Deployment Preparation and Staging Environment Setup

Prepared by: Muhammad Umar

Date: January 21, 2025

Version: 6.0

Objective

Day 6 focuses on preparing your marketplace for deployment by:

- Setting up a staging environment.
- Configuring hosting platforms.
- Ensuring readiness for a customer-facing application.

Building on the testing and optimization work from Day 5, this stage emphasizes seamless operation in a production-like environment. It also introduces industry-standard practices for managing different environments (e.g., TRN, DEV, SIT, UAT, PROD, DR).

Key Learning Outcomes

1. Set up and configure a staging environment:

- Select a hosting platform (e.g., Vercel or Netlify).
- Connect your GitHub repository.
- Configure build and deployment settings.
- Securely set up environment variables.
- Validate application functionality in a production-like environment.

2. Understand professional environment management (TRN, DEV, SIT, UAT, PROD).

3. Conduct staging environment testing and document results.

4. Create professional deployment documentation, including test case and performance reports.

5. Organize project files and documents in a structured GitHub repository with clear folder hierarchies.

Professional Environment Types

- **TRN (Training):** Onboarding and practice.
- **DEV (Development):** Local code writing and testing.
- **SIT (System Integration Testing):** Validating system integrations.
- **UAT (User Acceptance Testing):** Stakeholder testing and validation.
- **PROD (Production):** Live environment for users.
- **DR (Disaster Recovery):** Backup environment for emergencies.

Key Areas of Focus

1. **Deployment Strategy Planning**

   - Choose a hosting platform (e.g., Vercel, Netlify, AWS, Azure).
   - Finalize backend interactions with services like Sanity CMS and third-party APIs.

2. **Environment Variable Configuration**

   - Secure API keys, database credentials, and sensitive data using `.env` files.
   - Configure environment variables in the hosting platform for secure deployment.

3. **Staging Environment Setup**

   - Deploy the application to a staging environment.
   - Validate deployment builds and functionality.

4. **Staging Environment Testing**

   - Conduct functional, performance, and security testing.
   - Document results, focusing on responsiveness and error handling.

5. **Documentation Updates**

   - Create a `README.md` summarizing project activities.

- Organize project files and documentation.

Steps for Implementation

Step 1: Hosting Platform Setup

1. Choose a platform:
   - Vercel or Netlify for quick deployment.
   - AWS or Azure for advanced configurations.
2. Connect your repository:
   - Link your GitHub repository to the hosting platform.
   - Configure build settings and deployment scripts.

Step 2: Configure Environment Variables

1. Create a `.env` file containing sensitive variables:
   ```
   NEXT_PUBLIC_SANITY_PROJECT_ID=your_project_id
   NEXT_PUBLIC_SANITY_DATASET=production
   API_KEY=your_api_key
   ```
2. Upload variables to the hosting platform's dashboard securely.

Step 3: Deploy to Staging

1. Deploy the application through the hosting platform.

2. Validate the deployment to ensure error-free builds and basic functionality.

Step 4: Staging Environment Testing

1. Conduct functional testing using tools like Cypress and Postman.

2. Perform performance testing with Lighthouse or GTmetrix.

3. Conduct security testing by validating inputs and ensuring secure API communications.

4. Document test results in a CSV file with fields like Test Case ID, Description, Steps, Expected Result, Actual Result, Status, and Remarks.

Step 5: Documentation Updates

1. Create a `README.md` summarizing all project activities.

2. Organize project files into a structured hierarchy in the GitHub repository.

Expected Output

1. Fully deployed staging environment for the marketplace.

2. Securely configured environment variables.

3. Comprehensive test case and performance reports.

4. Organized project files and documentation in GitHub.

5. A professional `README.md` file summarizing project activities.

Submission Requirements

1. Submit the deployed staging environment link.
2. Upload a GitHub repository containing:
   - Documents folder (Day 1 to Day 6).
   - Test case report in CSV format.
   - Performance testing results.
   - Organized project files.
   - A `README.md` summarizing activities.
3. Deadline: January 22, 2025, 11:59 AM (Afternoon).

Checklist for Day 6

- Deployment Preparation: [X] / [ ]
- Staging Environment Testing: [X] / [ ]
- Documentation: [X] / [ ]
- Form Submission: [X] / [ ]
- Final Review: [X] / [ ]

FAQs

1. **Why is a staging environment necessary?**

- To test the application in a production-like setting without affecting live users.

2. **What should my test report include?**

- Details of all test cases (passed and failed), including Test Case ID, Description, Steps, Expected Result, Actual Result, Status, and Remarks.

3. **How do I document performance testing?**

- Use tools like Lighthouse or GTmetrix to generate reports and include them in the GitHub repository.

4. **What if I find major issues during staging tests?**

- Document the issues for now. Resolving them can be a post-hackathon activity.

5. **What is the purpose of the `README.md` file?**

- To provide an overview of your project, deployment steps, and results for easy understanding by others (e.g., clients).