Steps Taken to Build and Integrate Components

1. **Requirement Analysis:**
    a. Identified key components required for the project (e.g., header, footer, login/signup, product display, etc.).
    b. Broke down the features into smaller, reusable components.
2. **Component Structure:**
    a. Created a modular folder structure, such as components, pages, and styles.
    b. Used functional components with clear naming conventions.
3. **Design and UI/UX:**
    a. Designed wireframes or mockups to visualize component placement and functionality.
    b. Ensured responsiveness and accessibility during design.
4. **Development:**
    a. Developed individual components (e.g., buttons, cards, navigation bars) using HTML, CSS, and JavaScript/TypeScript.
    b. Utilized props and state in Next.js to make components dynamic.
    c. Integrated external data (e.g., APIs or CMS like Sanity) using server-side rendering (getServerSideProps) or static generation (getStaticProps).
5. **Testing and Debugging:**
    a. Tested components in isolation using tools like Storybook.
    b. Debugged functionality during integration to ensure proper interaction between components.

Challenges Faced and Solution

- **Challenge: State Management for Data Sharing**
  - Solution: Implemented React Context API to manage shared state across components instead of prop-drilling.

- **Challenge: Integrating External APIs or CMS**
  - Solution: Researched and used official documentation to configure API keys securely via .env files and ensure error handling during API requests.

- **Challenge: Responsiveness Issues**
  - Solution: Used CSS frameworks like Tailwind CSS or media queries to adjust layouts for various screen sizes effectively.

- **Challenge: Deployment Errors**
  - Solution: Resolved issues by fixing environment variable setups and ensuring compatibility with hosting platforms like Vercel or Netlify.

- **Challenge: Limited Time for Optimization**
  - Solution: Focused on essential functionality first (MVP) and planned enhancements for later.

Best Practices Followed During Development

1. **Modularity and Reusability:**
   - Designed components to be modular and reusable, reducing redundancy.

1. **Version Control:**
   - Used Git/GitHub for version control to track changes and collaborate effectively.

1. **Performance Optimization:**
   - Lazy-loaded images and components using Next.js features like next/image and dynamic imports.

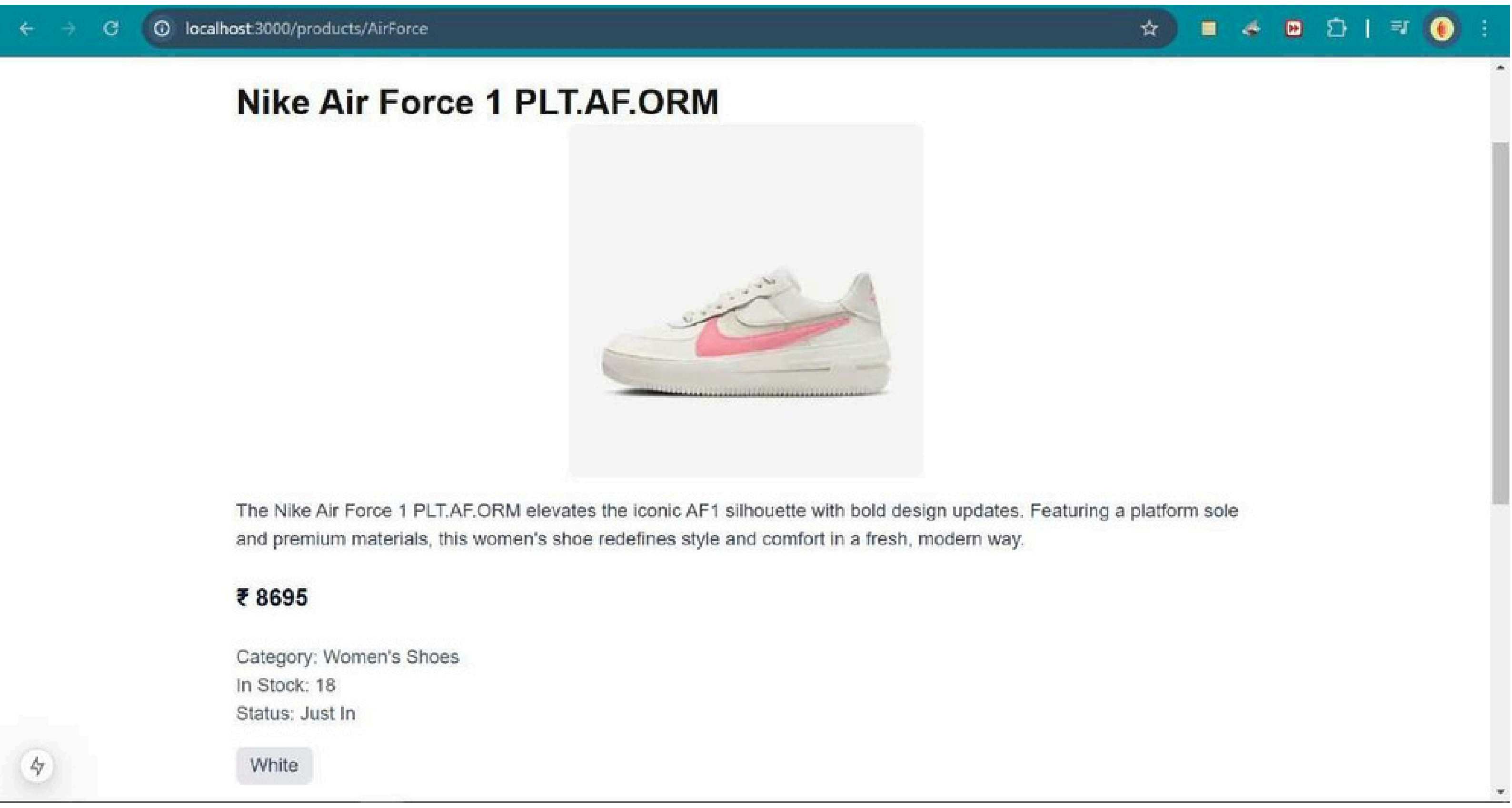
1. **Clear Coding Standards:**
   - Followed naming conventions and clean code principles (e.g., meaningful variable names, proper indentation).

