Prepared by Muhammad Nabeel

Day 6 - Deployment Preparation and Staging Environment Setup

Published: January 21, 2025

Name of the Marketplace : www.clothsite.com (not confirmed)

Objective

The focus of Day 6 is to prepare the application for deployment by setting up a staging environment, ensuring that the system is production-ready, and managing environments professionally. By the end of this phase, the project will be deployed to a staging platform, undergo testing, and be structured for efficient collaboration and deployment.

This phase also introduces critical stages in application lifecycle management, such as the roles of **TRN**, **DEV**, **SIT**, **UAT**, **PROD**, and **DR** environments. These stages are essential for structured deployment and ensuring the application's robustness in both non-production and production setups.

Key Learning Outcomes

Environment Setup and Deployment:

- Configure a staging environment on a hosting platform.
- Deploy the application to a staging environment and validate its functionality in a production-like setup.
- Securely manage environment variables and configurations.

Professional Environment Management:

- Implement the following key environments in deployment pipelines:

- **TRN (Training)**: This environment is dedicated to training and hands-on practice for developers, testers, or end-users to get acquainted with the platform's functionalities. It mirrors the actual production environment to some extent but is not meant for real-world use.
- **DEV (Development)**: The **DEV** environment is where new features are actively developed and tested in isolation. This environment is frequently updated with changes and ensures that the core application remains stable while new features are implemented.
- **SIT (System Integration Testing)**: This environment is used to validate the integration of various system modules, ensuring they work together without errors. Any issues found in SIT must be addressed before moving to UAT.
- **UAT (User Acceptance Testing)**: Here, end-users test the application to ensure it meets their expectations. User feedback from this environment often leads to final adjustments before production deployment.
- **PROD (Production)**: The **PROD** environment is the live environment where real users interact with the application. It must be stable, secure, and performant to meet end-user needs.
- **DR (Disaster Recovery)**: A backup environment that mirrors the **PROD** setup, but it is only used in emergencies when the production environment fails.

Testing in the Staging Environment:

- Perform **functional**, **performance**, and **security testing** to ensure that the application is stable and secure before going live.
- Document all issues and resolutions during testing to ensure smooth deployment and future reference.

Documentation and Repository Management:

- Create a **deployment document** that includes detailed test case reports and performance metrics to ensure proper testing and validation.
- Organize the project repository on GitHub for easy navigation, ensuring that collaborators can efficiently work with the project's files.

