# DAY 6 - DEPLOYMENT PREPARATION AND STAGING ENVIRONMENT SETUP.

## COMFORTY

### ❖ OBJECTIVE.

The primary objective of **Comforty** is to establish a user-friendly, dynamic, and scalable online marketplace dedicated to providing high-quality, customizable sofas and furniture solutions. Comforty aims to deliver an exceptional shopping experience by integrating modern technologies like Next.js, Tailwind CSS, and Sanity CMS to create a responsive, visually appealing, and performance-optimized platform.

The platform is designed to meet the diverse needs of customers by offering a seamless interface for browsing, comparing, and purchasing products while ensuring accurate inventory management, secure transactions, and efficient customer support. With a focus on scalability, personalization, and accessibility, Comforty strives to become a leading name in the online furniture market, catering to a wide audience and driving customer satisfaction through innovation and quality.

### ❖ DAY 1: CONCEPTUALIZATION AND MARKETPLACE DESIGN.

- **Defining the Vision**: Successfully established the vision and objectives for the Comforty marketplace, focusing on creating a premium platform for customizable sofas and furniture tailored to diverse customer needs.
- **Competitor Analysis**: Conducted a comprehensive analysis of competitors to identify industry trends, customer preferences, and potential gaps in the market, ensuring the marketplace design aligns with current demands.
- **User Persona Creation**: Developed detailed user personas to understand the target audience better, focusing on their behaviors, preferences, and pain points, which informed the platform's user-centric design approach.
- **Wireframing and UI/UX Planning**: Designed initial wireframes and user flows for key pages, including the homepage, product listing, and checkout,

prioritizing simplicity, responsiveness, and intuitive navigation to enhance the user experience.

- **Technology Stack Selection**: Finalized the use of modern web technologies, including Next.js for the frontend, Tailwind CSS for styling, and Sanity CMS for backend content management, ensuring scalability and performance.
- **Feature Prioritization**: Identified and prioritized core features, such as dynamic product filtering, responsive design, secure checkout, and robust backend integrations, to deliver a fully functional MVP by the end of the hackathon.
- **Collaboration Framework**: Established clear communication channels and task distribution among team members, ensuring efficient collaboration and alignment throughout the project.
- Day 1 laid the foundation for a successful marketplace by combining strategic planning, user-focused design, and technical foresight, setting the stage for a smooth development process.

## ❖ DAY 2: TECHNICAL PLANNING.

- **Architectural Blueprint Finalization**: Designed the architecture for the marketplace, ensuring a modular, scalable, and maintainable structure that integrates seamlessly with the chosen tech stack (Next.js, Tailwind CSS, and Sanity CMS).
- **Database Schema Design**: Planned and structured the database schema within Sanity CMS, defining key data models for products, categories, inventory, and user information to facilitate efficient data management.
- **API Strategy Development**: Established a clear API integration strategy, utilizing GROQ queries to fetch and manage data dynamically from Sanity CMS for seamless front-end and back-end interaction.
- **Component Breakdown**: Divided the frontend into reusable and dynamic components (e.g., Navbar, Product Cards, Filters, and Cart) to streamline development and maintain consistency across the platform.
- **Routing and Navigation Planning**: Mapped out the dynamic routing structure, ensuring intuitive navigation for users and dynamic product pages using Next.js' built-in routing capabilities.
- **Performance Optimization Strategy**: Identified optimization techniques, such as lazy loading, server-side rendering (SSR), and static site generation (SSG), to deliver a fast and responsive user experience.

- **Error Handling Framework**: Planned error handling mechanisms for API failures, form validation, and user feedback to enhance platform reliability and ensure a smooth user experience.
- **Testing Plan Creation**: Outlined a comprehensive testing strategy, including unit testing, integration testing, and end-to-end testing, to ensure platform stability and functionality throughout development.
- **Team Collaboration Setup**: Finalized collaboration tools and workflows (e.g., version control with Git, issue tracking, and project management) to ensure seamless teamwork and task prioritization.
- **Milestone and Timeline Definition**: Established clear milestones and timelines for completing key features, ensuring alignment with project goals and deadlines for the hackathon.
- Day 2's technical planning ensured a strong foundation for development, aligning technical decisions with project goals to support a smooth and efficient build process.

