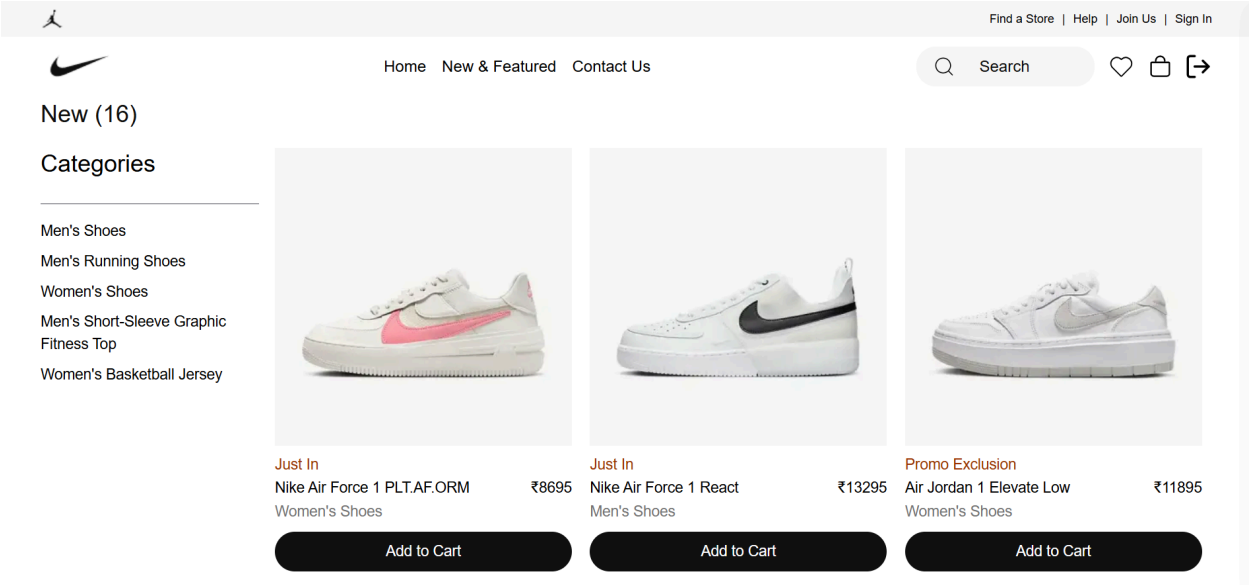
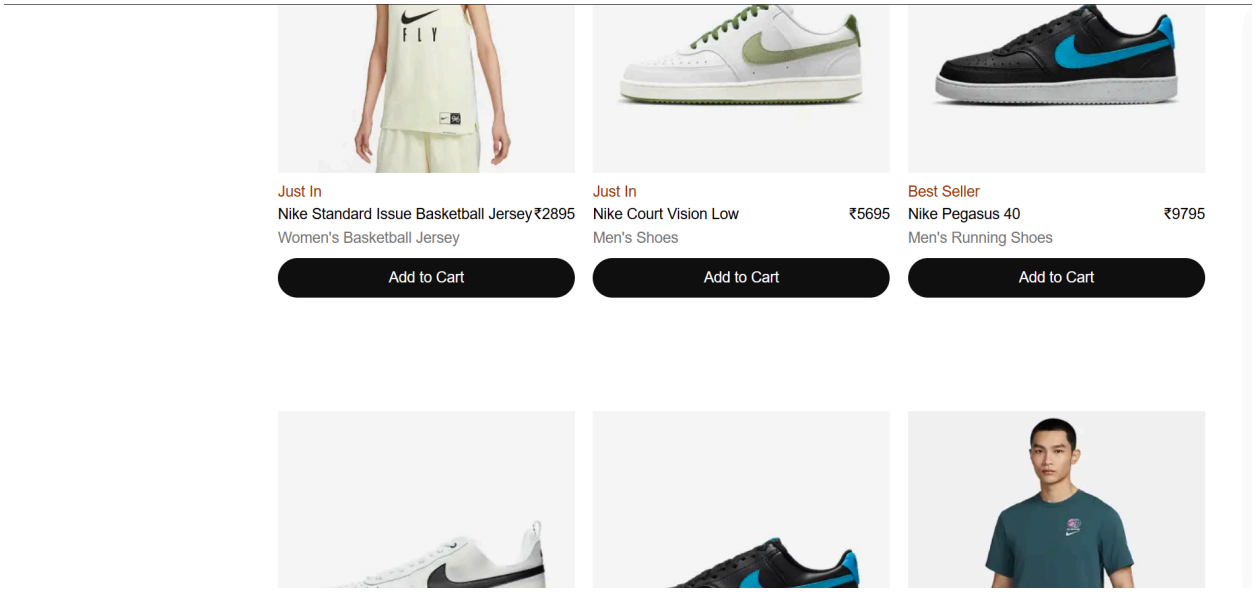
Testing, Error Handling and Backend Integration