Steps and Implementation

Step 1: Hosting Platform Setup

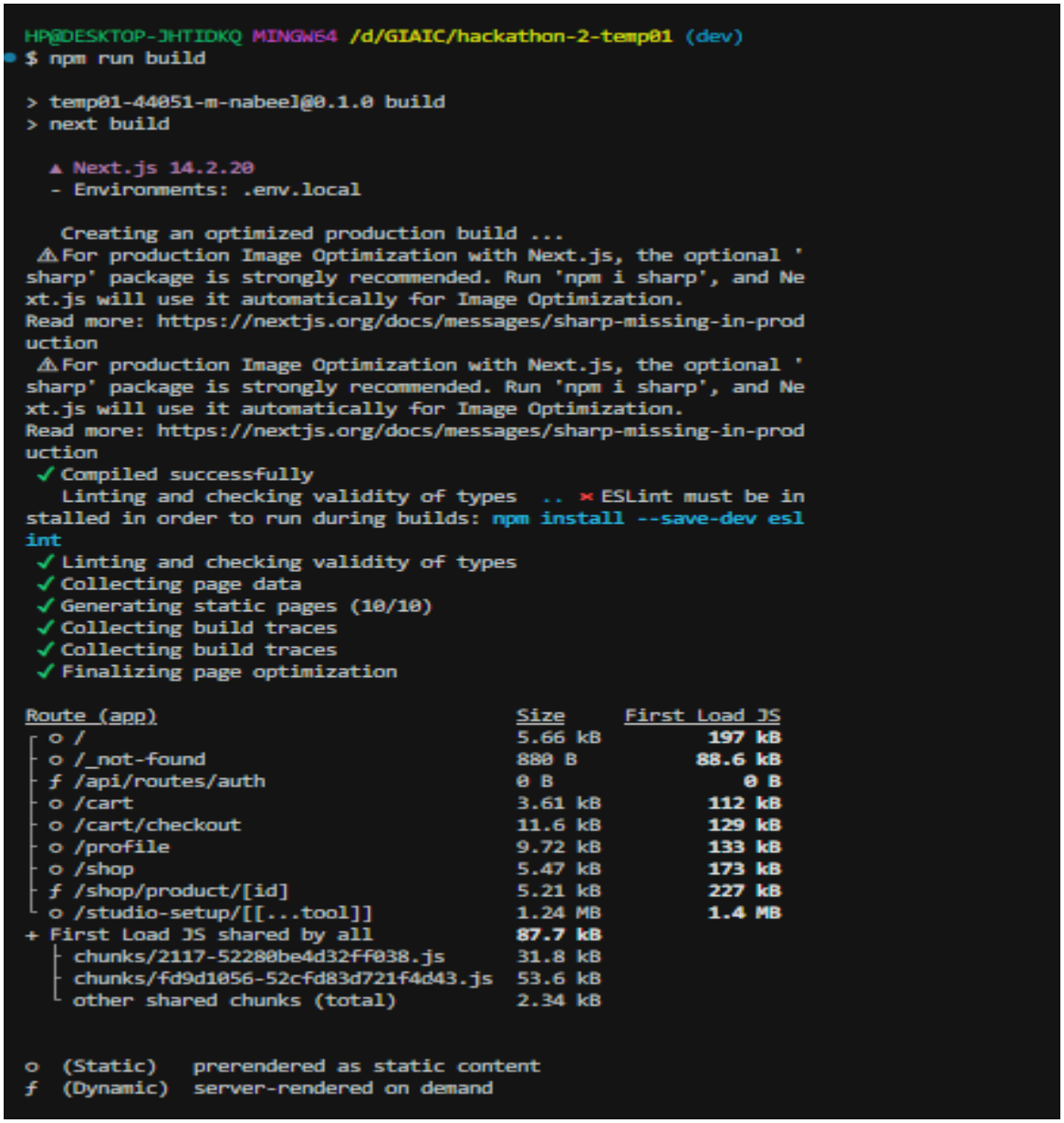
Choosing a Platform

- I selected **Vercel** for hosting because of its seamless integration with Next.js, automated deployment capabilities, and easy-to-use dashboard.

- Other platforms like Netlify, AWS, and Azure were considered, but Vercel was best suited for this project due to its performance and simplicity.

Connecting the Repository

1. Logged into Vercel and linked the GitHub repository.
2. Configured the build settings with the following values:
    - **Build Command**: `npm run build`
    - **Output Directory**: `.next`

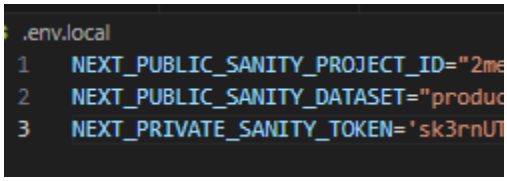


3. Verified deployment scripts in **`package.json`** for smooth builds.

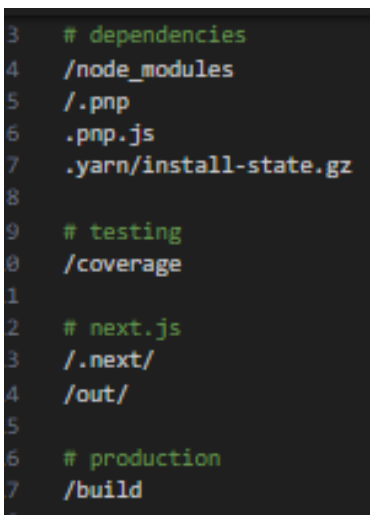
Step 2: Configuring Environment Variables

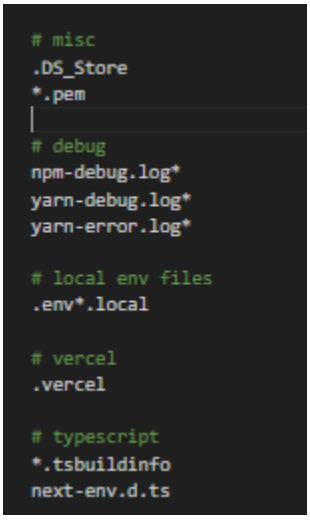
Creating a `.env` File

Created a local **`.env`** file to securely store sensitive information, such as:
plaintext



- Added the **`.env`** file and other to **`.gitignore`** to prevent accidental commits.

