Bing

Astro content collections are a way to organize and query your Markdown, MDX, JSON, and YAML files in the `src/content` folder. You can use Astro content collections for providing products in your store by following these steps:

- First, you need to define a collection schema for your products in the `src/content/config.ts` file using Zod, a TypeScript-friendly validation library. You can use `defineCollection` from `astro:content` to create a collection with a name and a schema. For example, you can create a collection of products with the following schema:

```
import { z, defineCollection } from "astro:content";

const productCollection = defineCollection({
  name: "products",
  schema: z.object({
    name: z.string(),
    description: z.string(),
    price: z.number(),
    image: z.string().optional(),
    category: z.string(),
  }),
});

export const collections = {
  products: productCollection,
};
```

- Next, you need to create JSON files in the `src/content/products` folder that match the schema. Each JSON file represents an entry in the collection. For example, you can create a file named `book.json` with the following content:

```
{
  "name": "The Hitchhiker's Guide to the Galaxy",
  "description": "A classic science fiction comedy novel by Douglas Adams.",
  "price": 9.99,
  "image": "/images/book.jpg",
  "category": "Books"
}
```

- Then, you can query the collection using Astro's built-in Content APIs. You can use `Astro.fetchContent` to get all the entries in the collection, or use `Astro.request.params.slug` to get a specific entry by its filename. For example, you can create an Astro component named `ProductCard.astro` that displays the product's information:

```
---
import { Astro } from "astro";

// Get the product by slug
const product = Astro.fetchContent("../content/products/[slug].json");
---

<div class="product-card">
  {product.image && <img src={product.image} alt={product.name} />}
  <h3>{product.name}</h3>
  <p>{product.description}</p>
  <p>${product.price}</p>
</div>
```

- Finally, you can use the collection entries in other collections or Astro components. You can reference an entry by its filename using the `$ref` keyword. For example, you can create a collection of reviews that reference a product entry in their frontmatter:

```
---
title: "A hilarious and witty adventure"
rating: 5
product: $ref:../products/book.json
---

# A hilarious and witty adventure

This book is one of my favorites. It's full of humor, imagination, and satire. I highly recommend it to anyone who loves science fiction and comedy.
```

You can access the referenced data using the `Astro.fetchContent` API as well. For example, you can create an Astro component named `ReviewCard.astro` that displays the review title, rating, and product name:

```
---
import { Astro } from "astro";

// Get the review by slug
const review = Astro.fetchContent("../content/reviews/[slug].md");
---

<div class="review-card">
  <h3>{review.title}</h3>
  <p>Rating: {review.rating}/5</p>
  <p>Product: {review.product.name}</p>
</div>
```

For more information about Astro content collections, you can check out the official documentation or this tutorial. I hope this helps you use Astro content collections for providing products in your store. 😊