E-Commerce Website Assignment Report

1. API Integration Process

1.1. Overview

The e-commerce website integrates Sanity CMS for managing product data, including fields for name, price, description, category, and more. Products are displayed dynamically on the frontend using GROQ queries.

1.2. GROQ Query

```
*[_type == 'products'] {
  name,
  price,
  description,
  category,
  "imageUrl": image.asset->url
}
```

1.3. API Integration Code

Below is an example implementation in Next.js:

```
import { createClient } from '@sanity/client';

const client = createClient({
  projectId: 'yourProjectId',
  dataset: 'production',
  useCdn: true,
```

```
});


export async function getStaticProps() {
  const products = await client.fetch(`*[_type == 'products'] {
    name,
    price,
    description,
    category,
    "imageUrl": image.asset->url
  }`);

  return { props: { products } };
}
```

2. Adjustments to Schemas

2.1. Products Schema

The schema for products includes fields for:

- Name (string)
- Price (number)
- Description (text)
- Image (image)
- Category (string with predefined options)
- Discount Percent (number)
- Colors (array of strings)
- Sizes (array of strings)

```
export default defineType({
  name: 'products',
  title: 'Products',
  type: 'document',
  fields: [
    { name: 'name', title: 'Name', type: 'string' },
    { name: 'price', title: 'Price', type: 'number' },
    { name: 'description', title: 'Description', type: 'text' },
    { name: 'image', title: 'Image', type: 'image' },
    {
      name: 'category',
      title: 'Category',
      type: 'string',
      options: {
        list: [
          { title: 'T-Shirt', value: 'tshirt' },
          { title: 'Short', value: 'short' },
          { title: 'Jeans', value: 'jeans' },
          { title: 'Hoodie', value: 'hoodie' },
          { title: 'Shirt', value: 'shirt' },
        ],
      },
    },
    { name: 'discountPercent', title: 'Discount Percent', type: 'number' },
    { name: 'new', title: 'New', type: 'boolean' },
    { name: 'colors', title: 'Colors', type: 'array', of: [{ type: 'string' }] },
    { name: 'sizes', title: 'Sizes', type: 'array', of: [{ type: 'string' }] },
```

```
  ],
});
```

3. Migration Steps and Tools

3.1. Tools Used

- Sanity CLI: For schema creation and deployment.

- @sanity/client: For data fetching in the frontend.

3.2. Migration Steps

1. Defined the `products` schema in Sanity CMS.

2. Populated the data manually via the Sanity Studio interface.