Day 6 – Deployment Preparation and Staging Environment Setup - Hekto

Overview:

On Day 6, the focus was on preparing the marketplace for deployment by setting up a staging environment, configuring the hosting platforms, and ensuring the Application's readiness for real-world use. This day was critical to simulate the Production environment and to catch any potential issues before deploying the Application to a live, customer-facing environment. The goal was to test the Application in a production-like setting, which helps to identify any deployment or configuration issues that might arise during the final deployment to Production.

Deployment Strategy Planning:

For the staging deployment, I opted for Vercel as the hosting platform because of its swift and efficient deployment process, which works well for both static and dynamic applications. Vercel's seamless integration with GitHub allowed me to connect my GitHub repository to the platform, facilitating continuous integration and continuous deployment. This setup ensured that changes pushed to the repository would automatically trigger deployments. I also configured the build settings and deployment scripts through Vercel's dashboard, ensuring that the build process, including bundling, environment setup, and resource compilation, was optimized for the staging environment. This approach provided a smooth deployment process and enabled thorough testing of the application in a production-like setting.

Environment Variables Configuration:

To securely store all sensitive configuration data, I created a .env file locally. This file included API keys for Stripe, Clerk, and Ship24, as well as Sanity CMS credentials and other environment-specific variables such as database connections. The .env file ensures that all sensitive information remains secure and is not exposed in the codebase.

After setting up the file, I securely uploaded these environment variables to the hosting platform's dashboard. This step ensures that the application can access the necessary credentials and settings in the staging environment while keeping sensitive data protected. By adopting this approach, I made sure the application functions correctly in the staging environment with the appropriate production or staging credentials.
This method not only protects sensitive information but also streamlines the deployment process by ensuring that all necessary environment variables are correctly configured and easily accessible.

Staging Environment Setup:

I successfully deployed the application to a staging environment, ensuring that the build process was completed without any errors. Post-deployment, I verified that the website loaded correctly and rigorously tested key functionalities such as product listings, product details pages, search, checkout, and cart operations. Everything performed seamlessly, indicating the application's readiness for more exhaustive testing. Furthermore, I inspected additional essential features to guarantee a consistent and error-free user experience. The successful deployment and flawless performance confirmed that the staging environment was well-prepared for further testing and validation.

Staging Environment Testing:

❖ Functional Testing:

I conducted thorough testing on the application's core functionalities, which included browsing through products, adding items to the cart, and finalizing the checkout process. Given that product data is sourced from Sanity CMS, I ensured that data retrieval and display on the site were accurate and seamless. Utilizing tools like Postman, I validated the APIs to confirm that data was being fetched and processed correctly from Sanity. The application performed flawlessly, accurately responding to all user interactions.

❖ Performance Testing:

I utilized the Lighthouse tool for performance testing to evaluate the website's efficiency. This tool provided crucial insights into performance, accessibility, best practices, and SEO. It helped me assess various metrics such as page load time, speed, and overall site responsiveness. Lighthouse also identified areas where improvements could be made to enhance the website's speed and navigation. This comprehensive testing ensured the site performs well under different conditions and provided a smooth, fast, and user-friendly experience.

❖ Security Testing:

I carried out comprehensive security testing to identify and mitigate potential vulnerabilities that could leave the site exposed to attacks. Ensuring the proper implementation of HTTPS was a priority to safeguard data during transmission. Additionally, I confirmed that sensitive information, such as API keys, was securely managed and not exposed. I also verified that the necessary secure HTTP headers were in place to provide an additional layer of protection against potential threats. This thorough approach ensured that the site was well-protected against common security issues.

❖ Test Reporting:

I meticulously recorded all test results and identified issues throughout the testing process. Each test case report detailed the Test Case ID, steps taken, expected outcomes, actual outcomes, and the status (pass or fail). Additionally, I noted any unresolved issues that need to be addressed later. This comprehensive documentation helped in tracking everything that was tested and pinpointing areas that required further attention.

Test Case ID	Description	Test Steps	Expected Result	Actual Result	Status	Security Assigned Level	To	Remarks
TC01	Validate product listing page	Verify displayed products correctly	Product is displayed correctly	Product displayed correctly	Passed	Medium	-	No issue found
TC02	Verify responsiveness	View on my mobile	Fully responsive	Fully responsive	Passed	Medium	-	No issue found
TC03	Check cart functionality	Verify product listing on the Home Page, Navigate to Products listed	Cart updates with added components	Cart updates as expected	Passed	High	-	Work as expected
TC04	Verify product listing on the Home Page	Enter a search term	Accurate search results	Accurate search results	Passed	Low	-	No issue found
TC05	Validate search functionality	Verify cart operations (adding, updating, removing items)	All operations work as expected	All operations work as expected	Passed	Medium	-	No issue found
TC06	Add an item to the cart	Click on a product from the listing	Product added to the cart	Product added to the cart	Passed	High	-	No issue found
TC07	Verify dynamic routing to product details page	Verify error handling in case of API failures	Product details page loads correctly	Product details page loads correctly	Passed	Low	-	No issue found
TC08	Optimize performance using Lighthouse	Simulate an API error	Performance should appear	Performance should appear	Passed	Medium	-	No issue found
TC09	Verify the filter functionality on the Products Page	Open the Home Page rendering be high	Consistent	Consistent	Passed	Low	-	No issue found
TC10	Verify cross-browser compatibility	Verify the filter category functionality on the Products Page (e.g., "Chairs")	All chairs show on the page	All chairs show on the page	Passed	Medium	-	No issue found
TC11	Verify user authentication functionality (signin and signout)	Test login with valid/invalid credentials and logout with valid credentials	User logs in and out successfully with valid credentials	User logs in and out successfully with valid credentials	Passed	High	-	No issue found

Final Checklist:

Deployment Preparation	Staging Environment Setting	Documentation	Form Submission	Final Review
✓	✓	✓	✓	✓

PREPARED BY: NADIA SHAIKH