Ionic Shop
mit Ionic 8 standalon und Angular 17

Erstelle eine neue Seite im Ordner „pages“ namens „shop“

```
PS C:\ionic_start\ersteapp> ionic g page pages/shop
```

Öffne die „shop.html“ und entferne vieles, damit das überbleibt:

shop.page.html U

src > app > pages > shop > shop.page.html > ion-header

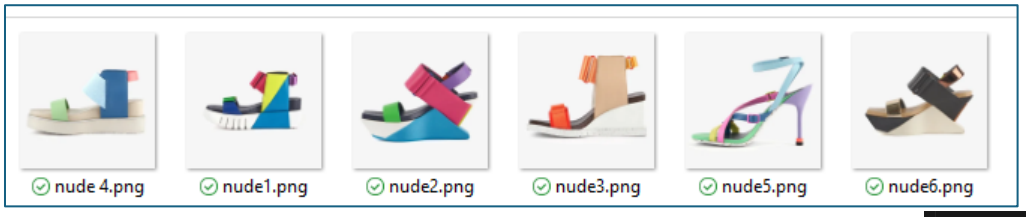
Go to component

```
<ion-header>
  <ion-toolbar>
    <ion-title>Shop</ion-title>
  </ion-toolbar>
</ion-header>

<ion-content >

</ion-content>
```

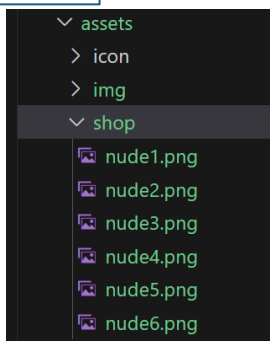
1)Daten im Shop anlegen

a)Kopiere die Bilder in den Ordner „assets“ in einen Ordner „shop“

--- > sorry doch lieber Schuhe 😊



360 x 360 Pixel



Folgende Modelle der Marke „united nude“:
https://unitednude.com/ https://unitednude.at/
Nude1 delta run, Keilsandalette, rainbow, 250,00
Nude2 delta, Plateausandalette, rainbow 280,00
Nude3 Ko Mid, Keilsandalette, malibu 200,00
Nude4 rico, Plateausandalette, summer 240,00
Nude5 strappy hi, Hi Heel Sandalette, 180,00
Nude6 delta, Plateausandalette, bronze 280,00

Code ist unterhalb zum Kopieren.

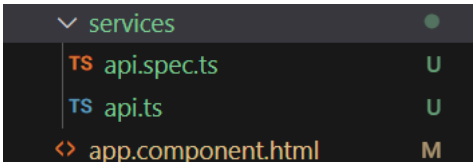
Service erstellen

Um die Daten für den Shop anzeigen lassen zu können, soll nun ein „Service“ erstellt werden.

Dazu soll mit Hilfe des Terminals die Datei „api“ im Unterfolder von den „services“ erstellt werden.

```
eapp> ionic g service services/api
```

Ergebnis:



Erstelle für alle Einträge (Produkte) im Shop ein Array: mit dem Namen „item“ – Strichpunkt am Ende:

items: any[] = [

{}

];

```
6   export class Api {
7     items: any[] = [
8       {}
9     ];
10
11  }
```

Innerhalb der geschwungenen Klammern sollen nun alle 6 Artikel angeführt werden. Hier ist das erste Element dargestellt, darunter als Kopiervorlage der Rest:

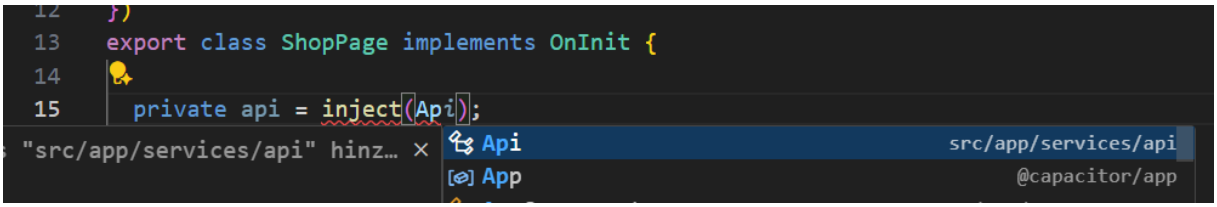
```
8   items: any[] = [
9     {
10      id: '1',
11      name: 'Delta run',
12      preis: 250,
13      rating: 4.5,
14      cover: 'assets/shop/nude1.png',
15      beschreibung: 'Keilsandalette, rainbow'
16    },
17    {
18      id: '2',
```

```
items: any[] = [
 {
   id: '1',
   name: 'Delta run',
   preis: 250,
   rating: 4.5,
   cover: 'assets/shop/nude1.png',
   beschreibung: 'Keilsandalette, rainbow'
 },
 {
```

```
  id: '2',
  name: 'Delta',
  preis: 280,
  rating: 4.7,
  cover: 'assets/shop/nude2.png',
  beschreibung: 'Plateausandalette, rainbow'
 },
 {
  id: '3',
  name: 'Ko Mid',
  preis: 200,
  rating: 4.2,
  cover: 'assets/shop/nude3.png',
  beschreibung: 'Keilsandalette, malibu'
 },
 {
  id: '4',
  name: 'Rico',
  preis: 240,
  rating: 4.8,
  cover: 'assets/shop/nude4.png',
  beschreibung: 'Plateausandalette, summer'
 },
 {
  id: '5',
  name: 'Strappy Hi',
  preis: 180,
  rating: 4.5,
  cover: 'assets/shop/nude5.png',
  beschreibung: 'Hi Heel Sandalette'
 },
 {
  id: '6',
  name: 'Delta',
  preis: 280,
  rating: 4.6,
  cover: 'assets/shop/nude6.png',
  beschreibung: 'Plateausandalette, bronze'
 }
```

2)Daten in shop.page.ts in eine Funktion einbauen

In der Export-Klasse füge mittels „inject“ das eben angelegte API-Service ein und beachte den gleichzeitigen IMPORT durch Anklick des Vorschlages:



Dadurch wird der Import eingefügt, der unbedingt nötig ist: siehe Zeile 2
auch in Zeile 1: inject - beachten

```
TS api.ts  U    TS shop.page.ts 1, U
1   import { Component, OnInit } from '@angular/core';
2   import { CommonModule } from '@angular/common';
3   import { FormsModule } from '@angular/forms';
4   import { IonContent, IonHeader, IonTitle, IonToolba
5   import { Api } from 'src/app/services/api';
6
7   @Component({
```

```
14  export class ShopPage implements OnInit {
15
16    private api = inject(Api);
17
18    constructor() { }
```

Funktion erstellen

In die „ngOninit-Funktion“ schreibe

this.getItems();

```
18    constructor() { }
19
20    ngOnInit() {
21      this.getItems();
22    }
23
24  }
```

Erstelle danach eine neue Funktion, die die Items aufrufen wird: dann ist auch die rote Fehlerinfo verschwunden.

```
18    constructor() { }
19
20    ngOnInit() {
21      this.getItems();
22    }
23
24    getItems(){}
25  }
```

Darin werden „allItems“ mit den Ergebnissen aus dem API „angefüttert“.

this.allItems = this.api.items;
this.items = [...this.allItems];

```
24    getItems(){
25      this.allItems = this.api.items;
26      this.items = [...this.allItems];
27
28    }
```

Dafür aber in der Klasse oben müssen diese Variablen angelegt werden:

Items vom Typ „any“ und initialisiert als „leeres Array“, genauso die Variable „allItems“

items: any[] = [];

allItems: any[] = [];

