DAY 6 - DEPLOYMENT PREPARATION AND STAGING ENVIRONMENT SETUP

Objective:
Day 6 focuses on preparing the marketplace for deployment by setting up a staging environment, configuring hosting platforms, and ensuring readiness for a customer-facing application. This stage ensures the marketplace operates seamlessly in a production-like environment.

Key Learning Outcomes:

- Setting up and configuring a staging environment.
- Understanding professional environment management (TRN, DEV, SIT, UAT, PROD, DR).
- Conducting staging environment testing and documenting results.
- Creating professional deployment documentation.
- Organizing project files in a structured GitHub repository.

Professional Environment Types:

1. **TRN (Training):** Used for onboarding and practice.
2. **DEV (Development):** Local environment for writing and testing code.
3. **SIT (System Integration Testing):** Validates integrations between systems.
4. **UAT (User Acceptance Testing):** Stakeholders test functionality.
5. **PROD (Production):** Live customer-facing environment.
6. **DR (Disaster Recovery):** Backup environment for emergencies.

Key Areas of Focus:

1. **Deployment Strategy Planning:** Choosing a hosting platform (Vercel, Netlify, AWS, Azure).

2. **Environment Variable Configuration:** Securely storing API keys, database credentials.
3. **Staging Environment Setup:** Deploying the application in a production-like setting.
4. **Staging Environment Testing:** Functional, performance, and security testing.
5. **Documentation Updates:** Summarizing all activities in a README.md file.

Steps for Implementation:

Step 1: Hosting Platform Setup

- Choose a platform like Vercel or Netlify.
- Connect the GitHub repository to the platform.

Step 2: Configure Environment Variables

- Create a `.env` file with sensitive information.
- Securely upload variables to the hosting platform.

Step 3: Deploy to Staging

- Deploy the application.
- Validate the build process and basic functionality.

Step 4: Staging Environment Testing

- Functional Testing: Verify product listing, search, cart operations.
- Performance Testing: Use Lighthouse or GTmetrix.
- Security Testing: Validate HTTPS, secure API communications.
- Document all test results in CSV format.

Step 5: Documentation Updates

- Create a `README.md` file summarizing activities.
- Organize project files in a structured folder hierarchy.

Expected Output:

1. Fully deployed staging environment.

2. Secure environment variable configuration.
3. Documented test cases and performance reports.
4. Organized project files in GitHub.
5. A professional `README.md` file.

Submission Requirements:

- Staging environment deployed link.
- GitHub repository with:
  - Documents folder (Days 1-6 documents).
  - Test case report (CSV format).
  - Performance testing results.
  - Organized project files.
  - README.md file summarizing activities.

Checklist for Day 6:

- ☑ ~~Deployment Preparation~~
- ☑ ~~Staging Environment Testing~~
- ☑ ~~Documentation~~
- ☑ ~~Form Submission~~
- ☑ ~~Final Review~~

FAQs:

1. **Why is a staging environment necessary?**
   - It allows testing in a production-like setting without affecting live users.
2. **What should my test report include?**
   - All test cases (passed and failed) with Test Case ID, Description, Steps, Expected Result, Actual Result, Status, and Remarks.
3. **How do I document performance testing?**
   - Use Lighthouse or GTmetrix to generate a performance report.

4. **What if major issues are found during staging tests?**
    - Document issues for now; resolution can be part of post-hackathon activities.
5. **What is the purpose of the README.md file?**
    - It provides an overview of the project, deployment steps, and results.