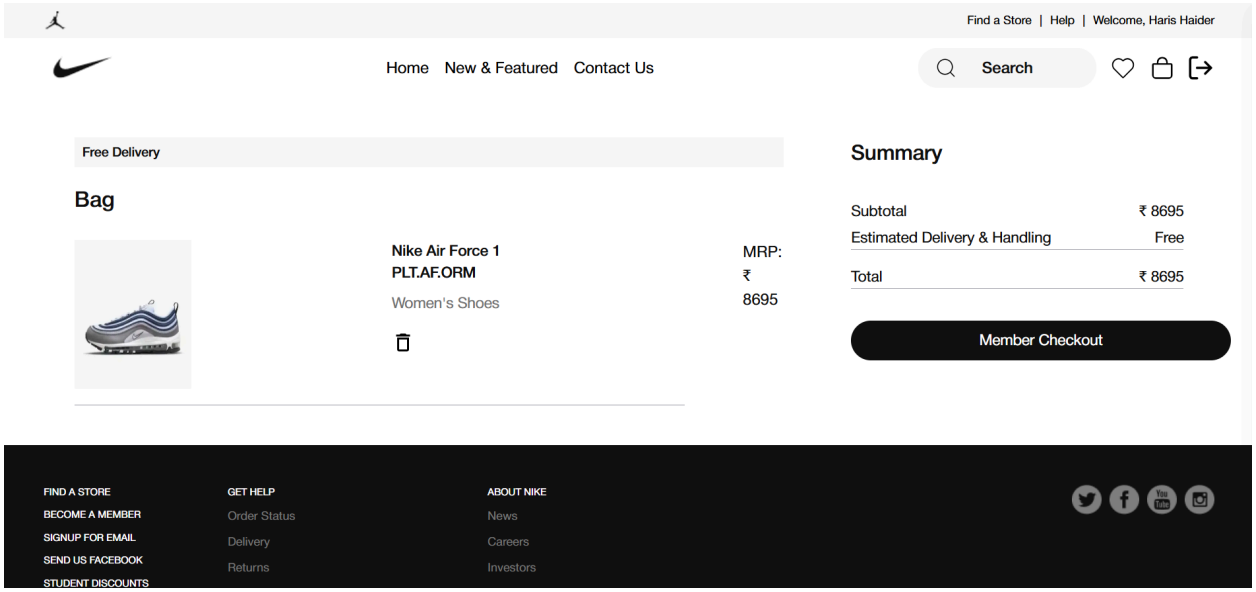
The task for day 5 was to focus on testing, error handling with backend integration following are the steps i did during day 5:

1. Functional Testing:

- Task: To Check that all the features of a marketplace is properly working or not.
  - ☐ Checked All the products are listed correctly with dynamic data like name, price, image, tags etc.
  - ☐ Check Product Details pages are correctly set up and showing dynamic data coming from Sanity CMS.
  - ☐ Tested addToCart functionality is properly working with subTotal price and products.