1. **Responsive and Accessible Design:**
   - Used semantic HTML and ARIA roles to make the website accessible to all users.

1. **Testing:**
   - Ensured cross-browser compatibility by testing on Chrome, Firefox, and Safari.
   - Debugged and resolved console errors and warnings.

1. **Documentation:**
   - Documented key features and component usage for easier understanding by team members or future developers.

Screen Shots



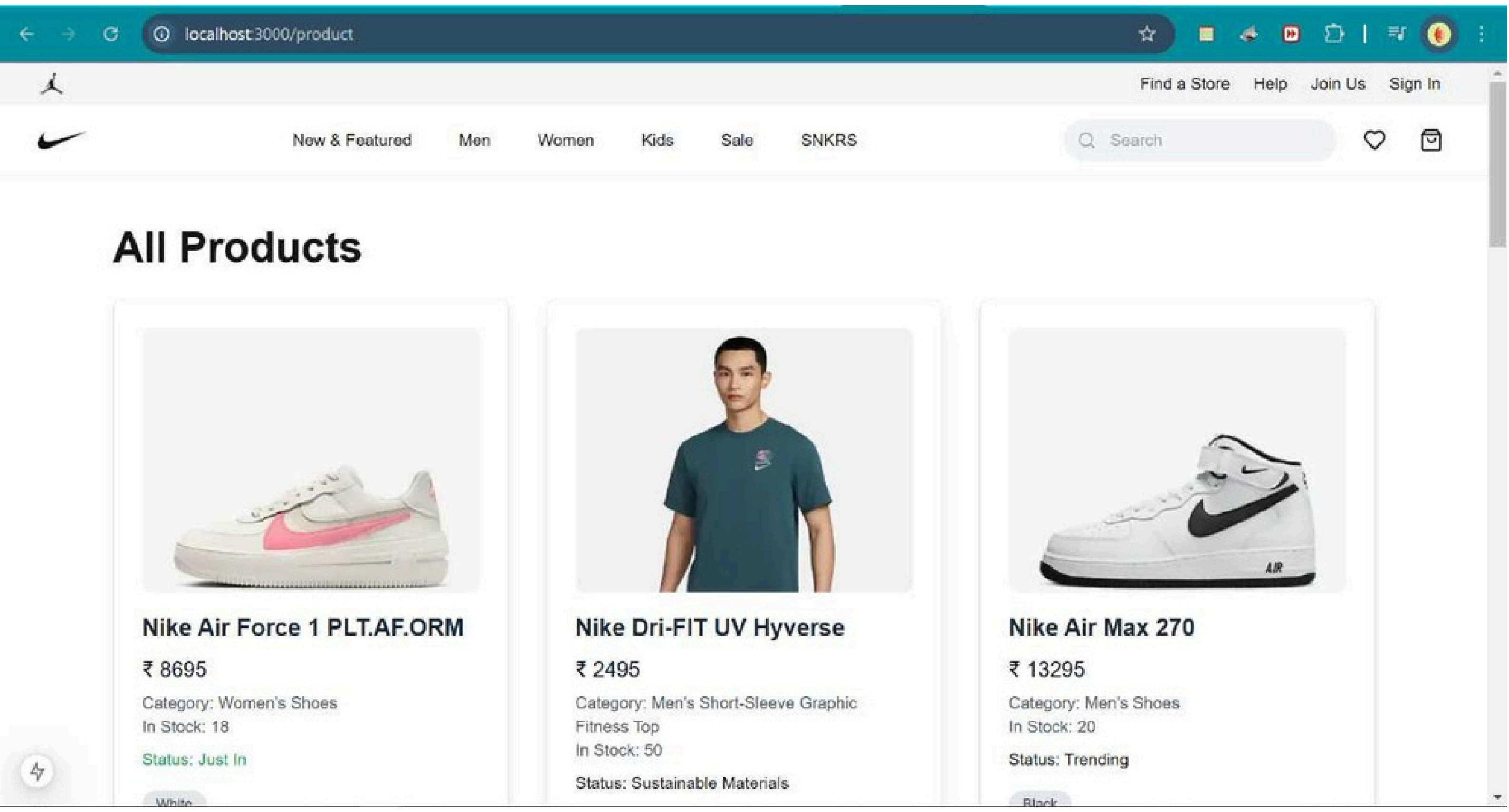
localhost:3000/products/AirForce

Nike Air Force 1 PLT.AF.ORM

The Nike Air Force 1 PLT.AF.ORM elevates the iconic AF1 silhouette with bold design updates. Featuring a platform sole and premium materials, this women's shoe redefines style and comfort in a fresh, modern way.

