Aufgabe

Ziel: In dieser Aufgabe erstellst du eine einfache Shop-Anwendung mit Angular, die Produktdaten aus einer JSON-Datei lädt und in mehreren Komponenten darstellt.

Der Fokus liegt auf:

- Komponenten erstellen
- Arbeiten mit Daten
- Datenübergabe mit @Input()
- Anzeige von Produktübersicht

Situation: Du arbeitest an einer Angular-Anwendung. Für einen Mini-Shop sollen Produkte angezeigt werden, die in einem service gespeichert sind.

Die Anwendung besteht aus mehreren Komponenten, die gemeinsam die Benutzeroberfläche aufbauen. Jede Komponente darf eigenständig gestylt werden (z. B. über die jeweilige CSS-Datei).

Komponentenübersicht

Du arbeitest mit folgenden Komponenten:

1. **HeaderComponent**
2. **ProductListComponent**
3. **ProductCardComponent**
4. **ProductDetailsComponent (Zusatzaufgabe)**

1. Projekt vorbereiten

- Erstelle ein neues Angular-Projekt mit dem Namen “minishop” oder verwende ein bestehendes
- Starte die Anwendung im Browser und überprüfe, ob sie korrekt läuft

2. Komponenten erstellen

Erstelle mit der Angular CLI folgende Komponenten:

- header
- product-list
- product-card

3. Service erstellen und bereitgestellte JSON als produkte anlegen

- Erstelle einen service: ng generate service products
- Lege ein property namens *products* an das alle produkte enthält
- Erstelle eine methode die alle produkte zurückgibt

4. HeaderComponent (Teil der Hauptaufgabe)

- Zeige im Header:
  - den Namen des Shops (z. B. „Mini Shop“)
- Der Header wird einmal oben in der Anwendung angezeigt

5. ProductListComponent

- Lade die Produktdaten aus der service
- Speichere die Daten in einer Variable
- Zeige für jedes Produkt eine **ProductCardComponent** an
- Übergib das jeweilige Produkt an die ProductCardComponent

6. ProductCardComponent

- Zeige Grundinformationen eines Produkts an:
  - Name
  - Preis
  - Beschreibung

  Zusatz, siehe zusatzaufgabe

- Binde eine **ProductDetailsComponent** in das Template ein
- Übergib das Produkt an die ProductDetailsComponent

Styling-Hinweis

- Jede Komponente darf individuell gestylt werden
- Verwende dafür die jeweilige CSS-Datei der Komponente
- Das Styling ist frei wählbar und dient der besseren Übersicht

---

Zusatzaufgabe:

1. Erstelle eine neue Komponente: ProductDetailsComponent

2. ProductDetailsComponent
   zeigt zusätzliche Produktinformationen an:
   - Verfügbarkeit
   - usw.

3. Einbindung der Detail-Komponente
   - Die ProductDetailsComponent wird im Template der ProductCardComponent eingebunden
   - Die Produktdaten werden ausschließlich über @Input() an die ProductDetailsComponent übergeben

4. Anzeige der Details
   - Die Produktdetails werden immer oder optional nach einem Klick angezeigt
   - Falls optional:
     - Die Anzeige erfolgt über ein (click)-Event in der ProductCardComponent
     - Eine Boolean-Variable steuert die Sichtbarkeit
     - Die Sichtbarkeit wird mit *ngIf umgesetzt