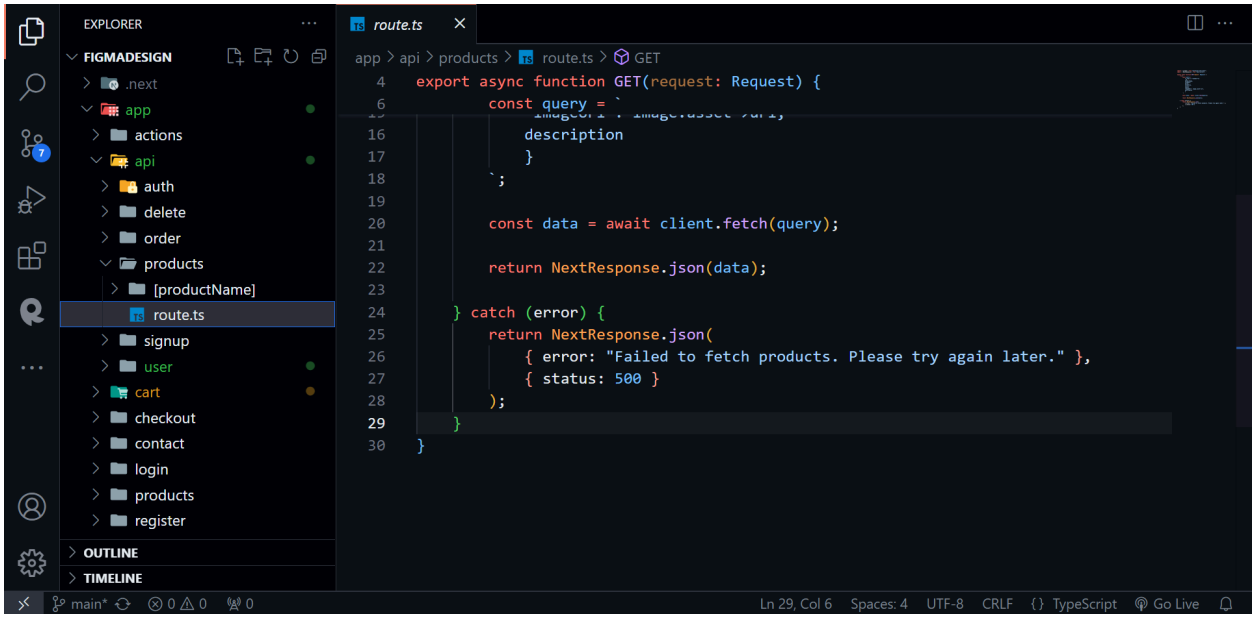


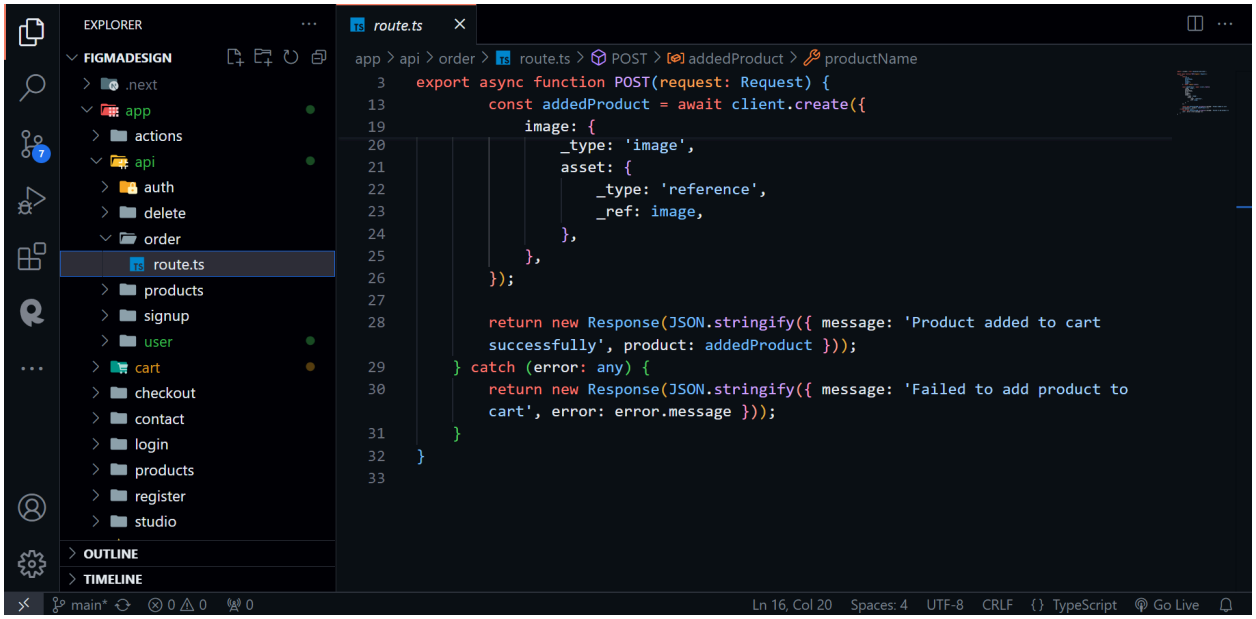


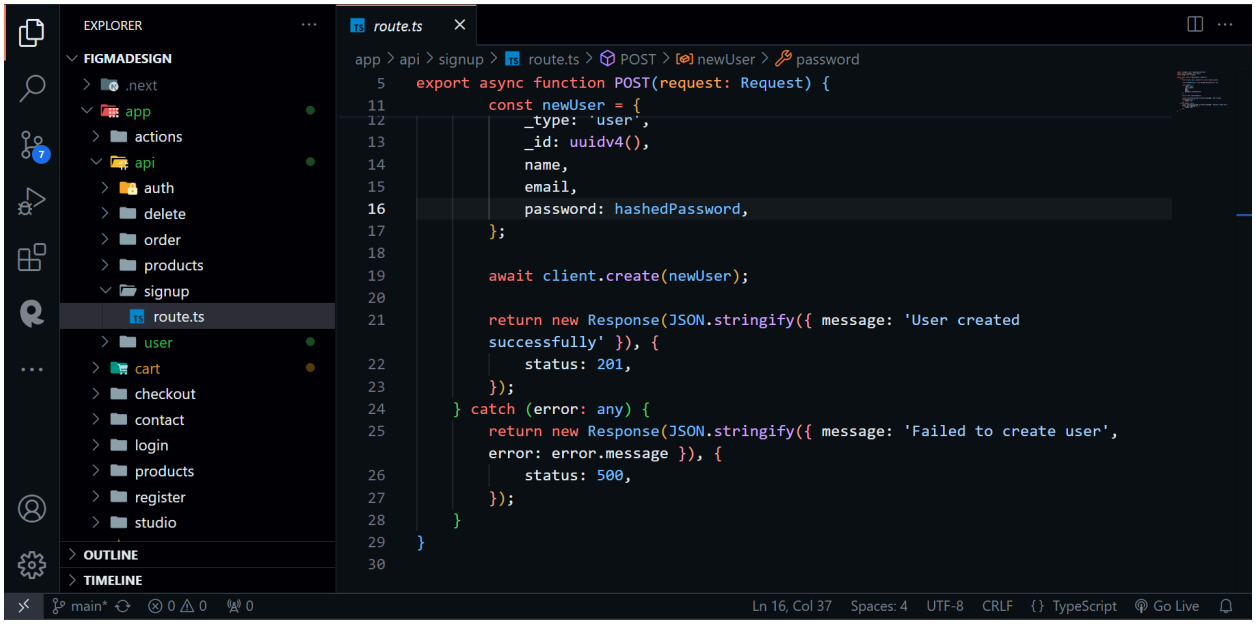


2. Error Handling:

- Implemented Error Handling at products fetching, details and in cart adding functionalities.
- Result: React-Hot-Toast will notify with both error and success message.