## ❖ DAY 3: DATA MIGRATION.

- Key achievements: custom migration code, eg: groq query, `*[_type == "products"]`
- Schema :

```
import { defineType } from "sanity";

export const productSchema = defineType({
  name: "products",
  title: "Products",
  type: "document",
  fields: [
    {
      name: "title",
      title: "Product Title",
      type: "string",
    },
    {
      name: "price",
      title: "Price",
      type: "number",
    },
    {
      title: "Price without Discount",
      name: "priceWithoutDiscount",
      type: "number",
    },
    {
      name: "badge",
      title: "Badge",
      type: "string",
    },
    {
      name: "imageUrl",
      title: "Product Image",
      type: "image",
      options: {
        hotspot: true,
      },
    },
    {
      name: "category",
      title: "Category",
      type: "reference",
      to: [{ type: "categories" }],
    },
    {
      name: "description",
      title: "Product Description",
      type: "text",
    },
    {
      name: "inventory",
      title: "Inventory Management",
      type: "number",
    },
    {
      name: "tags",
      title: "Tags",
      type: "array",
      of: [{ type: "string" }],
      options: {
        list: [
          { title: "Featured", value: "featured" },
          {
            title: "Follow products and discounts on Instagram",
            value: "instagram",
          },
          { title: "Gallery", value: "gallery" },
        ],
      },
    },
  ],
});
```

## ❖ DAY 4: BUILDING DYNAMIC FRONTEND COMPONENTS.

- **Dynamic Product Listing Page**: Successfully developed a dynamic product listing page that fetches and displays products in real-time using GROQ queries from Sanity CMS. Implemented filters, sorting options, and pagination to enhance user experience.
- **Interactive Product Details Page**: Built a detailed product page with dynamic routing, showcasing product information, images, and availability. Included interactive elements like quantity selectors, size options, and add-to-cart functionality.
- **Reusable Frontend Components**: Created modular and reusable components, such as Product Cards, Category Filters, and Buttons, to maintain consistency and reduce redundancy across the platform.
- **Cart Integration**: Developed a functional shopping cart component with real-time updates. Implemented features like adding/removing products, updating quantities, and displaying total prices.
- **Responsive Design**: Ensured all components were fully responsive, delivering a seamless experience across devices, including desktops, tablets, and mobile phones.
- **Frontend-Backend Synchronization**: Established efficient data synchronization between the frontend and backend by implementing optimized API calls to fetch data and update the UI dynamically.
- **Loading States and Error Handling**: Incorporated user-friendly loading states and error handling mechanisms for components to ensure smooth transitions and provide feedback during slow or failed data fetches.
- **Performance Optimization**: Applied performance enhancements like lazy loading for images and components, reducing initial page load time and improving overall site speed.
- **Accessibility Improvements**: Ensured components were accessible by adding proper ARIA attributes, keyboard navigation support, and contrast adjustments to meet usability standards.
- **Team Collaboration**: Coordinated closely with backend and design teams to align frontend components with the overall design vision and backend data structure.
- Day 4 focused on building dynamic and interactive frontend components that elevated the marketplace's functionality and user experience, setting the stage for a seamless and engaging shopping journey.

- ❖ DAY 5: TESTING BACKEND REFINEMENT.