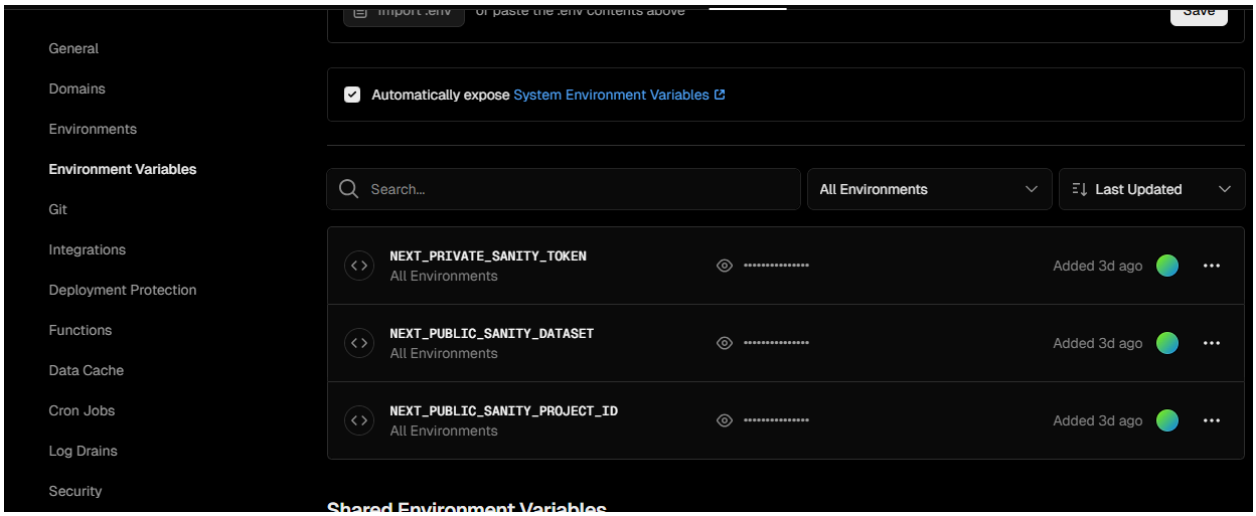




Uploading Variables to Vercel

1. Navigated to the **Environment Settings** section in the Vercel dashboard.

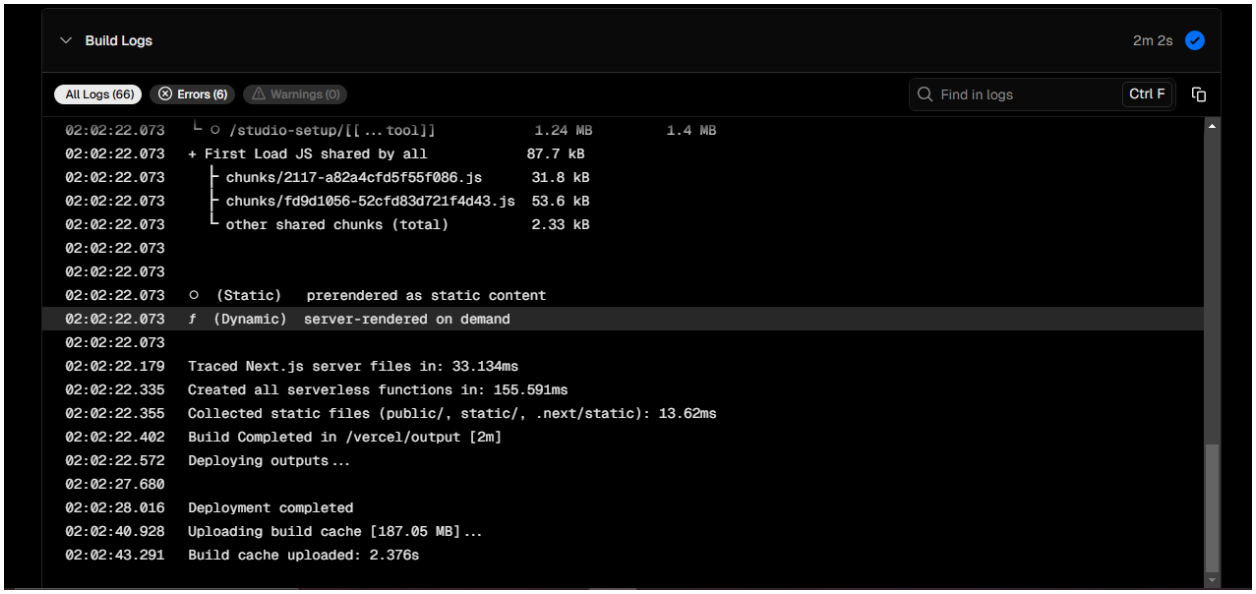
2. Added all required environment variables, ensuring names matched those in the code.