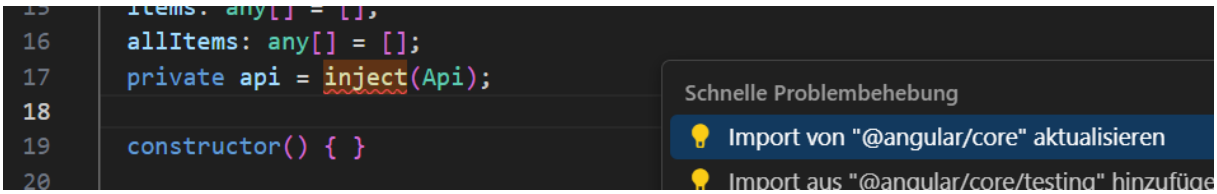
```
14  export class ShopPage implements OnInit {
15    items: any[] = [];
16    allItems: any[] = [];
17    private api = inject(Api);
18
19    constructor() { }
20
21    ngOnInit() {
22      this.getItems();
23    }
24
25    getItems(){
26      this.allItems = this.api.items;
27      this.items = [...this.allItems];
28
29    }
```

Fehlerinfo:

```
14  export class ShopPage implements OnInit {
15    items: any[] = [];
16    allItems: any[] = [];
17    private api = inject(Api);
18
```

Hoover mit der Maus darüber und übernimm die „Schnell Problembehebung“

Wähle das oberste der Lösungen:



3)Anzeigen in der „shop.page.html“:

Dafür lege im Content typische Elemente an wie

- ion-row
- ion-col

In jeder „col“ soll dann ein Artikel mit Bild und Beschreibung angezeigt werden. Es wird auch eine Variation eingebaut werden, die auf die Bildschirmgröße Rücksicht nehmen wird.

```
<ion-content >
  <ion-row>
    <ion-col>

    </ion-col>
  </ion-row>
</ion-content>
```

Um durch alle Einträge durchzugehen (looping) benutze eine Schleife. Die Ordnung (track) könnte auch nach „item.id“ erfolgen, hier aber etwas großzügiger durch „i“, welche nach einem Index erfolgt.

In der row, also vor der col:

@for(item of items; track i; let i = $index){}

```
<ion-content >
  <ion-row>
    @for(item of items; track i; let i = $index){}
    <ion-col>
```

ABER: in die Funktion muss jede „col“ hinein, daher muss die schließende geschwungenen Klammer nach dem Ende der <ion-col> gezogen werden!!!

```
<ion-content >
  <ion-row>
    @for(item of items; track i; let i = $index){
    <ion-col>
    </ion-col>
  }   <!-- hier schließende Klammer-->
  </ion-row>
</ion-content>
```

Innerhalb der „col“ dann „thumbnail“:

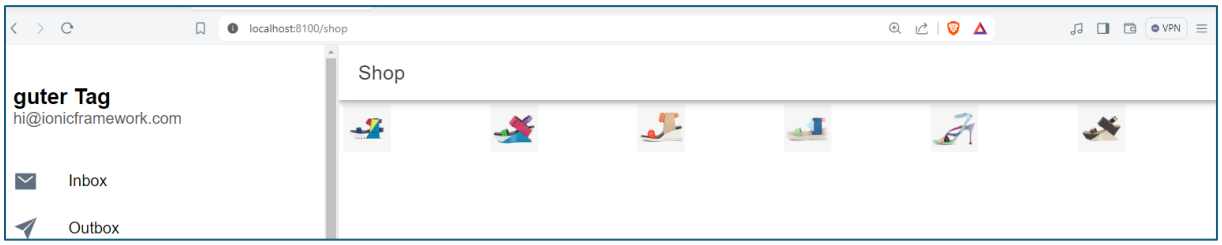
```
    <ion-col>
      <ion-thumbnail></ion-thumbnail>
    </ion-col>
  </ion-row>
```

In das „thumbnail“ (Fingernagel = Mini-Bild) wird ein Bild gestellt:

```
10      <ion-col>
11        <ion-thumbnail>
12          <ion-img [src]="item?.cover"></ion-img>
13        </ion-thumbnail>
14      </ion-col>
15    }  <!-- hier schließende Klammer-->
```