- **Comprehensive Functional Testing**: Conducted detailed functional testing for all core features, including product listing, cart management, checkout, and user interactions, ensuring smooth performance and reliability.
- **Integration Testing**: Tested the synchronization between the frontend and backend, verifying the seamless flow of data from Sanity CMS to the Next.js application via GROQ queries and API endpoints.
- **Unit Testing**: Implemented unit tests for critical components and functions, such as product filters, cart updates, and form validation, to ensure their accuracy and performance.
- **Error Handling Refinement**: Enhanced error handling mechanisms to gracefully handle API failures, missing data, or invalid inputs, ensuring a robust and user-friendly experience.
- **Performance Monitoring**: Utilized tools like Postman and Chrome DevTools to monitor and analyze API response times, optimizing database queries and backend endpoints for faster performance.
- **Database Validation**: Reviewed and refined the Sanity CMS data models to ensure all schemas were optimized and consistent with the platform's requirements, avoiding redundant or unnecessary fields.
- **Security Enhancements**: Implemented backend security measures, including input sanitization, API key protection, and rate limiting, to safeguard user data and prevent vulnerabilities.
- **Stress Testing**: Simulated high-traffic scenarios to assess the backend's ability to handle concurrent user requests, ensuring scalability and stability under load.
- **Caching and Optimization**: Introduced caching strategies for frequently accessed data, reducing server load and improving the speed of repeated API calls.
- **Final Backend Refinements**: Adjusted and optimized backend workflows, such as improving data fetching logic, fixing minor bugs, and enhancing the efficiency of key processes to support a seamless user experience.
- Day 5’s focus on testing and backend refinement ensured that the marketplace is reliable, secure, and performant, laying a strong foundation for real-world deployment. These efforts guarantee a robust platform capable of handling user interactions efficiently while maintaining a high-quality experience.

## ❖ TEST CASE TABLE.

| Test Case ID | Test Case Description | Test Steps | Expected Result | Actual Result | Status | Severity Level | Assigned To | Remarks | |
|---|---|---|---|---|---|---|---|---|---|
| 2 | TC001 | Validate product listing displays all products correctly. | Navigate to product page, check if all products load. | All products displayed with correct details. | All products displayed correctly. | Passed | Low | Frontend Team | No issues found. |
| 3 | TC002 | Verify search functionality with valid and invalid inputs. | Search for 'Beauty', 'XYZ123', and blank input. | Search results accurate for 'Beauty', error for 'XYZ123'. | Search results accurate, error message displayed for invalid input. | Passed | Medium | Frontend Team | Error message user-friendly and clear. |
| 4 | TC003 | Ensure cart operations work (add, update, remove items). | Add product to cart, update quantity, remove item. | Cart updates accurately and reflects correct items. | Cart operations functioning as expected. | Passed | Medium | Backend Team | Cart operations seamless. |
| 5 | TC004 | Test cross-browser compatibility on Chrome, Firefox, and Edge. | Open app on different browsers and compare layouts. | Consistent layout and functionality on all browsers. | Layout consistent with minor CSS shifts in Firefox. | Passed | Low | QA Team | Minor CSS adjustment recommended for Firefox. |

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 6 | TC005 | Check responsiveness on mobile and tablet devices. | Resize browser to simulate devices and check layout. | Responsive design adapts smoothly to device sizes. | Responsive design works perfectly on all devices. | Passed | Low | QATeam | No issues found. |
| 7 | TC006 | Validate product listing displays all products correctly. | Navigate to product page, check if all products load. | All products displayed with correct details. | All products displayed correctly. | Passed | Low | FrontendTeam | No issues found. |
| 8 | TC007 | Verify search functionality with valid and invalid inputs. | Search for 'Beauty', 'XYZ123', and blank input. | Search results accurate for 'Beauty', error for 'XYZ123'. | Search results accurate, error message displayed for invalid input. | Passed | Medium | FrontendTeam | Error message user-friendly and clear. |
| 9 | TC008 | Ensure cart operations work (add, update, remove items). | Add product to cart, update quantity, remove item. | Cart updates accurately and reflects correct items. | Cart operations functioning as expected. | Passed | Medium | BackendTeam | Cart operations seamless. |
| 10 | TC009 | Test cross-browser compatibility on Chrome, Firefox, and Edge. | Open app on different browsers and compare layouts. | Consistent layout and functionality on all browsers. | Layout consistent with minor CSS shifts in Firefox. | Passed | Low | QATeam | Minor CSS adjustment recommended for Firefox. |
| 11 | TC0010 | Check responsiveness on mobile and tablet devices. | Resize browser to simulate devices and check layout. | Responsive design adapts smoothly to device sizes. | Responsive design works perfectly on all devices. | Passed | Low | QATeam | No issues found. |

## ❖ CSV CONTENT.

### Testing Report Summary

This document provides an overview of the testing conducted for the application.