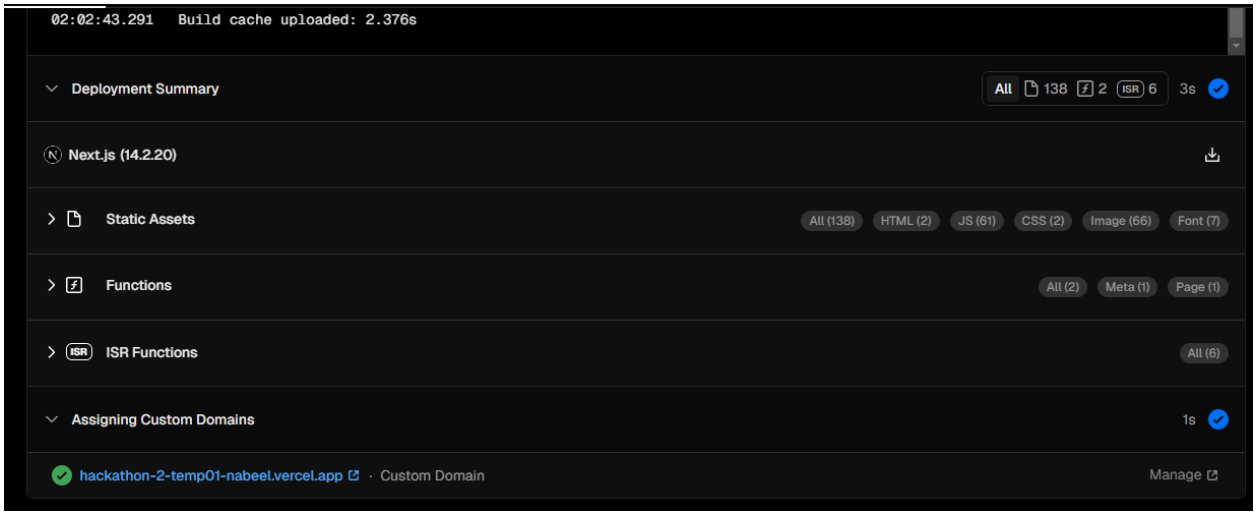


Step 3: Deploying to Staging

Deployment Process

- Triggered the first deployment to Vercel's staging environment.
- Monitored the build process, which completed successfully without errors.





Validating Deployment

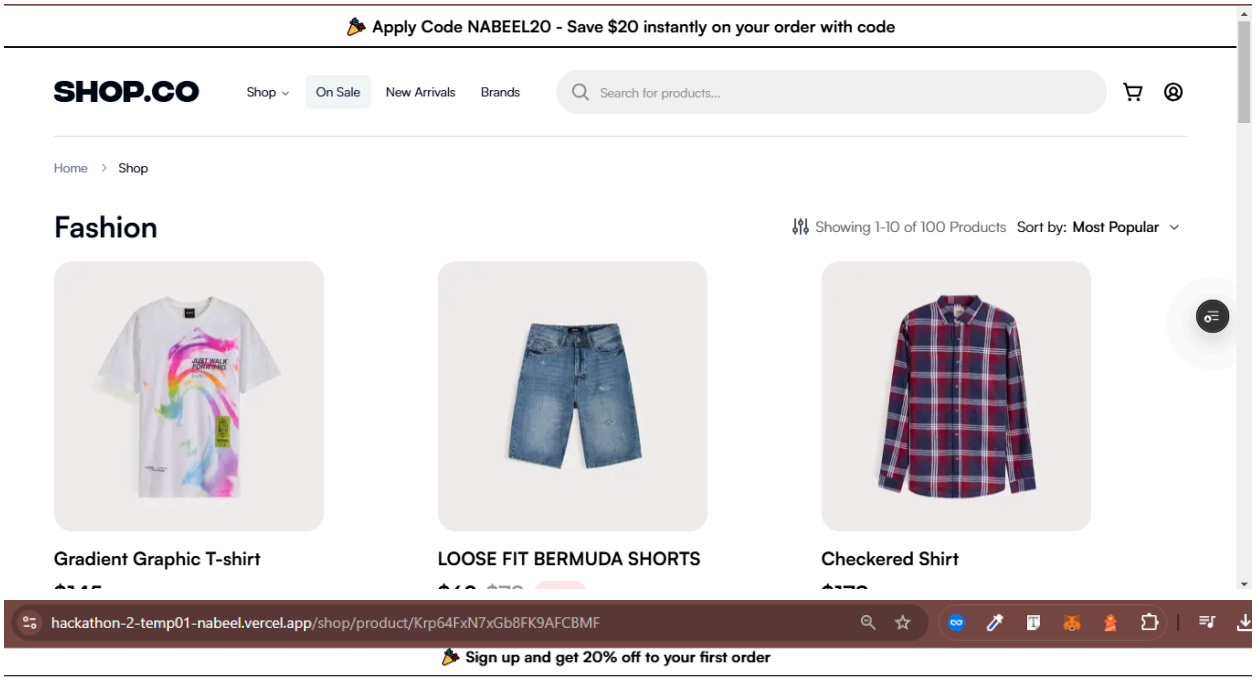
1. Accessed the staging URL provided by Vercel.
2. Performed initial validation:
    - Tested navigation across all pages.
    - Verified API responses and ensured data was displayed correctly.
    - Confirmed proper layout rendering on multiple devices.