₹ 8695

Category: Women's Shoes
In Stock: 18
Status: Just In

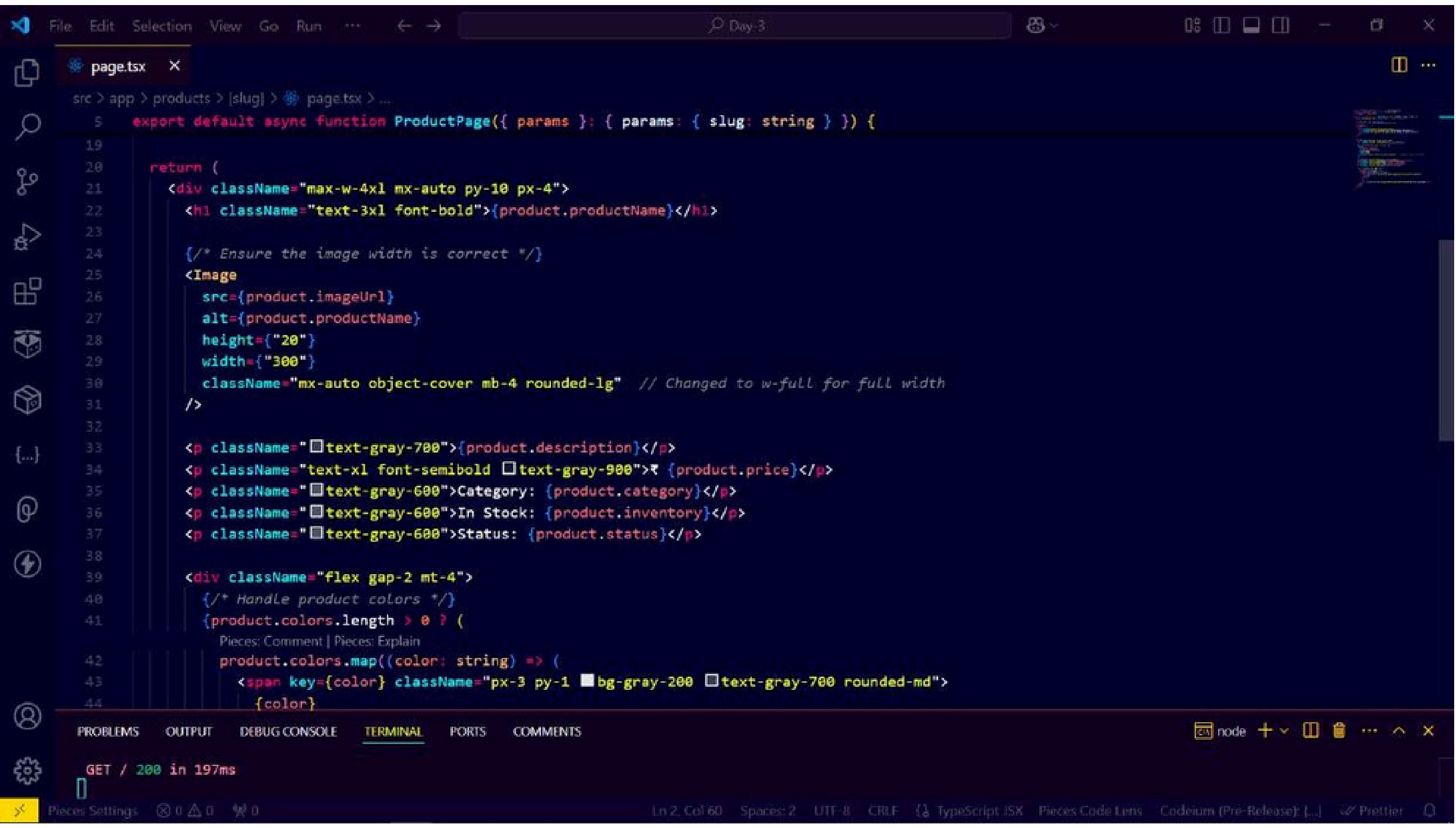


All Products

Nike Air Force 1 PLT.AF.ORM
₹ 8695
Category: Women's Shoes
In Stock: 18
Status: Just In

Nike Dri-FIT UV Hyverse