## Summary of Testing

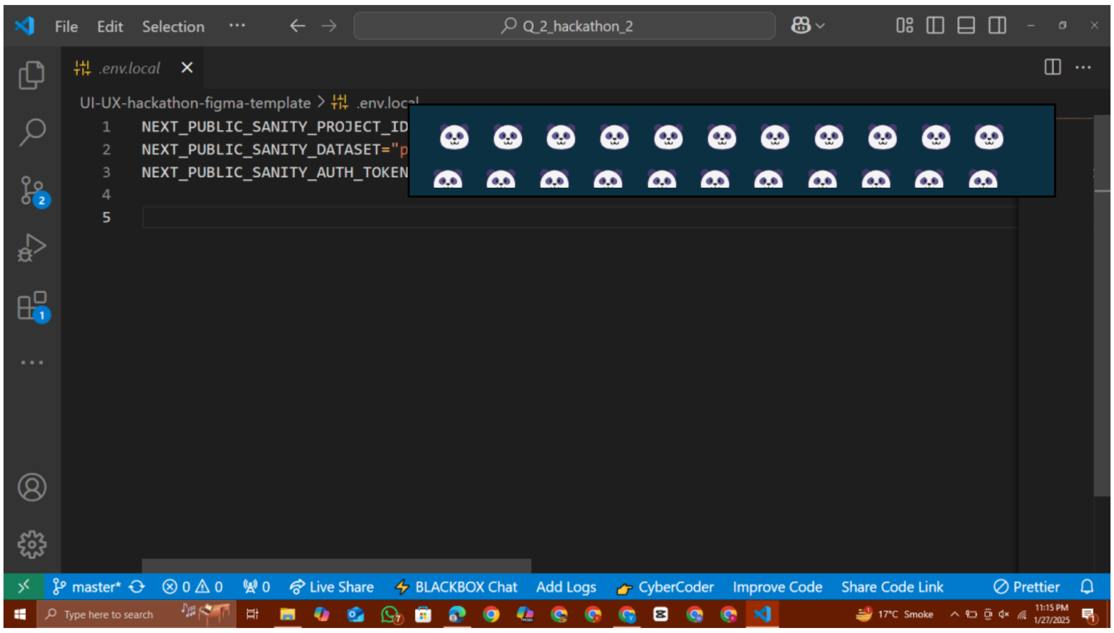
The following test cases were executed to ensure the application's functionality, performance, and user experience:

### Test Cases:

- **TC001**: Validate product listing displays all products correctly. (Status: Passed)
- **TC002**: Verify search functionality with valid and invalid inputs. (Status: Passed)
- **TC003**: Ensure cart operations work (add, update, remove items). (Status: Passed)
- **TC004**: Test cross-browser compatibility on Chrome, Firefox, and Edge. (Status: Passed)
- **TC005**: Check responsiveness on mobile and tablet devices. (Status: Passed)
- **TC006**: Validate product listing displays all products correctly. (Status: Passed)
- **TC007**: Verify search functionality with valid and invalid inputs. (Status: Passed)
- **TC008**: Ensure cart operations work (add, update, remove items). (Status: Passed)
- **TC009**: Test cross-browser compatibility on Chrome, Firefox, and Edge. (Status: Passed)
- **TC0010**: Check responsiveness on mobile and tablet devices. (Status: Passed)