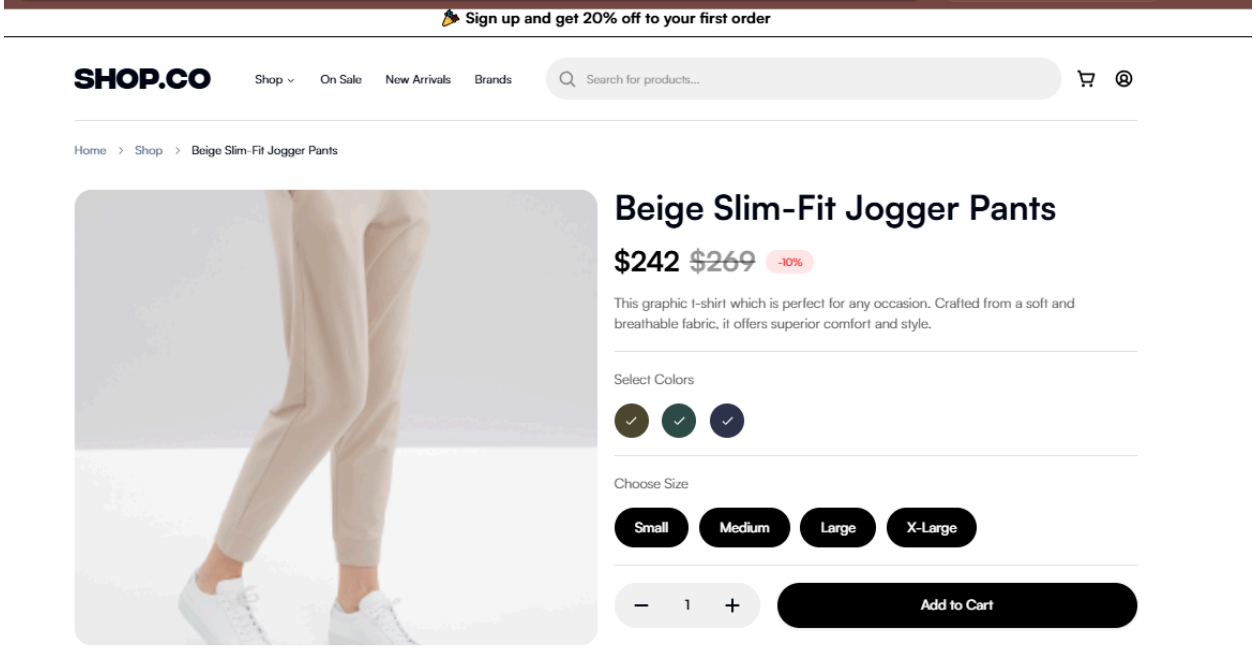
Step 4: Staging Environment Testing

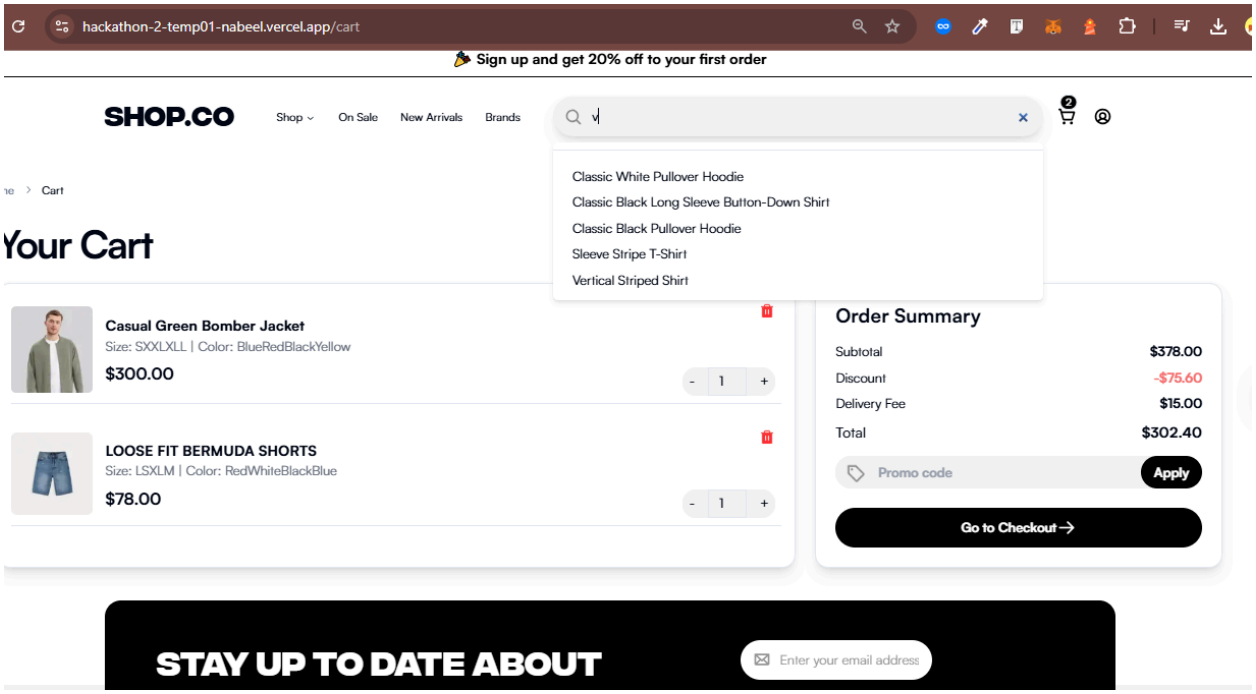
Testing Types

- **Functional Testing**: Verified core functionalities such as:

