We will create a custom pipe.

But first add some built-in pipes to your templates. For example, Book Inventory could use the built-in currency pipe (you might want to change the numeric values in the ,ts file), uppercase pipe and date pipe (set `new Date()` in the .ts)

To create the custom pipe, we will follow these steps:

- **Custom Pipe:**
  Create `BookFilterPipe` that filters books based on a search term matching the title, author, or ISBN.
- **Component (TS):**
  - Import `BookFilterPipe` and `FormsModule` in your BookInventory component.
  - Add a `searchTerm` property to hold the user input.
  - Include the custom pipe in the component's `imports` array.
- **Template (HTML):**
  - Add an input bound to `searchTerm` with two-way binding.
  - Chain the pipe in your `*ngFor` loop to filter the inventory.

1. Create the Book Filter Custom Pipe

File Location - this pipe is available to all components: `src/app/book-filter.pipe.ts`

Steps - note that using the Angular CLI takes care of some of these steps for you:

- **a. Import Required Items:**
  Import `Pipe` and `PipeTransform` from `@angular/core` and import your `Book` interface/model (assumed to be in `src/app/book.ts`).
- **b. Decorate Your Pipe:**
  Use the `@Pipe` decorator with a name such as `"bookFilter"`. This name is what you will use in your template.
- **c. Implement the transform() Method:**
  In the method, check if a search term exists. If not, return the original list. Otherwise, convert the search term and each book's title, author, and ISBN to

lowercase and filter the list to include books where any field includes the search term.

Example Code:

```
// src/app/pipes/book-filter.pipe.ts
import { Pipe, PipeTransform } from '@angular/core';
import { Book } from '../book';

@Pipe({
  name: 'bookFilter'
})
export class BookFilterPipe implements PipeTransform {
  transform(books: Book[], searchTerm: string): Book[] {
    if (!books || !searchTerm) {
      return books;
    }
    searchTerm = searchTerm.toLowerCase();
    return books.filter(book =>
      book.title.toLowerCase().includes(searchTerm) ||
      book.author.toLowerCase().includes(searchTerm) ||
      book.ISBN.toLowerCase().includes(searchTerm)
    );
  }
}
```

2. Update the Book Inventory Component (TypeScript)

File Location: `src/app/book-inventory/book-inventory.component.ts`

Steps:

- **a. Import the Pipe and FormsModule:**
  Since you'll use the filter pipe in the template and also need two-way binding for the search input, import your `BookFilterPipe` and `FormsModule`.

- **b. Add a `searchTerm` Property:**
  Declare a property (for example, `searchTerm: string = ''`) to hold the user's search input.

- **c. Add the Pipe to the Component's Imports:**
  With Angular 17 standalone components, include the custom pipe and FormsModule in the `imports` array so that Angular knows about them.

