Test Plan - Question

Draft a brief test plan that includes:

1. Types of functional tests you would prioritise.
2. A rationale for your chosen testing approach.
3. Tools/frameworks you would consider for automation.

Test Plan

Objective

This is to ensure the core functionalities for Demoblaze site (product categorizing, product browsing, shopping cart management, order placement and checkout) work correctly, provide a smooth user experience, and maintain stability across browsers and devices.

Test Types

Test Type	**Why?**	**Entry Criteria**	**Exit Criteria**
Smoke Testing	Quickly verify core features like homepage, product listing, sign in, login and cart	QA environment is setup with necessary hardware, operating system and supporting software	Smoke test is successfully completed and defects found during smoke test are tracked and closed.
Functional Testing	Validate main workflows, browsing products, adding to cart, placing orders	Smoke testing has been successfully completed, and build is accepted for testing.	All functional requirements verified; defects logged/resolved
Regression Testing	Ensure new changes (site code changes/updates or change requirements) don't break existing flows	Functional tests stable	No critical/high-severity defects after updates
UI Testing	Check UI consistency, layout and readability across pages	Functional pages deployed	Pages render correctly as per design; no layout issues
Browser Compatibility Testing	Verify the site functionality on different browsers (Chrome/FF/Safari)	Functional tests stable	Site renders and works correctly on target browsers
Integration Testing	Verify interactions between modules: product selection -> add to cart -> place order	Integration components have been connected and available for testing.	No showstopper and high severity defects are open, and all known issues are documented.

Test Approach

Testing will begin with smoke testing and functional tests to ensure the core flows work status. After the site updates, regression will be performed to ensure that the existing features are working as usual and not broken. UI testing will be performed to test the responsiveness and check layout, then the browser compatibility will be performed to ensure a consistent user experience.

Test cases will be prioritized so that critical or high-risk tests (features and functions) are performed first (Risk Based Testing). Problems or issues found will be reported via a project management tool. Problems with the highest severity (e.g. showstopper) will be escalated immediately.

While the aspiration is to execute all these test cases successfully, if many critical defects are encountered, testing will be halted until a new build is deployed (after which testing will recommence). It is important to note however, that at the end of the final cycle, 100% test coverage will have to be achieved.

During the Functional and Regression Testing phase, QA team will execute below tasks:

- Smoke test suite will be executed first to ensure that the system is testable.
- Functional test execution will be started by executing the high priority test cases first.
- Defects will be tracked in defect management system to ensure defect closure.
- Test cases with high impact and the impacted areas after defect fixes will be tested during the regression test execution phase.

Automation Testing will be performed for repetitive tests with regression suite using Selenium/ Cypress/ Playwright improving efficiency and reliability.

Recommended Tools/Frameworks for Automation

- Playwright
  A feasible modern automation framework for testing web applications. It supports multiple browsers such as, Chromium, FF, WebKit. Playwright handles dynamic web pages efficiently.

- Cypress
  Cypress is a JavaScript-based automation tool focused on E2E testing for web applications. It runs directly in the browser and allows real-time interaction and debugging.

- Selenium WebDriver
  Selenium widely used automation tool, and it allows controlling browsers programmatically. Selenium supports multiple programming languages such as, C#, Java, Python etc. Supports Junit and TestNG.

- TestNG
  Manage test configurations and generate reports. Consist with annotations, detailed reports and parallel executions.

- POM Design Pattern
  Structured framework coding for maintainable and reliable testing approach. Separates the test logic from the UI structure.