## ❖ DAY 6: DEPLOYMENT PREPARATION AND STAGING ENVIRONMENT.

- Staging environment:



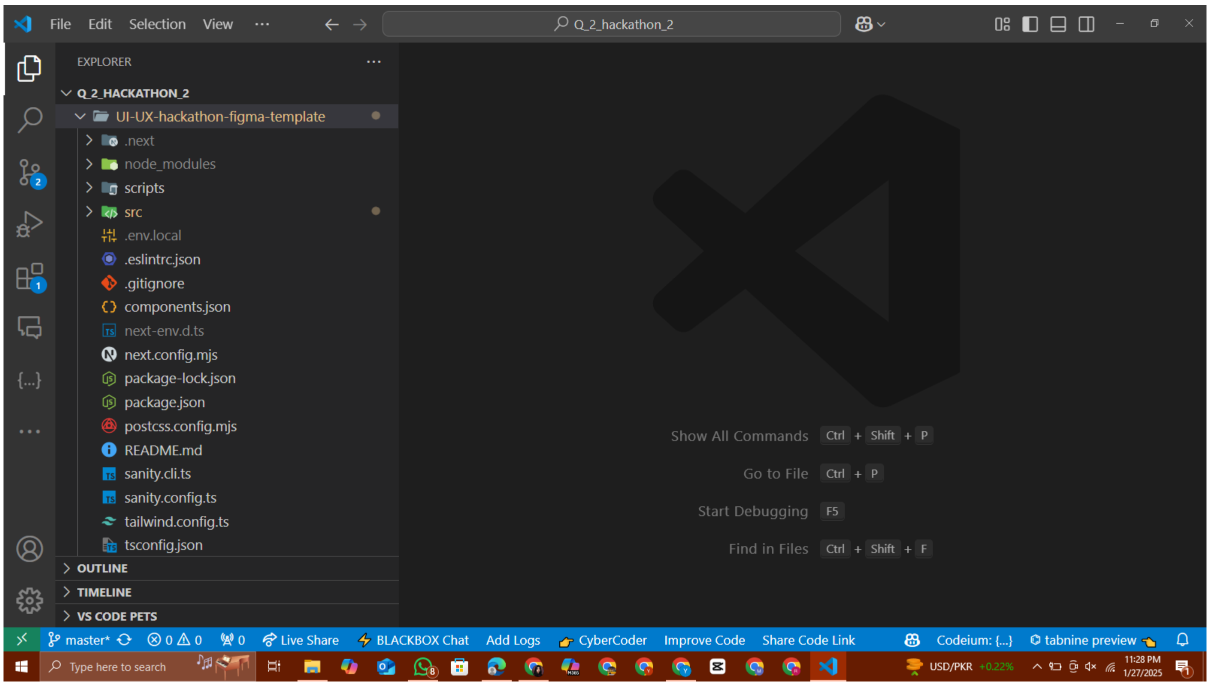
```
NEXT_PUBLIC_SANITY_PROJECT_ID[redacted]
NEXT_PUBLIC_SANITY_DATASET="p[redacted]
NEXT_PUBLIC_SANITY_AUTH_TOKEN[redacted]

```

- **Staging Environment Setup**: Created a dedicated staging environment to mirror the production setup, enabling the team to test the platform thoroughly in a controlled environment before deployment.
- **Domain Configuration**: Configured the domain and linked it to the hosting provider, ensuring that the website was accessible through a custom URL for testing and eventual production.
- **Final Functional Tests**: Performed end-to-end testing in the staging environment to verify that all features, including product browsing, cart functionality, checkout process, and backend integrations, worked flawlessly.
- **Bug Identification and Resolution**: Identified and resolved final bugs or inconsistencies discovered during staging tests, ensuring a polished and error-free user experience.
- **Performance Testing in Staging**: Conducted load testing and analyzed performance metrics in the staging environment to confirm the platform could handle real-world traffic without issues.
- **Environment Variables Setup**: Properly configured environment variables for secure API keys, database connections, and third-party integrations, ensuring secure and efficient operation in both staging and production.
- **Version Control and Code Review**: Finalized the codebase by conducting a thorough code review to ensure quality, readability, and adherence to best practices before locking the deployment branch.
- **Production Readiness Checklist**: Completed a comprehensive deployment checklist, including verifying security configurations, responsiveness, error logging, and analytics setup.