₹ 2495
Category: Men's Short-Sleeve Graphic Fitness Top
In Stock: 50
Status: Sustainable Materials

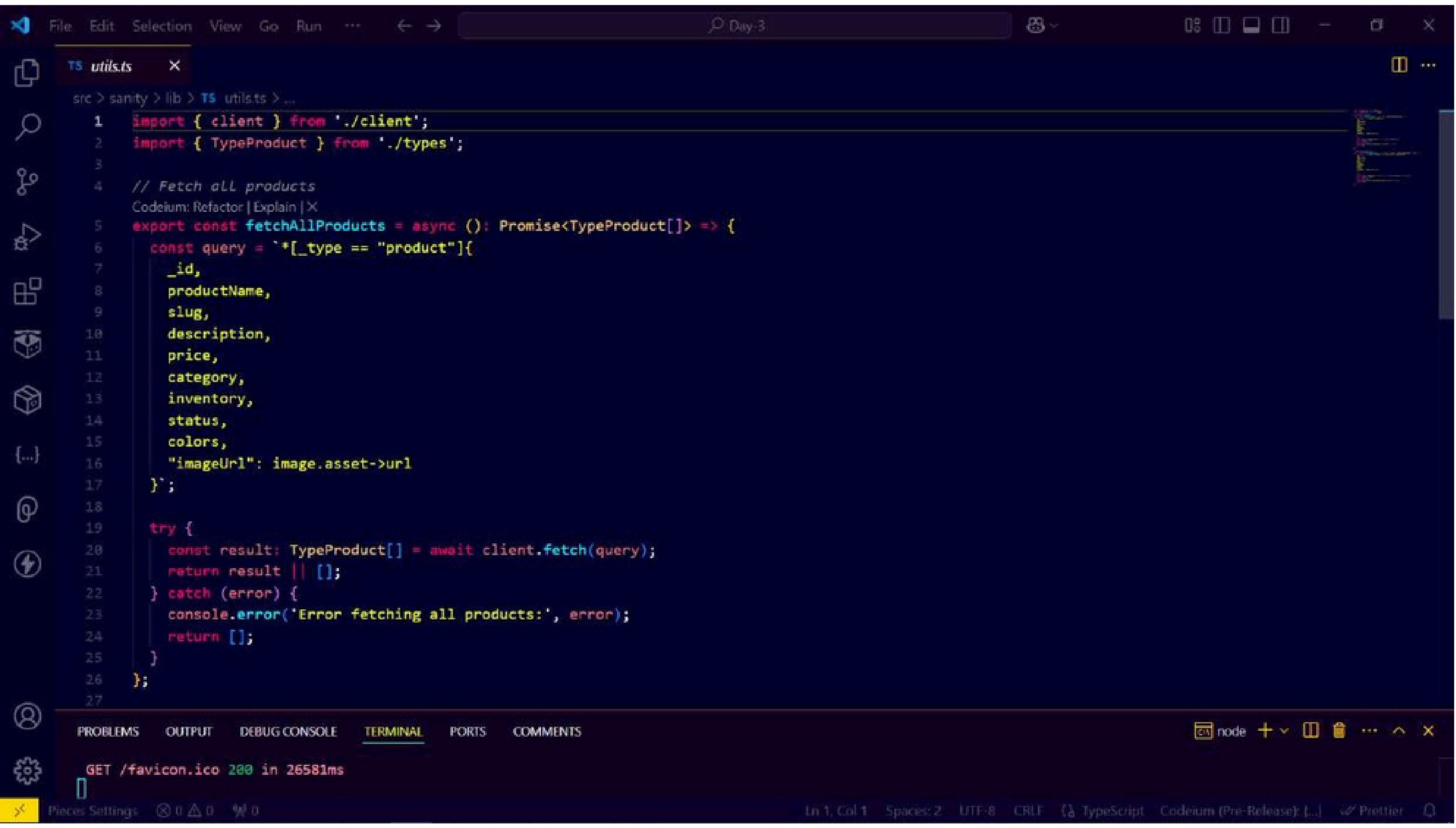
Nike Air Max 270
₹ 13295
Category: Men's Shoes
In Stock: 20
Status: Trending



