Test Plan

To effectively test the **demoblaze.com website**, it's essential to cover various aspects such as

1. **Functional Tests** 
   1. Manual Tests
   2. Automated Tests
2. **Non-Functional Tests** 
   1. Performance
   2. Security
   3. Usability
   4. Accessibility.

Prioritization

1. **Functional Tests (P1)**
   1. Manual Tests (P1.1)
   2. Automation Test (P1.2) – But can be done in parallel with development team using Sift Left style of Automation.
2. **Performance Tests (P2)**
   1. Load Test (P2.1)
   2. Stress Test (P2.2)
   3. Soak Test (P2.3)
   4. Spike Test (P2.4)
3. **Usability Tests (P3)**
4. **Security Tests (P4)** – Can be done only once per major release.
5. **Accessibility Test (P5)** – Can be done only once at the first release and then only when any new UI elements are added or updated.

Tasks Breakdown

Below is a list of **tasks** along with **estimated durations and prioritization** based on importance:

1. **Functional Testing:**
   1. **Manual Tests**
      1. **Task: To create and document manual tests to include below functionalities (1 day)**
         1. Verify navigation: Ensure all links, buttons, and menus navigate users correctly.
         2. Verify Log in and sign out flow for users.
         3. Verify new user Sign up flow.
         4. Verify Products can be loaded for all categories.
         5. Verify user can view products in new screen.
         6. Verify Products can be added to cart.
         7. Verify Products can be deleted from cart.
         8. Verify user can place order and complete purchase.
      2. **Task: To Execute the manual tests when demo blaze release is available to test on QA environment (2 days)**
   2. **Automated Tests**
      1. **Task: To create Automated test for Demo Blaze UI using cypress as base framework (4 days)**
         1. Verify demo blaze Home Page
         2. Verify Log in Flow
         3. Verify Sign up Flow
         4. Verify can load products in each product category.
            1. Phones
            2. Mobiles
            3. Laptops
         5. Verify user can view a single Product.
         6. Verify add to cart functionality.
         7. Verify delete products form cart.
         8. Verify user can place order.
2. **Performance Tests**
   1. **Task: Create a Performance test Script for major calls flows and APIs including (2 days)**
      1. Login and Logout
      2. New user Sign up.
      3. Load Products per category
      4. Phones
      5. Laptops
      6. Monitors
      7. View products page
      8. Add product to Cart.
      9. Check Out and Purchase request.
      10. Home Page Load for <https://www.demoblaze.com/index.html>
   2. **Task : Execute Performance Tests (3 days)**
      1. **Load Test:** Demo blaze can handle expected user traffic.
         * Output:
           1. Get the anticipated traffic on Demo blaze website for every day in hits per second from the Product Owner/ Business stakeholders.
           2. Perform the Load test to check if the application can handle the expected traffic.
      2. **Stress Test:** To check for 20% more than anticipated traffic on the demo blaze website.
         * Output:
           1. To find the breakpoint of the application
           2. To check if application under stress can be scaled either horizontally or vertically
      3. **Spike Test:** This is to simulate a sudden spike or peak load on the application for APIs like add to cart and checkout. This will be useful when we have any mega sale event like 10.10 or 11.11 sale when the users will suddenly rush to the website to buy at discounted price
         * Output:
           1. To check that the application can handle a sudden burst in traffic.
           2. To check if we need to scale up the infra and deployment before the sale.
           3. To check that the response time is not impacted.
           4. To check the failure rate is <2% and user experience is not dropped.
      4. **Soak Test:** This to run a performance testing on demo blaze for >24 hours to 48 hours.
         * Output:
           1. To Determine if the application have any memory leaks or dead logs over long run.
           2. To check the response time for critical APIs, remain same over period of longer time.
           3. To check there is no sudden crash or application failures.
3. **Security Tests**
   1. **Task : Document and Execute Security Tests to include: (3 days)**
      1. **Input Validation Testing:** Test input fields (such as forms, search bars, and URL parameters) to ensure they properly sanitize user input. This helps prevent common attacks like SQL injection and cross-site scripting (XSS).