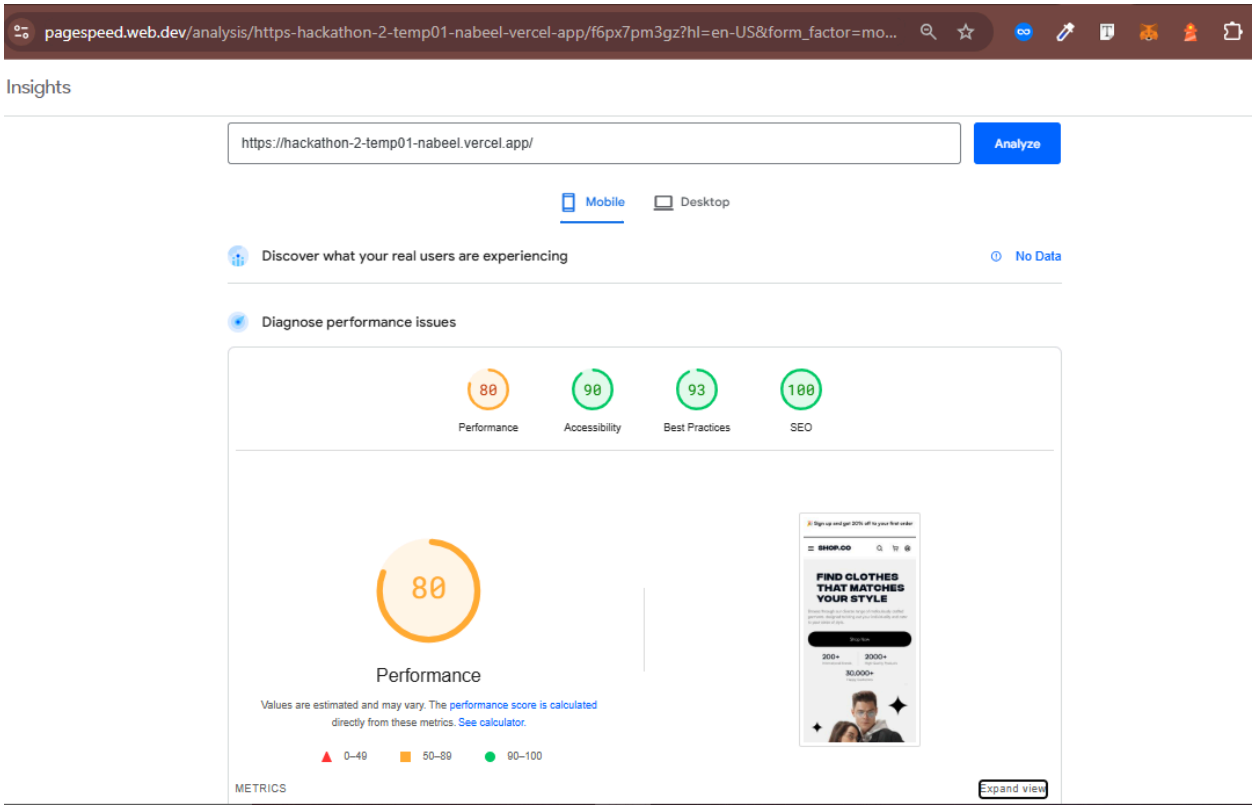
- Product listing.
- Search feature.
- Add-to-cart and checkout workflows.







- **Performance Testing**:
  - Conducted a Lighthouse audit to evaluate key metrics:



pagespeed.web.dev/analysis/https-hackathon-2-temp01-nabeel-vercel-app/f6px7pm3gz?hl=en-US&form_factor=des...

ghts

https://hackathon-2-temp01-nabeel.vercel.app/

Analyze

Mobile | Desktop

Discover what your real users are experiencing — No Data

Diagnose performance issues

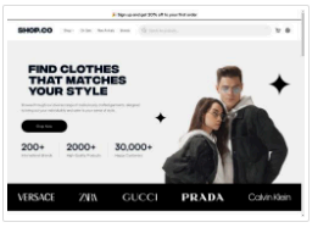
Performance	Accessibility	Best Practices	SEO
94	91	96	100

94

Performance

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

▲ 0–49 ■ 50–89 ● 90–100



https://hackathon-2-temp01-nabeel.vercel.app/

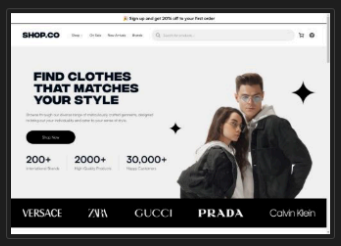
Performance	Accessibility	Best Practices	SEO
98	91	96	100