File Edit Selection View Go Run ···
Day-3
page.tsx
src > app > products > [slug] > page.tsx > ...
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS
node
Pieces Settings
Ln 2, Col 60 Spaces: 2 UTF-8 CRLF TypeScript JSX Pieces Code Lens Codeium (Pre-Release): {...} Prettier

```tsx
export default async function ProductPage({ params }: { params: { slug: string } }) {

  return (
    <div className="max-w-4xl mx-auto py-10 px-4">
      <h1 className="text-3xl font-bold">{product.productName}</h1>

      {/* Ensure the image width is correct */}
      <Image
        src={product.imageUrl}
        alt={product.productName}
        height={"20"}
        width={"300"}
        className="mx-auto object-cover mb-4 rounded-lg"  // Changed to w-full for full width
      />

      <p className="text-gray-700">{product.description}</p>
      <p className="text-xl font-semibold text-gray-900">₹ {product.price}</p>
      <p className="text-gray-600">Category: {product.category}</p>
      <p className="text-gray-600">In Stock: {product.inventory}</p>
      <p className="text-gray-600">Status: {product.status}</p>

      <div className="flex gap-2 mt-4">
        {/* Handle product colors */}
        {product.colors.length > 0 ? (
        Pieces: Comment | Pieces: Explain
          product.colors.map((color: string) => (
            <span key={color} className="px-3 py-1 bg-gray-200 text-gray-700 rounded-md">
              {color}
```