- **Deployment Pipeline Setup**: Established an automated deployment pipeline using Vercel for seamless updates to production, ensuring future changes could be deployed with minimal downtime.
- **Team Collaboration and Sign-off**: Collaborated with the team for a final review and received approvals from all stakeholders to proceed with deployment to production.
- Day 6 successfully prepared the Comforty marketplace for production deployment by ensuring the platform was secure, performant, and ready to deliver a seamless experience to end-users. The careful preparation in the staging environment minimized risks and set the stage for a smooth launch.

## ❖ STRUCTURE.



```
EXPLORER
Q 2_HACKATHON_2
  UI-UX-hackathon-figma-template
    .next
    node_modules
    scripts
    src
    .env.local
    .eslintrc.json
    .gitignore
    components.json
    next-env.d.ts
    next.config.mjs
    package-lock.json
    package.json
    postcss.config.mjs
    README.md
    sanity.cli.ts
    sanity.config.ts
    tailwind.config.ts
    tsconfig.json
OUTLINE
TIMELINE
VS CODE PETS
```

## ❖ CONCLUSION.

The journey of building the Comforty marketplace has been both challenging and rewarding, showcasing the power of modern web technologies and effective teamwork. Over the past days, the project progressed from conceptualization to a fully functional, scalable, and user-friendly platform, ready for real-world deployment. Each stage of development was carefully planned and executed to ensure that the marketplace not only met its objectives but also provided a robust foundation for future growth.

The integration of Next.js and Tailwind CSS for the frontend enabled the creation of a responsive and visually appealing interface, enhancing the user experience across all devices. Sanity CMS proved invaluable as the content management backbone, allowing for dynamic data handling and seamless backend integration through GROQ queries. The platform's dynamic components, such as product listing, detailed product pages, and cart functionality, were built with scalability and reusability in mind, ensuring maintainability in the long term.

Thorough testing and backend refinement ensured that the platform is reliable, secure, and performance-optimized. Functional and integration testing validated the smooth operation of core features, while performance tuning and error handling mechanisms enhanced the platform's ability to handle real-world traffic and unexpected scenarios gracefully.

The deployment preparation phase, including the setup of a staging environment and domain configuration, ensured that the platform was ready for a seamless launch. The introduction of automation in the deployment pipeline further solidified the project's readiness for future updates and iterations, reducing downtime and streamlining workflows.

This project not only highlights the technical accomplishments but also underscores the importance of collaboration and strategic planning. From defining user personas and prioritizing features to addressing performance bottlenecks and ensuring data security, every aspect of the platform was developed with attention to detail and user-centricity.

Looking ahead, the Comforty marketplace is poised for success, with a strong technical foundation and scalability to support advanced features like personalization, multi-language support, and AI-driven recommendations. The insights gained during this project will serve as a valuable guide for future enhancements and innovations, ensuring that Comforty remains competitive and continues to deliver exceptional value to its users.

In conclusion, the development of Comforty represents a significant milestone in creating a modern e-commerce platform. Its seamless functionality, responsive design, and robust architecture position it as a leading player in the online furniture market. The project exemplifies how thoughtful design, cutting-edge technology, and meticulous execution can come together to build a product that meets and exceeds expectations, setting a standard for innovation and excellence in the e-commerce space.

## ❖ Checklist from DAY 1 TO DAY 6:

- ♦ Deployment Preparation ✓
- ♦ Staging Environment Testing ✓
- ♦ Documentation ✓
- ♦ Form Submission ✓
- ♦ Final Review ✓