Day 6 - Deployment Preparation and Staging Environment

On Day 6, I focused on getting my marketplace ready for deployment by setting up a staging environment. This step ensures the application runs smoothly in a production-like setup before going live. Building on the testing and optimization from the previous day, I made sure everything was in place for a seamless experience.

1. Setting Up the Hosting Platform

For hosting, I decided to use **Vercel** as my deployment platform.

Steps I Followed:

- First, I created an account on **Vercel** and logged in.
- Then, I linked my **GitHub repository** using Vercel's built-in integration feature.
- After that, I adjusted the build and deployment settings according to my project's needs.
- Finally, I verified that the staging environment was deployed successfully.

2. Configuring Environment Variables for Deployment

To securely store sensitive information like API keys and database credentials, I created an **.env** file.

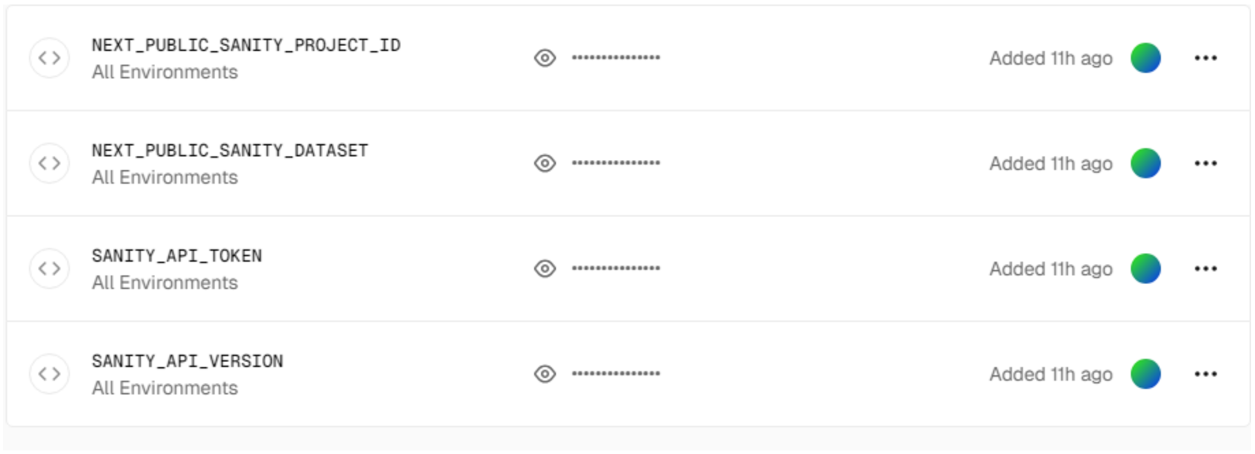
Steps I Followed:

- I added a `.env` file in the root directory of my project.
- Inside the file, I stored key variables such as:

  ```
  makefile
  CopyEdit
  NEXT_PUBLIC_SANITY_PROJECT_ID=my_project_id
  NEXT_PUBLIC_SANITY_DATASET=production
  SANITY_API_TOKEN=my_api_key
  ```

- To keep the credentials secure, I uploaded them directly to **Vercel's environment variable settings** instead of exposing them in the code.

This setup ensures that the application remains secure while running in a controlled staging environment before the final deployment.



```
NEXT_PUBLIC_SANITY_PROJECT_ID    ••••••••••••••••    Added 11h ago
All Environments
NEXT_PUBLIC_SANITY_DATASET       ••••••••••••••••    Added 11h ago
All Environments
SANITY_API_TOKEN                 ••••••••••••••••    Added 11h ago
All Environments
SANITY_API_VERSION               ••••••••••••••••    Added 11h ago
All Environments
```

3. Deploying the Application to the Staging Environment

Before deploying, I first ran the following command in the terminal to check for any errors:

```
npm run build
```

Once I was satisfied that there were no issues, I proceeded with the deployment by clicking the **Deploy** button on **Vercel**.