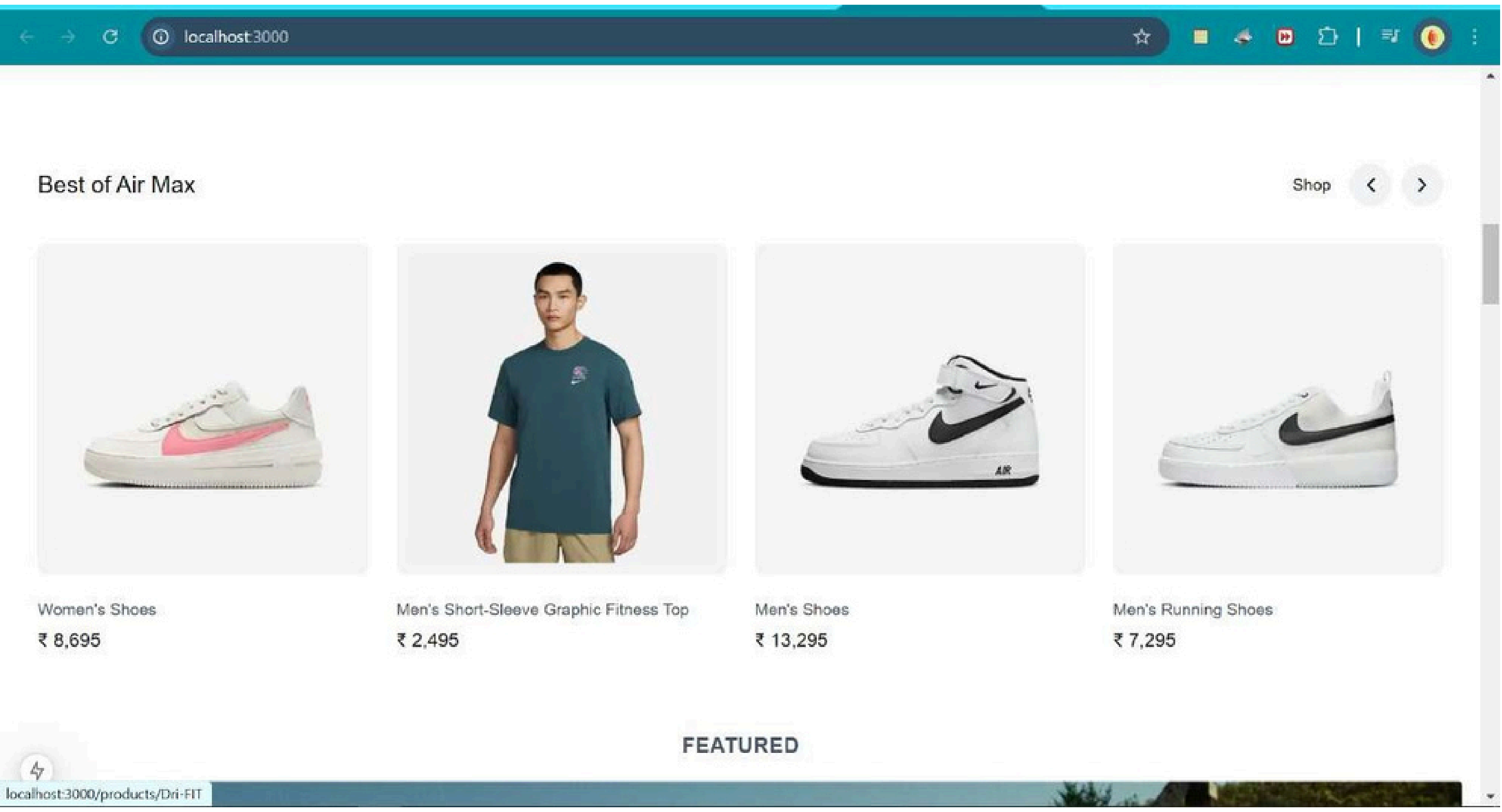
```
GET / 200 in 197ms
```



```typescript
import { client } from './client';
import { TypeProduct } from './types';

// Fetch all products
export const fetchAllProducts = async (): Promise<TypeProduct[]> => {
  const query = `*[_type == "product"]{
    _id,
    productName,
    slug,
    description,
    price,
    category,
    inventory,
    status,
    colors,
    "imageUrl": image.asset->url
  }`;

  try {
    const result: TypeProduct[] = await client.fetch(query);
    return result || [];
  } catch (error) {
    console.error('Error fetching all products:', error);
    return [];
  }
};
```

```
GET /favicon.ico 200 in 26581ms
```

File Edit Selection View Go Run ···
Day-3
utils.ts
src > sanity > lib > TS utils.ts > ...
Codeium: Refactor | Explain | ×
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS
node
Pieces Settings ⊗ 0 ⚠ 0 ⚲ 0
Ln 1, Col 1 Spaces: 2 UTF-8 CRLF {} TypeScript Codeium (Pre-Release): {...} ✓ Prettier



