Expected time commitment: 6–8 hours
- Submit within 72 hours of receiving this assignment.

□ Evaluation Criteria We'll evaluate based on:

- Thoughtfulness and clarity of the test strategy
- Coverage and quality of test cases
- Ability to think through edge cases and negative scenarios
- □□Code documentation and readability

□ Submission Instructions

- Package your files in a ZIP folder or upload them to a GitHub repository.
- Email to: [varsha@analyticsvidhya.com, sangeeta@analyticsvidhya.com]
- Subject: SDET Assignment - [Your Full Name]
- If using GitHub, include the repo link in your email. based on this details test this

1. Test Strategy Document

Overview The Analytics Vidhya webpage serves as a platform for data science enthusiasts, providing resources, articles, and community engagement. The site likely includes features such as user registration, article browsing, and community forums.

Scope of Testing

- In-Scope:

- Functional testing of user interactions (registration, login, article navigation) - UI/UX testing for layout and design consistency - Compatibility testing across different browsers and devices - Responsive design testing on various screen sizes - Negative testing for invalid inputs - Accessibility testing for compliance with standards (e.g., WCAG) - Basic performance testing (load times, responsiveness) - Surface-level security checks (e.g., input validation)

- Out-of-Scope:

- Backend testing (APIs, database) - Detailed security testing (penetration testing) - Load testing beyond basic performance checks

Assumptions

- No access to backend APIs or source code.
- The webpage is assumed to be functional based on visual inspection.
- User roles and permissions are not defined; testing will assume a general user perspective.

Types of Testing Covered

- Functional Testing: Verify that all features work as intended.
- UI/UX Testing: Assess the visual design and user experience.
- Compatibility Testing: Test across major browsers (Chrome, Firefox, Safari, Edge) and devices (desktop, tablet, mobile).
- Responsive Design Testing: Ensure the site adapts to different screen sizes.
- Negative Testing: Test invalid inputs and error handling.
- Accessibility Testing: Check for compliance with accessibility standards.
- Performance Testing: Measure load times and responsiveness.
- Security Testing: Conduct basic checks for vulnerabilities.

Test Environments

- Browsers: Chrome, Firefox, Safari, Edge
- Devices: Desktop (Windows, macOS), Mobile (iOS, Android)
- OS: Windows 10, macOS, Android, iOS

Tools You Would Use

- Manual Testing: Browser Developer Tools for UI/UX and responsive testing.
- Automation Testing:

- Frontend: Selenium with Python for UI testing. - Backend: Postman for API testing.

- Accessibility Testing: Axe or WAVE for automated accessibility checks.
- Performance Testing: Google PageSpeed Insights for basic performance metrics.

---

2. Test Cases

Test Case ID	Title	Description	Preconditions	Test Steps	Expected Result	Actual Result	Priority/Severity
TC001	Verify Homepage Load	Ensure the homepage loads successfully.	None	1. Open the homepage URL.	Homepage loads without errors.	High	
TC002	User Registration	Test user registration functionality.	None	1. Click on 'Sign Up'. 2. Fill in the form. 3. Submit.	User is registered and redirected.	High	
TC003	Article Navigation	Verify navigation to articles.	User is on the homepage.	1. Click on an article link.	Article page loads successfully.	Medium	
TC004	Invalid Login	Test login with invalid credentials.	None	1. Click on 'Login'. 2. Enter invalid credentials. 3. Submit.	Error message displayed.	High	
TC005	Responsive Design	Check responsiveness on mobile devices.	None	1. Open the homepage on a mobile device.	Layout adjusts to fit the screen.	Medium	
TC006	Accessibility Check	Verify accessibility features.	None	1. Use accessibility tool (e.g., Axe).	No critical accessibility issues found.	Medium	
TC007	Performance Check	Measure page load time.	None	1. Open the homepage. 2. Measure load time using tools.	Load time is under 3 seconds.	Medium	
TC008	Negative Input Handling	Test form validation with empty fields.	None	1. Click on 'Sign Up'. 2. Leave fields empty. 3. Submit.	Error messages for required fields.	High	

| TC009 | Cross-Browser Compatibility| Verify site functionality on different browsers. | None | 1. Open the site on Chrome, Firefox, Safari, Edge. | Site functions correctly