My Experience Building an E-commerce Website: Challenges, Learnings, and Suggestions

Starting the Project

When I started working on my e-commerce website project, I had a clear vision of what I wanted to achieve: an easy-to-use platform offering various dresses, from traditional to western wear. However, bringing that vision to life was no small task. Here's my journey, the challenges I faced, what I learned, and areas where improvements can be made.

Phase 1: Setting Up the Project

Challenges:

1. **Understanding the Requirements:**
   - Determining the best structure for the project was tricky as there are many ways to organize a Next.js app.
   - Choosing the features for the website (like categories, filters, cart, and payment integration) took time.
2. **Environment Setup:**
   - Setting up the Next.js framework with TypeScript was smooth initially, but ensuring compatibility with other tools like Sanity CMS introduced errors.
   - I faced issues configuring the `.env` file and making sure sensitive data was hidden properly.

Learnings:

- Plan out the project's folder structure beforehand.
- Always verify environment variables work both locally and in production.
- Use tools like `dotenv` for local development to manage sensitive information.

Phase 2: Data Integration and APIs

Challenges:

1. **Fetching Data from APIs:**
   - While connecting the website to my backend API (`https://template-03-api.vercel.app/api/products`), data was not being displayed on the live site despite working on localhost.
   - Setting up environment variables on Vercel for production required multiple tries to debug why data was missing.
   - Issues like "No products found" errors due to missing API responses took hours to resolve.
2. **Sanity CMS Integration:**
   - Uploading product images to Sanity and mapping them to the products took longer than expected.
   - Properly setting up the Sanity client with tokens was confusing.

Learnings:

- Always test API endpoints using tools like Postman before integrating them into the frontend.
- Keep a checklist for setting up environment variables and test them after deployment.
- Sanity's documentation is very helpful—relying on it saved me a lot of time.

Phase 3: Frontend Development

Challenges:

1. **Styling:**
    - Making the website responsive was difficult as I needed to ensure it worked well on both mobile and desktop devices.
    - Managing global styles in `global.css` and keeping component-specific styles consistent was challenging.
2. **Dynamic Routing:**
    - Implementing dynamic routes in Next.js for product pages (`/products/:id`) required a clear understanding of the app folder structure.
    - Issues with server-side rendering (SSR) and static site generation (SSG) caused mismatches between server and client HTML.

Learnings:

- Always test the site on multiple devices and screen sizes during development, not after.
- Understand the difference between SSR, SSG, and client-side rendering (CSR) to pick the best option for each page.
- Dynamic routing in Next.js is powerful but requires clean folder structuring and naming conventions.

Phase 4: Deploying to Vercel

Challenges:

1. **Deployment Errors:**
    - After deployment, the API data wasn't loading on the live site. This was due to missing environment variables.
    - Hydration errors occurred due to mismatches in server-rendered and client-rendered content.
2. **Environment Variables:**
    - Setting up `NEXT_PUBLIC_` variables correctly on Vercel took several attempts.
    - Debugging missing variables or incorrect configurations slowed down progress.

Learnings:

- Environment variables must be set for all environments (production, preview, development) in the Vercel dashboard.
- Use Vercel logs to debug runtime issues—they provide great insight into what's wrong.

Phase 5: Adding Features

Challenges:

1. **Shopping Cart Logic:**
    - Managing the cart state globally across components required learning context API effectively.
    - Ensuring cart items persisted between page reloads required additional localStorage integration.
2. **Payment Integration:**
    - Integrating payment gateways like Stripe was complicated due to token handling and API security concerns.

3. **Filters and Search:**
    - Implementing efficient filtering for categories and price ranges required optimizing API calls and frontend rendering.

Learnings:

- Break down complex features into smaller steps to handle them one at a time.
- Use local Storage or a backend database to persist cart and user data effectively.
- Keep the UI simple for filters and search to ensure smooth functionality.

General Learnings:

- **Debugging Skills:** Logs (both server and browser) are your best friends when things go wrong.
- **Patience:** Many issues took hours or even days to fix, but persistence paid off.
- **Documentation is Key:** Reading through Next.js and Sanity documentation helped solve many problems.
- **Testing:** Testing both locally and on the live site ensures consistency.

Suggestions for Improvement:

1. **Project Planning:**
    - Allocate time for learning new tools before implementing them.
    - Use a project management tool like Trello to track progress and avoid missing steps.
2. **Backend and API Handling:**
    - Always test API endpoints before integrating them.
    - Keep a fallback mechanism in case an API fails to load (e.g., a “No products available” message).
3. **Collaboration:**
    - Share the project with teammates or mentors early for feedback and improvements.
4. **UI/UX Testing:**
    - Get feedback from users for design improvements and ensure the site is intuitive to use.
5. **Deployment:**
    - Double-check environment variables before deploying.
    - Monitor logs regularly after deployment to catch and fix bugs quickly.

Final Thoughts

Building an e-commerce website was a challenging but incredibly rewarding experience. It required me to step out of my comfort zone, learn new technologies, and troubleshoot complex issues. Every error taught me something new, and by the end of the project, I felt more confident in my development skills. For anyone attempting a similar project, my advice is to stay patient, read the documentation, and take one step at a time. You’ve got this!