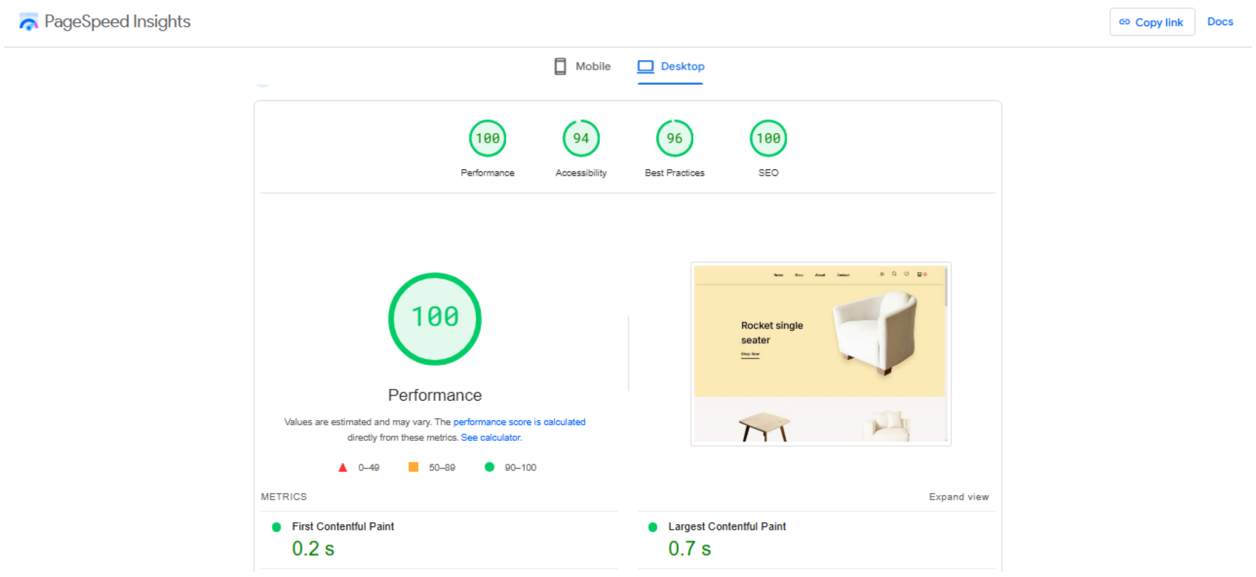
4. Testing the Staging Environment

To ensure everything was working as expected, I performed thorough testing.

Testing Methods Used:

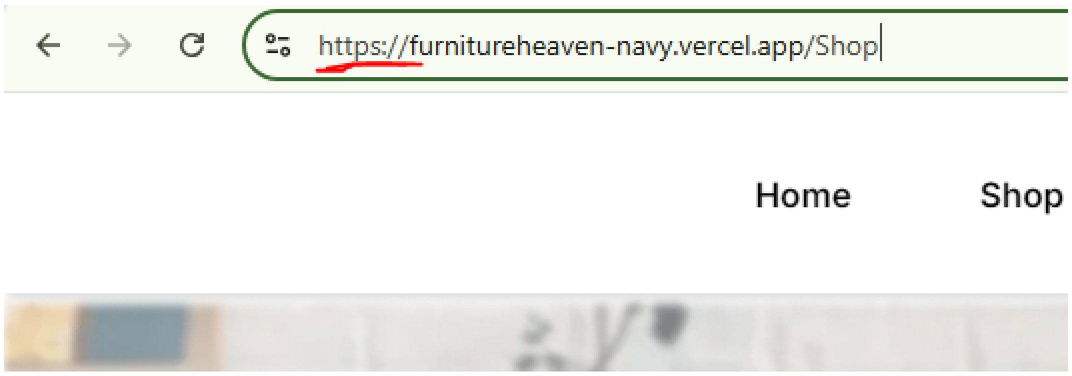
- **Functional Testing**: Manually tested key features like product listings, cart functionality, and navigation.
- **Performance Testing**: Used **Lighthouse** to analyze the app's speed, responsiveness, and overall performance.

This process helped me verify that the application was ready for the next stage before moving toward full deployment.



Values are estimated and may vary. The performance score is calculated directly from these metrics. See calculator.

1. **Security Testing**: I validated input fields to ensure there were no vulnerabilities like, confirmed **HTTPS** was enabled, and checked if sensitive data was properly encrypted.


https://furnitureheaven-navy.vercel.app/Shop

5. Results from Staging Testing

During testing, I analyzed the application's functionality, performance, and security to ensure a smooth user experience.

Test Results:

- **Functional Testing**: All major features, including the product page and cart, worked as expected without any errors.
- **Performance Testing**: The website passed all performance tests, loading within an acceptable timeframe as confirmed by **Lighthouse**.
- **Security Testing**: All input fields were secure, **HTTPS** was active, and no security vulnerabilities were detected.

6. Documentation Created

To keep track of the deployment and testing process, I created the following documentation:

- **Test Case Report**: Documented test cases, including descriptions, steps, expected results, and actual outcomes.
- **Performance Report**: Added **Lighthouse** performance test results for future reference.
- **Deployment Documentation**: Updated the **README.md** file with details on the deployment process, testing results, and instructions for running the app.
- **File Organization**: Maintained a well-structured **documents/** folder containing all test reports and performance data.

7. Project Structure and Professional Setup

To ensure a clean and professional setup, I followed a structured approach:

- **src/** → Contains all source code files.
- **public/** → Stores static files like images and assets.
- **documents/** → Holds project documentation, including test reports and performance data.

This structured organization makes the project easy to manage, scale, and maintain.

README.md

The **README.md** file includes a summary of the project, along with detailed instructions for setting up, deploying, and testing the application.

Conclusion

By setting up a staging environment and ensuring all configurations were in place, I laid a solid foundation for the customer-facing application. The testing confirmed the app's functionality, performance, and security were up to production standards.

The deployment was smooth with Vercel, and environment variables were managed securely for seamless integration. Performance insights from Lighthouse, combined with manual and security testing, helped ensure reliability.

Comprehensive documentation was created to support future maintenance and collaboration.

This setup ensures the marketplace is ready to move into production, providing a smooth experience for both users and stakeholders.