localhost:3000

Best of Air Max

Shop

Women's Shoes
₹ 8,695

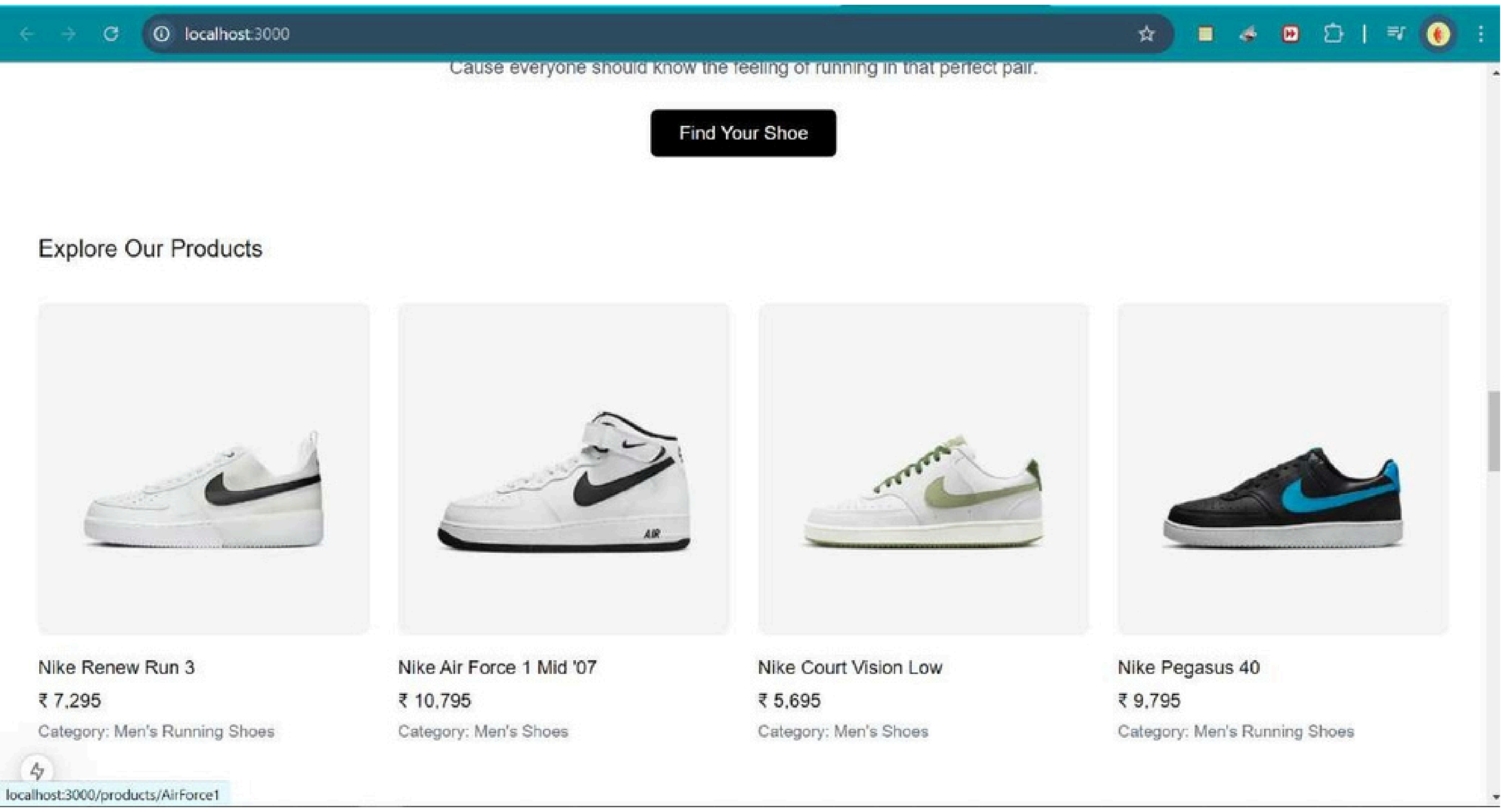
Men's Short-Sleeve Graphic Fitness Top
₹ 2,495

Men's Shoes
₹ 13,295

Men's Running Shoes
₹ 7,295

FEATURED

localhost:3000/products/Dri-FIT



Cause everyone should know the feeling of running in that perfect pair.

Find Your Shoe

Explore Our Products

Nike Renew Run 3
₹ 7,295
Category: Men's Running Shoes

Nike Air Force 1 Mid '07
₹ 10,795
Category: Men's Shoes

Nike Court Vision Low
₹ 5,695
Category: Men's Shoes

Nike Pegasus 40
₹ 9,795
Category: Men's Running Shoes