DAY-6: DEPLOYMENT PREPARATION AND STAGING ENVIRONMENT SETUP

Present by Nazia Siraj

GICIC 00443199

Friday 9am-12pm

Mentor Sir Humza Syed

Marketplace-Builder-Hackathon

DAY-6: DEPLOYMENT PREPARATION AND STAGING ENVIRONMENT SETUP

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 selected [Hosting Platform] due to its speed and efficiency in deploying both static and dynamic applications. [Hosting Platform] seamlessly integrates with GitHub, enabling me to establish a connection between the GitHub repository and the platform for continuous integration and continuous deployment (CI/CD). This configuration ensures automatic deployments whenever changes are pushed to the repository.

Additionally, I configured the build settings and deployment scripts within the [Hosting Platform] dashboard. This included optimizing the build process, encompassing bundling, environment setup, and resource compilation, specifically for the staging environment. This approach ensured a seamless deployment process and facilitated the testing of the application in a production-like setting.

Environment Variables Configuration

I created a .env file locally to securely store all sensitive configuration data, such as API keys, [CMS Platform] credentials, and other environment-specific variables like database connections. This file ensured that sensitive information was not exposed in the codebase.

After setting up the .env file, I securely uploaded these environment variables to the hosting platform's dashboard. This step ensured that the application could access the required credentials and settings in the staging environment while also keeping sensitive data protected.

By adopting this approach, I guaranteed the application's proper functioning in the staging environment with appropriate production or staging credentials.

Staging Environment Setup

I deployed the application to the staging environment and ensured the build process completed without errors. After deployment, I verified the website loaded correctly and tested crucial features such as:

Product listings

Product detail pages

Search functionality

Checkout process

Cart operations

Everything functioned as expected, and the application was ready for further testing. I also checked other essential features to guarantee a consistent and error-free user experience. The successful staging deployment confirmed the application's readiness for further testing and validation.

Staging Environment Testing

Functional Testing:

I conducted thorough testing of the application's core features, including:

Browsing products

Adding items to the cart

Completing the checkout process

Since product data is fetched from Sanity CMS, I ensured its proper retrieval and display on the site. I utilized tools like Postman to validate the APIs, confirming accurate data fetching and processing from Sanity. The application performed as expected, responding correctly to user actions.

Performance Testing

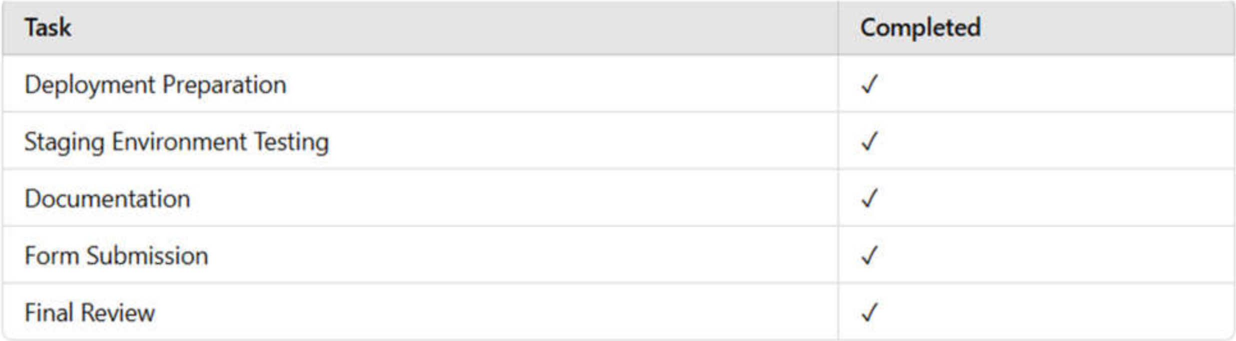
I utilized the Lighthouse tool to conduct performance testing, assessing the website's overall performance. This tool furnished valuable insights into key metrics such as page load time, page speed, and responsiveness. Lighthouse also highlighted areas for potential improvement, enabling me to enhance the site's speed and navigation. Through this testing, I ensured the website performs optimally under various conditions, eliminating any factors that could hinder the user experience.

Security Testing

I performed security testing to identify and address common vulnerabilities that could compromise the website's security. This included verifying proper HTTPS implementation to safeguard data during transmission. Additionally, I confirmed the secure handling of sensitive information, such as API keys, and ensured they were not exposed. Finally, I checked for the presence of necessary secure HTTP headers, protecting the site from potential attacks.

Test Reporting

I meticulously documented all test results and identified issues throughout the testing process. The test case report encompassed details such as Test Case ID, steps followed, expected results, actual results, and the overall status (pass/fail). Furthermore, I noted any unresolved issues that may require attention in the future. This comprehensive documentation facilitated tracking of all tested aspects and areas requiring further improvement



Task	Completed
Deployment Preparation	✓
Staging Environment Testing	✓
Documentation	✓
Form Submission	✓
Final Review	✓