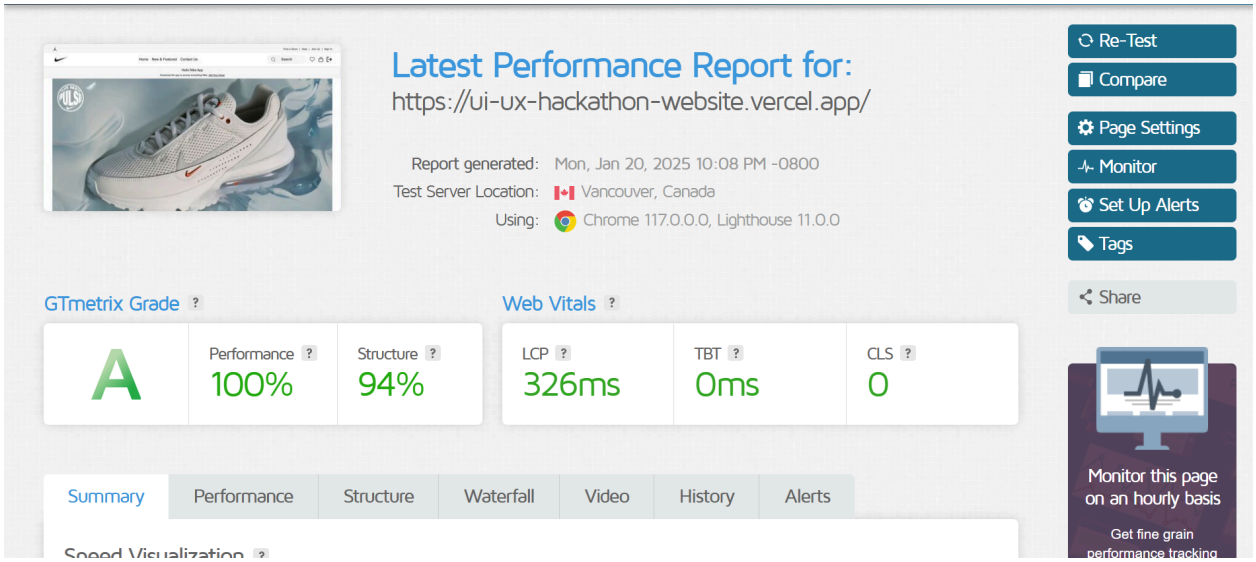

EXPLORER
FIGMADESIGN
.next
app
actions
api
auth
delete
order
products
[productName]
route.ts
signup
user
cart
checkout
contact
login
products
register
OUTLINE
TIMELINE
route.ts
app > api > products > route.ts > GET
export async function GET(request: Request) {
const query = `
description
}
`;
const data = await client.fetch(query);
return NextResponse.json(data);
} catch (error) {
return NextResponse.json(
{ error: "Failed to fetch products. Please try again later." },
{ status: 500 }
);
}
}
main*
Ln 29, Col 6
Spaces: 4
UTF-8
CRLF
TypeScript
Go Live


EXPLORER
FIGMADESIGN
.next
app
actions
api
auth
delete
order
route.ts
products
signup
user
cart
checkout
contact
login
products
register
studio
OUTLINE
TIMELINE
route.ts
app > api > order > route.ts > POST > addedProduct > productName
export async function POST(request: Request) {
const addedProduct = await client.create({
image: {
_type: 'image',
asset: {
_type: 'reference',
_ref: image,
},
},
});
return new Response(JSON.stringify({ message: 'Product added to cart successfully', product: addedProduct }));
} catch (error: any) {
return new Response(JSON.stringify({ message: 'Failed to add product to cart', error: error.message }));
}
}
main*
Ln 16, Col 20
Spaces: 4
UTF-8
CRLF
TypeScript
Go Live



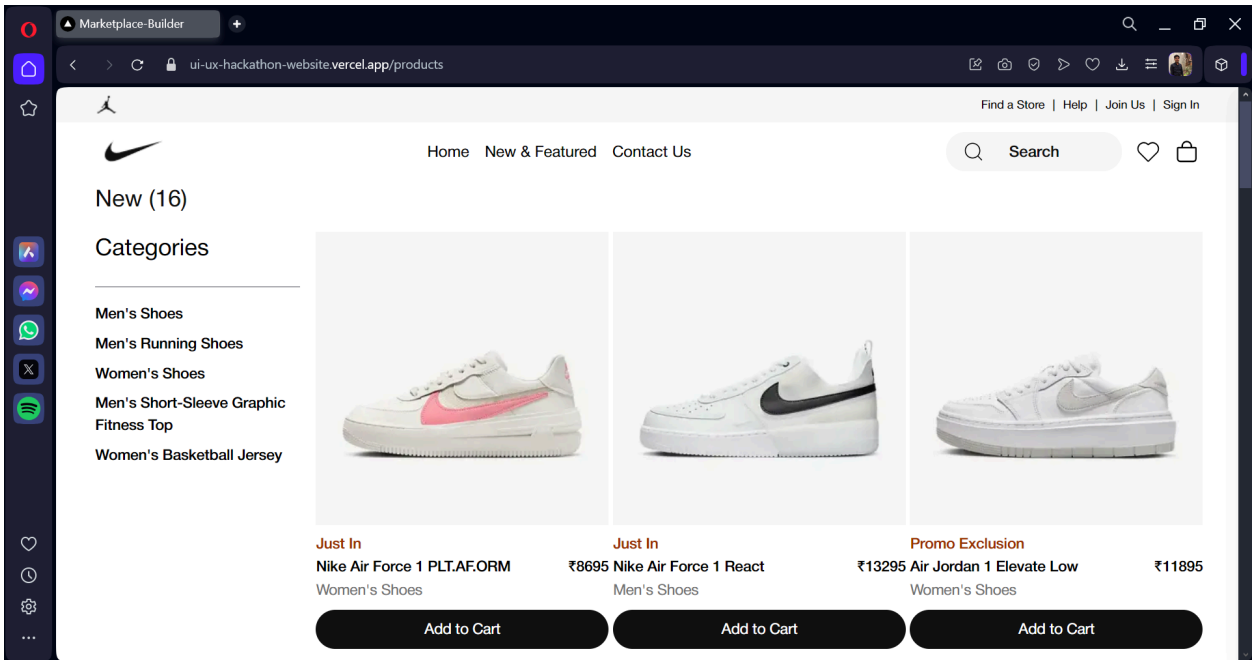
3. Performance Testing:

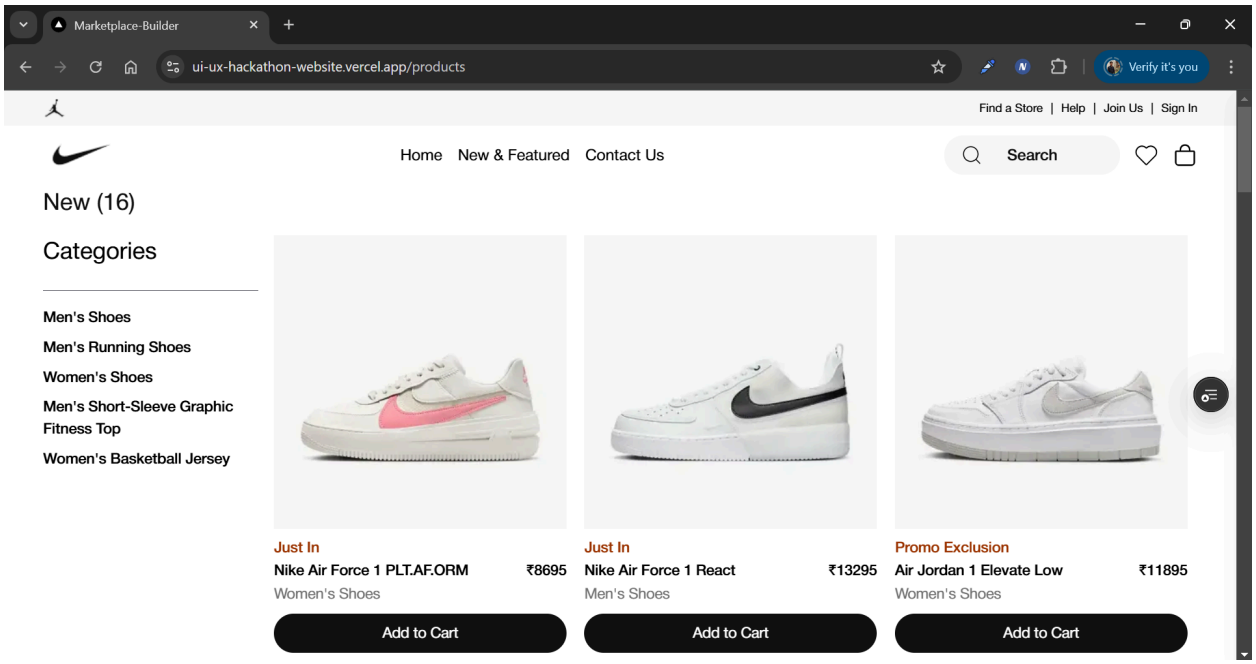
- I Tested performance of a website on a platform called gtmetrix.com