98

Performance

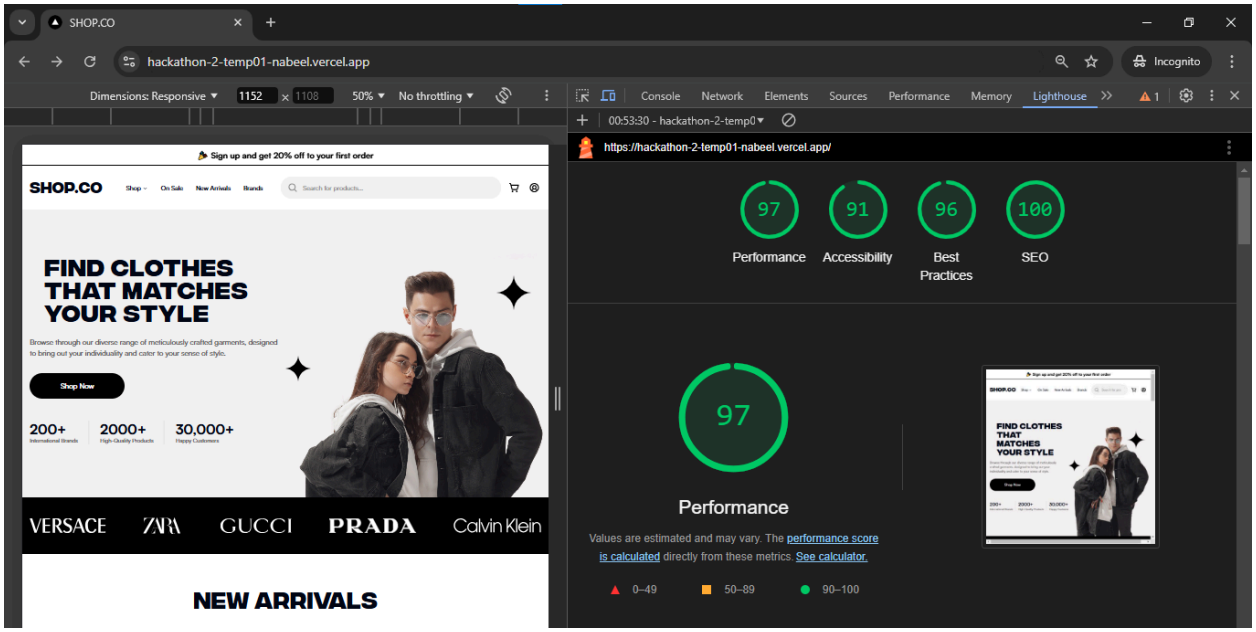
Values are estimated and may vary. The performance score is calculated directly from these metrics. See calculator.

▲ 0–49 ■ 50–89 ● 90–100



METRICS — Expand view

First Contentful Paint	Largest Contentful Paint
0.2 s	1.1 s
Total Blocking Time	Cumulative Layout Shift

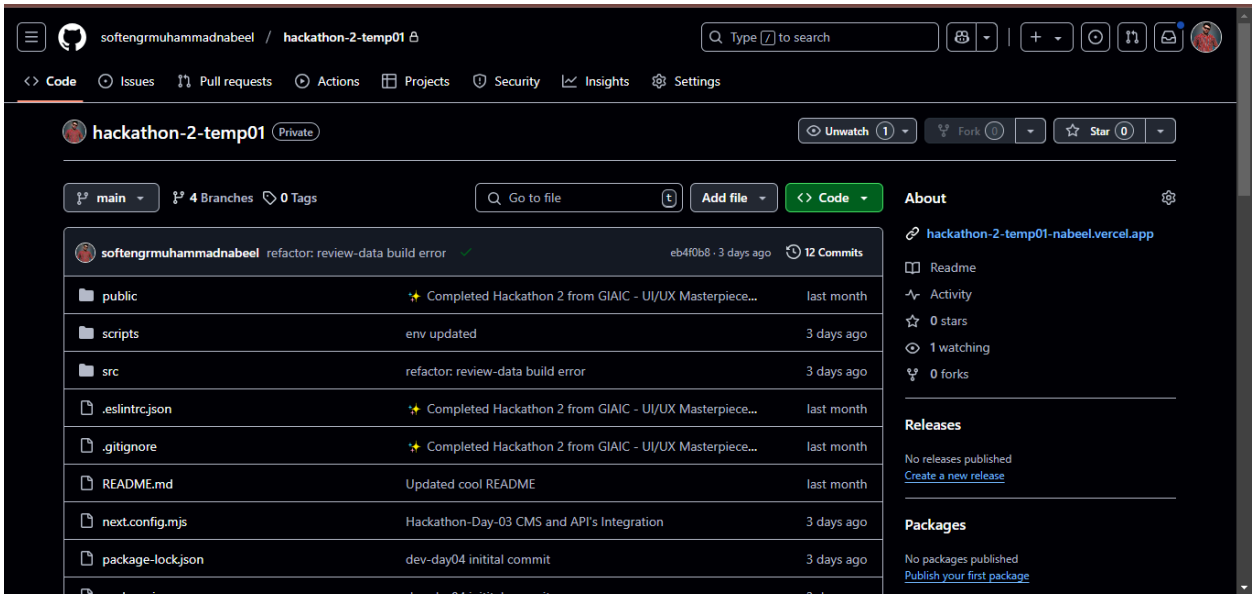


- **Security Testing**:
  - Ensured secure API communications (HTTPS).
  - Validated input sanitization for forms and search fields to prevent malicious injections.