Example Code:

```
// src/app/book-inventory/book-inventory.component.ts
import { Component, OnInit } from '@angular/core';
import { Book } from '../book';
import { CommonModule } from '@angular/common';
import { FormsModule } from '@angular/forms';
import { HoverHighlightDirective } from './hover-
highlight.directive';
import { BookFilterPipe } from '../pipes/book-filter.pipe';

@Component({
  selector: 'app-book-inventory',
  standalone: true,
  imports: [CommonModule, HoverHighlightDirective,
FormsModule, BookFilterPipe],
  templateUrl: './book-inventory.component.html',
  styleUrls: ['./book-inventory.component.css']
})
export class BookInventoryComponent implements OnInit {
  // Hard-coded list of books.
  inventory: Book[] = [
    {
      ISBN: "978-1492056205",
      title: "Angular Up & Running",
      author: "Shyam Seshadri",
      year: 2018,
      price: 39.99,
      featured: true,
      coverImages: ["/assets/angular-up-and-running.png"]
    },
    {
      ISBN: "978-1593279509",
      title: "Eloquent JavaScript, 3rd Edition",
      author: "Marijn Haverbeke",
      year: 2018,
      price: 29.99,
      featured: false,
      coverImages: ["/assets/eloquent-javascript.jpg"]
    },
    {
      ISBN: "978-1491904244",
      title: "You Don't Know JS Yet: Get Started",
      author: "Kyle Simpson",
      year: 2020,
      price: 34.99,
      featured: false,
      coverImages: ["/assets/ydkjs-cover.jpg"]
    },
    {
      ISBN: "978-1449331818",
      title: "Learning JavaScript Design Patterns",
      author: "Addy Osmani",
      year: 2012,
      price: 25.99,
      featured: true,
      coverImages: ["/assets/js-design-patterns.png"]
    }
  ];

  // Property to store the user's search term.
  searchTerm: string = '';

  ngOnInit(): void {
    // Any initialization logic if needed.
  }

  // Use trackBy to help Angular uniquely identify each
book (using ISBN).
  trackByISBN(index: number, book: Book): string {
    return book.ISBN;
  }

  // Delete a book from the inventory.
  deleteBook(book: Book): void {
    this.inventory = this.inventory.filter(b => b.ISBN !==
book.ISBN);
  }
}
```

3. Update the Book Inventory Template (HTML)

File Location: `src/app/book-inventory/book-inventory.component.html`

Steps:

- **a. Add a Search Input:**
  At the top of your template, add an input box bound to the `searchTerm` property using two-way binding (`[(ngModel)]`).

- **b. Apply the Filter Pipe in the ngFor Loop:**
  Pipe your inventory array through the custom `bookFilter` pipe with the search term as its argument. This will filter the list dynamically.

- **c. Maintain the Existing Structure:**
  Continue rendering the book cards as you did previously.

Example Code:

```
<div class="container my-4">
  <h1 class="mb-4 text-center">Current Inventory</h1>

  <!-- Search input for filtering books -->
  <div class="mb-3 text-center">
    <input type="text" [(ngModel)]="searchTerm"
placeholder="Search books" class="form-control w-50 mx-
auto" />
  </div>

  <p *ngIf="inventory.length === 0" class="alert alert-info
text-center">
    There are no books in inventory.
  </p>

  <div class="row">
    <!-- Apply the bookFilter pipe with searchTerm -->
    <div class="col-12 col-sm-6 col-md-4 col-lg-3 mb-4"
         *ngFor="let book of (inventory |
bookFilter:searchTerm); trackBy: trackByISBN">
      <div class="card h-100 shadow-sm"
appHoverHighlight="#e1e1e1">
        <div class="ratio ratio-4x3">
          <img *ngIf="book.coverImages.length > 0"
               [src]="book.coverImages[0]"
               class="card-img-top"
               alt="{{ book.title }} cover"
               style="object-fit: cover;">
        </div>
        <div class="card-body d-flex flex-column">
          <h5 class="card-title">{{ book.title }}</h5>
          <p class="card-text flex-grow-1">
            <strong>Author:</strong> {{ book.author }}<br>
            <strong>Year:</strong> {{ book.year }}<br>
            <strong>ISBN:</strong> {{ book.ISBN }}<br>
            <strong>Price:</strong> {{ book.price |
currency:'USD':'symbol' }}
          </p>
          <p *ngIf="book.featured" class="badge bg-primary
mb-2">Featured Book</p>
          <div [ngSwitch]="book.price > 30 ? 'expensive' :
'affordable'" class="mb-2">
            <p *ngSwitchCase="'expensive'" class="text-
danger mb-0">Premium selection!</p>
            <p *ngSwitchCase="'affordable'" class="text-
success mb-0">Budget-friendly!</p>
            <p *ngSwitchDefault class="text-muted
mb-0">Great value!</p>
          </div>
          <button (click)="deleteBook(book)" class="btn
btn-danger mt-auto">Delete</button>
        </div>
      </div>
    </div>
  </div>
</div>
```