      2. **Authentication Testing:** Verify that the authentication mechanism securely handles user credentials (e.g., passwords) and prevents common vulnerabilities like brute force attacks, session fixation, and account enumeration.
      3. **Authorization Testing:** Test whether users are only able to access resources and perform actions that they are authorized to do. Ensure that privilege escalation and insecure direct object references are prevented.
      4. **Session Management Testing:** Check how session tokens are generated, transmitted, and stored. Test for session fixation, session hijacking, and session timeout vulnerabilities.
      5. **Sensitive Data Exposure Testing:** Ensure that sensitive data such as credit card information, passwords, and personal details are encrypted both in transit (using HTTPS) and at rest (in databases).
      6. **Payment Gateway Security Testing:** Verify the security of payment processing systems and gateways to prevent attacks like credit card skimming, man-in-the-middle (MITM) attacks, and payment fraud.
      7. **Cross-Site Request Forgery (CSRF) Testing:** Test for CSRF vulnerabilities by crafting malicious requests that execute unauthorized actions on behalf of authenticated users. Verify that proper anti-CSRF tokens are implemented.
      8. **Security Headers Testing:** Check if security headers such as Content Security Policy (CSP), Strict Transport Security (HSTS), and X-Content-Type-Options are correctly configured to mitigate various web-based attacks.
      9. **Third-Party Integration Testing:** Assess the security of third-party libraries, plugins, and APIs used by the ecommerce platform. Verify that they are up-to-date, properly configured, and do not introduce security vulnerabilities.
      10. **Security Misconfiguration:** Check if any API keys are exposed over network response for configs.
      11. **Denial of service attacks:** Make sure that APIs have rate limiting implemented.
4. **Usability Tests**
   1. **Task : To execute usability tests on Demo Blaze website (2 days)**
      1. **UI/UX consistency:** Ensure a consistent and intuitive user experience across different devices and browsers.
      2. **Mobile responsiveness:** Test the website's responsiveness on various mobile devices.
5. **Accessibility Tests**
   1. **Task : To execute accessibility tests on Demo Blaze website (2 days)**
      1. **Keyboard Navigation Testing:** Verify that users can navigate through the website using only the keyboard, without relying on a mouse. Test tab order, focus indicators, and keyboard shortcuts.
      2. **Screen Reader Compatibility Testing:** Test the website with popular screen reader software (such as JAWS, NVDA, or VoiceOver) to ensure that users with visual impairments can access and understand the content.
      3. **Color Contrast Testing:** Check the color contrast ratios between text and background colors to ensure readability, particularly for users with low vision or color blindness. Tools like WebAIM's Color Contrast Checker can help with this.
      4. **Semantic HTML Testing:** Ensure that HTML elements are used semantically and appropriately (e.g., using headings, lists, labels) to provide context and structure for screen readers and other assistive technologies.
      5. **Alternative Text (Alt Text) Testing:** Verify that all images and non-text content have descriptive and meaningful alternative text (alt text) to provide information to users who cannot see the images.
      6. **Form Field Labels and Errors Testing:** Check that form fields have associated labels and error messages are descriptive and clearly associated with the corresponding form fields. This helps users understand input requirements and errors.
      7. **Focus Management Testing:** Test focus management to ensure that keyboard users can easily identify and navigate through interactive elements. Ensure that focus styles are visible and consistent.
      8. **Resizing and Zooming Testing:** Verify that the website's layout remains usable and content remains readable when the user zooms in or changes text size. Test responsiveness to different viewport sizes.
      9. **Skip Navigation Link Testing:** Ensure that a "skip to main content" link is available and functional, allowing users to bypass repetitive navigation elements and access the main content directly.

**Note:** Keep in mind that these estimates are approximate and can vary depending on the complexity of the website and the testing environment. Additionally, it's essential to iterate on testing tasks continuously and adjust priorities based on feedback and emerging issues.