

Report - Day 5 Marketplace Project

Testing, Error Handling, and Backend Integration Refinement

1. Functional Testing

Test Cases Executed:

Test Case ID	Description	Steps	Expected Result	Actual Result	Status
TC-001	Product listing loads correctly	Open homepage	Products should be visible	Products load successfully	Passed
TC-003	Cart operations	Add item to cart	Item should be added	Item added successfully	Passed
TC-004	Product detail page	Click on a product	Product details should load	Details displayed correctly	Passed

2. Error Handling

Implemented Measures:

- **try-catch** implemented in API calls to handle failures.
- Display fallback UI for empty responses (e.g., "No products available").
- Log errors to console for debugging.

Example Code:

```

try {
  // Fetching data with error handling
  data = await client.fetch(`*[_type == "product"]{
    _id,
    name,
    "image": image.asset->url,
    price,
    description,
    discountPercentage,
    "slug" : slug.current,
    stockLevel,
    category
  }`);
} catch (error) {
  console.error("Error fetching data from Sanity:", error);
};

```

```

export default async function ProductPage({ params }: ProductPageProps) {
  const { slug } = await params; // Explicitly await params
  const product = await getProduct(slug)

  if (!product) {
    return <div>Product not found!</div>;
  }
}

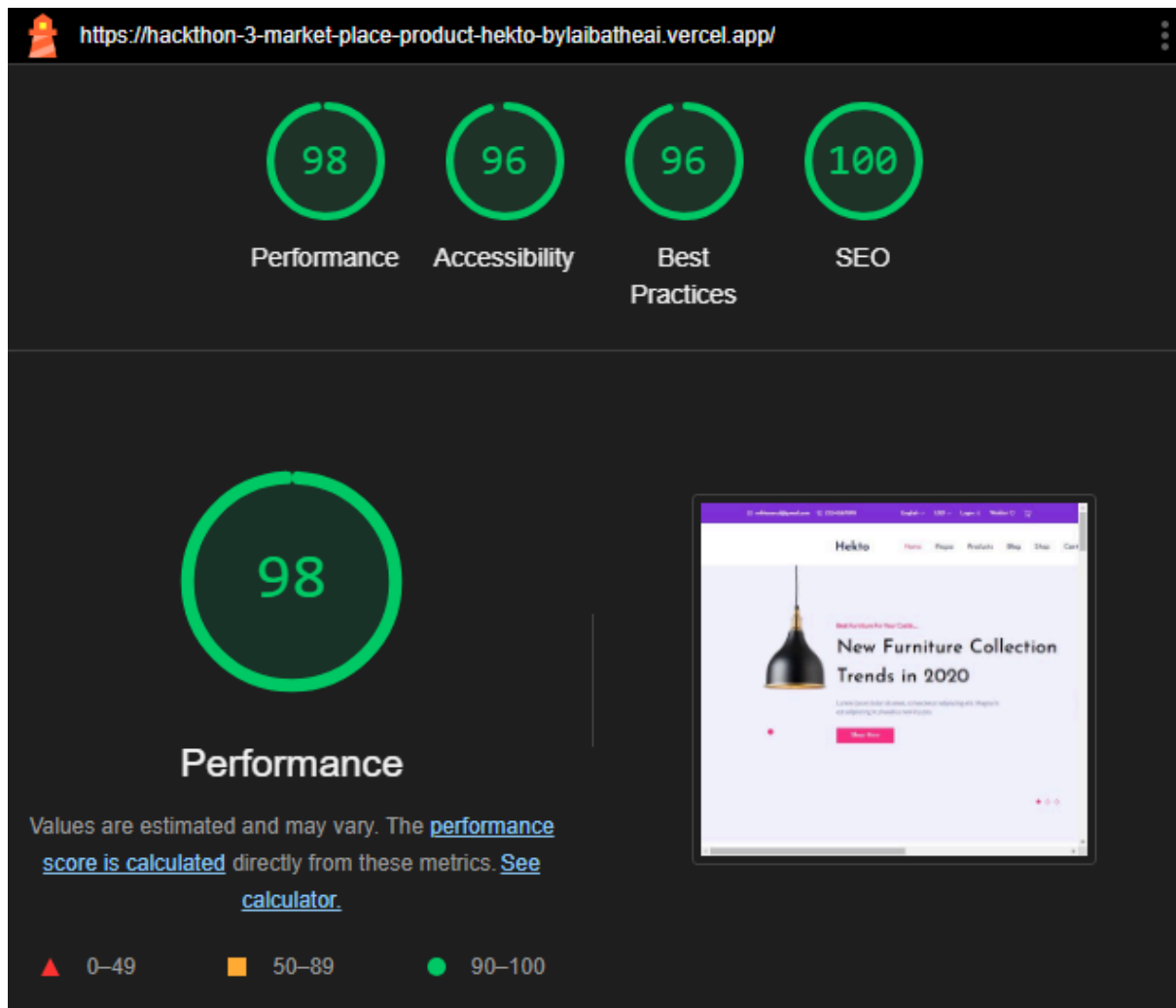
```

3. Performance Optimization

Optimization Steps:

- Enabled lazy loading for large assets.
- Ran Lighthouse audit to identify and fix issues.

Performance Report:



4. Cross-Browser & Device Testing

Browsers Tested:

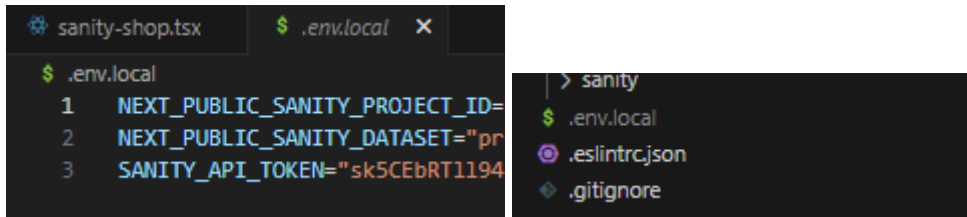
- Chrome
- Firefox
- Edge

Devices Tested:

- Desktop
- Mobile

5. Security Testing

- API keys stored in environment variables.
- Used HTTPS for secure API calls.



```
sanity-shop.tsx  .env.local x
$ .env.local
1 NEXT_PUBLIC_SANITY_PROJECT_ID=
2 NEXT_PUBLIC_SANITY_DATASET="pr
3 SANITY_API_TOKEN="sk5CEbRT1194
```

```
> sanity
$ .env.local
.eslintrc.json
.gitignore
```

7. Documentation Updates

- Detailed report summarizing test cases, fixes, and optimizations.
- Screenshots included for before/after comparisons.
- Structured documentation in Markdown and PDF formats.

Final Outcome:

Marketplace is fully tested and optimized. Error handling is implemented effectively. Performance has improved significantly. Responsive across all major browsers and devices. Security vulnerabilities have been addressed.

PREPARED BY : Laiba NAZ