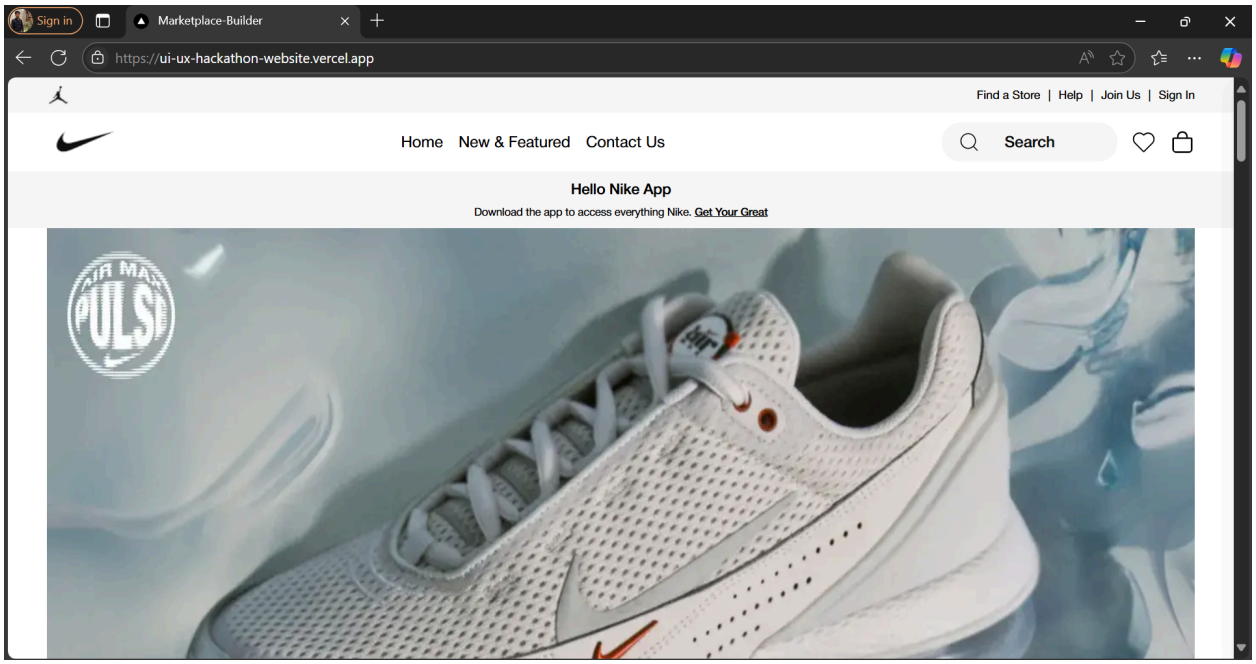


4. Cross Browser and Device Testing:

- Task: Ensure capability and speed across different browsers.
  - ☐ Tested on Chrome, Brave, Opera and Edge browser.
  - ☐ Tested on mobile phones two different resolution laptops and one tablet.