Test Case Report : [CLICK HERE]

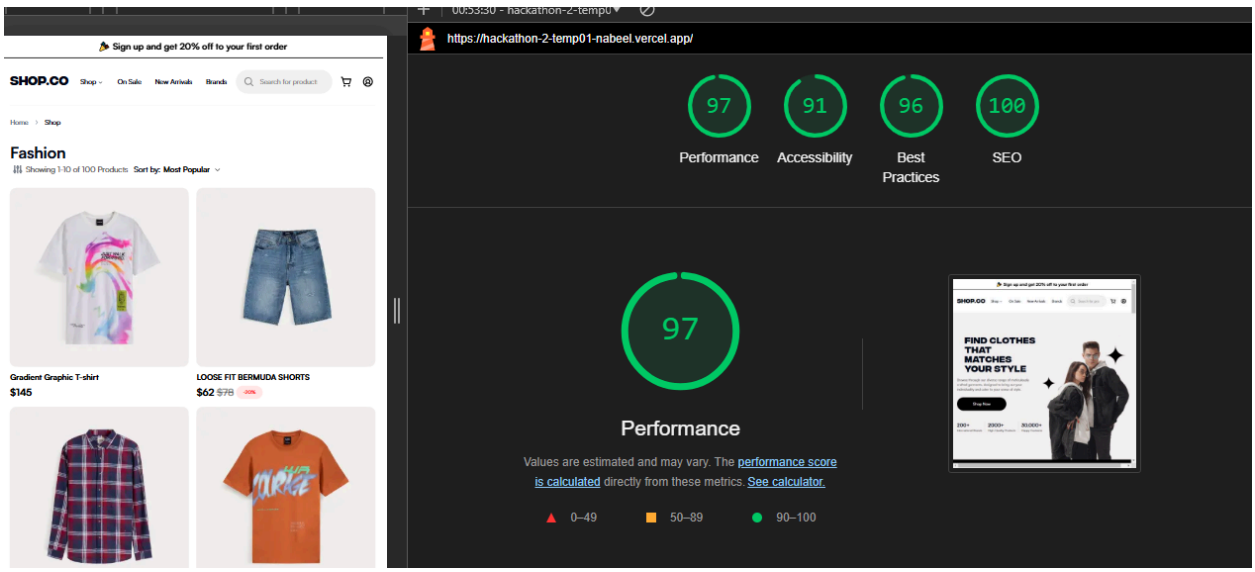
Step 5: Repository Organization

1. Structured the GitHub repository into clearly defined folders:
   - Contains application source code.
   - Stores static assets like images and fonts.
   - Includes test case reports, deployment documentation, and performance metrics.
   - Configuration files for CI/CD workflows.
2. Added a README file with:
   - Project overview.
   - Deployment instructions.
   - Folder structure description.



Performance Metrics

- Lighthouse Report:
  - **Performance**: 97.
  - **Accessibility**: 91.
  - **Best Practices**: 96.
  - **SEO**: 100.
- **API Response Time**: ~180ms.
- **Deployment Build Time**: ~2 minutes.



Conclusion

Day 6 of the project marked a significant milestone in preparing the application for deployment. The key achievements were the successful setup of a staging environment and the completion of initial testing phases to ensure the application is production-ready. Through the careful selection of Vercel as the hosting platform, environment variables were securely managed, and deployment scripts were verified for smooth operation.

By deploying the application to a staging environment, we were able to perform comprehensive testing, covering functional, performance, and security aspects. The staging environment provided a close replica of the production environment, which was crucial in ensuring that the application behaves as expected before the final launch. Functional testing confirmed that core features like product listing, search, and checkout workflows were working seamlessly. Performance testing, through tools like Lighthouse, highlighted a solid performance score, ensuring that the application is optimized for end-users. Security tests also validated the integrity of the application, ensuring it is protected from potential vulnerabilities.

Additionally, the repository structure was organized to facilitate efficient collaboration and deployment. The inclusion of test case reports and performance metrics ensures a transparent process, with detailed documentation available for future reference.

By completing these steps, the project is now fully prepared for the next phase of deployment, with a stable, secure, and optimized application ready for user acceptance testing (UAT). The meticulous planning of key environments, along with the testing conducted in the staging environment, ensures that the transition to production will be smooth and successful. This stage also highlighted the importance of managing environments professionally and the critical role each environment plays in maintaining the overall health and stability of the application throughout its lifecycle.

Prepared by Muhammad Nabeel