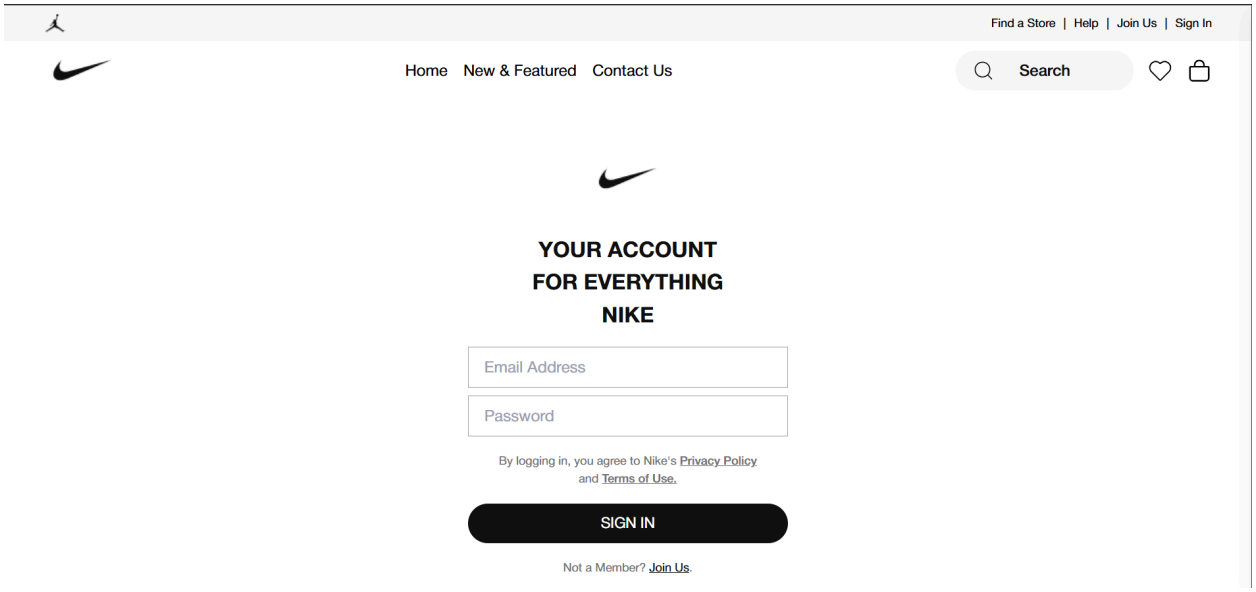


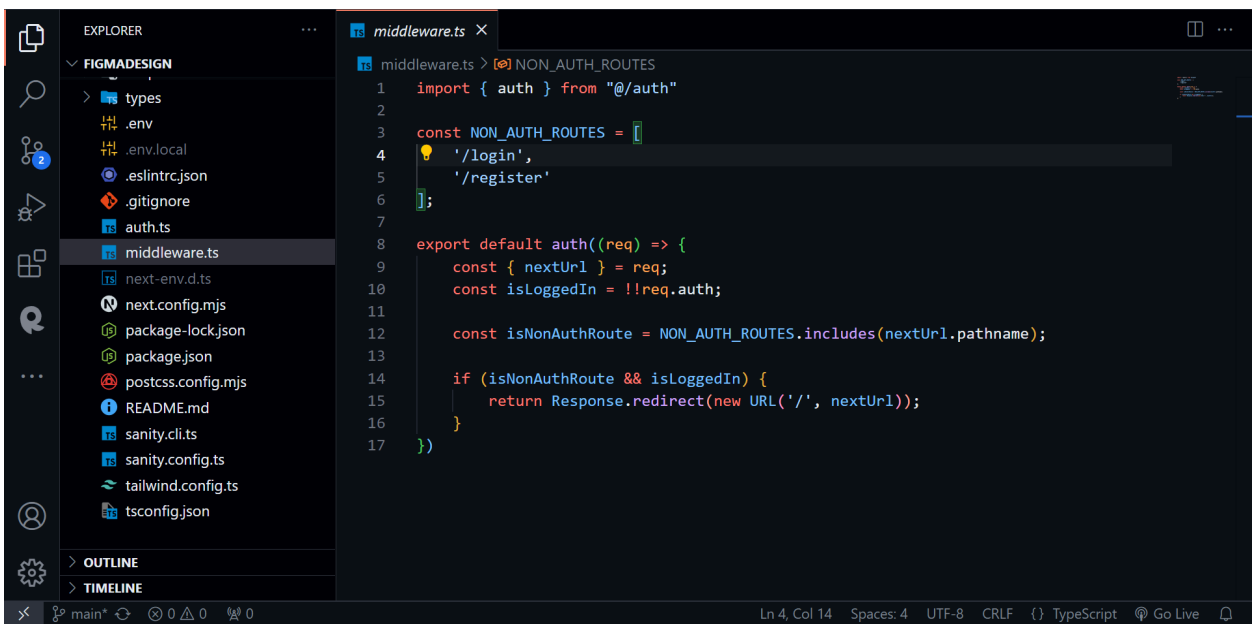




5. Security Testing:

- Marketplace is secured by authentication using Auth.js only authenticated users with all details can place an order.
- Also Implemented middleware to enhance security.





6. User Acceptance Testing (UAT):

- Collected feedback from non tech users.
  Feedback:
  - ☐ Got positive feedback with few enhancements.

7. Documentation Updates:

- Updated documentation to reflect changes in schema and functionalities.

Updated Schema according to use case:

```
export const productSchema = {
    name: 'product',
    title: 'Product',
    type: 'document',
    fields: [
        {
            name: 'productName',
            title: 'Product Name',
            type: 'string',
        },
        {
            name: 'category',
            title: 'Category',
            type: 'string',
        },
        {
            name: 'price',
            title: 'Price',
            type: 'number',
        },
        {
            name: 'inventory',
            title: 'Inventory',
            type: 'number',
        },
        {
            name: 'colors',
            title: 'Colors',
            type: 'array',
            of: [{ type: 'string' }],
        },
        {
            name: 'status',
            title: 'Status',
            type: 'string',
        },
        {
            name: 'image',
            title: 'Image',
```

```
            type: 'image',
            options: {
                hotspot: true,
            },
        },
        {
            name: 'description',
            title: 'Description',
            type: 'text',
        },
    ],
}
```

Conclusion:

- Today’s task updated my application and enhance its features and functionality.