Ergebnis:
die Bilder werden schon angezeigt – bei Eingabe „shop“ in der URL

http://localhost:8100/shop



Verbesserung: Erstellen von „cards“

Diese Cards bieten die schöne Möglichkeit von einzelnen Rahmen, die dann später pro Shop-Item mit Bild und Beschreibung und entsprechender Hintergrundfarbe eine gute Abgrenzung bieten werden.

```
10      <ion-col>
11        <ion-card>
12        <ion-thumbnail>
13          <ion-img [src]="item?.cover"></ion-img>
14        </ion-thumbnail>
15      </ion-card>
16      </ion-col>
17    }  <!-- hier schließende Klammer-->
```

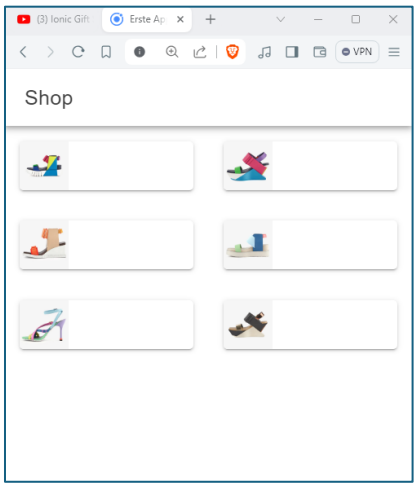
Verbesserung durch die Anpassung an die Bildschirmgrößen

Bei einer Anzahl von 12 Spalten pro Reihe ergibt sich z.B. bei einem „sizeXS“ für ein Smartphone eine Aufteilung bei 6 eine Anzeige von 2 Bildern pro Zeile, was sinnvoll ist. Die Berechnung erfolgt nämlich: 12 / 6 = 2 pro Zeile

```
 9      @for(item of items; track i; let i = $index){
10      <ion-col sizeLg="3" sizeMd="4" sizeSm="6" sizeXL="3" sizeXs="6">
11        <!--Aufteilung bei 12 Spalten pro Reihe-->
12        <ion-card>
```

<ion-col sizeLg="3" sizeMd="4" sizeSm="6" sizeXL="3" sizeXs="6">
<!--Aufteilung bei 12 Spalten pro Reihe-->

Ergebnis bei XS:

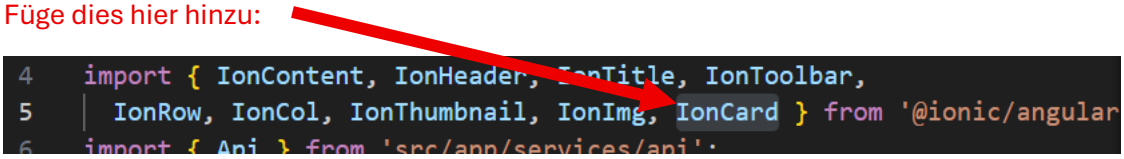


Card – Problem:

Dann tun wir das:

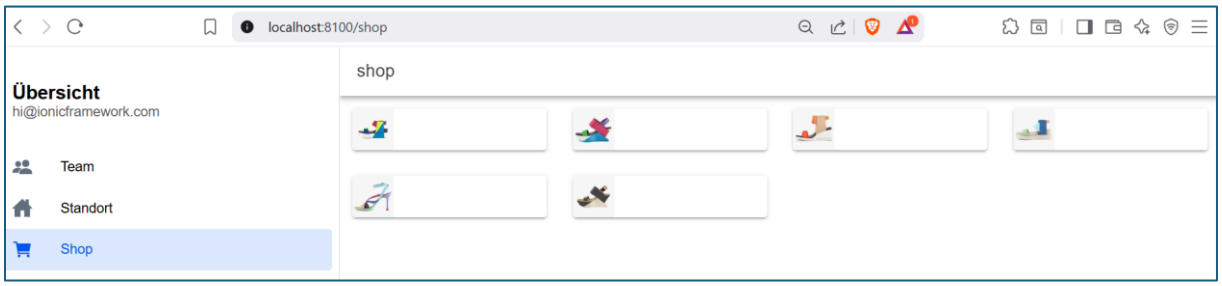
```
[ng] Application bundle generation failed. [3.939 seconds] - 2025-10-12T18:41:32.179Z
[ng] X [ERROR] NG8001: 'ion-card' is not a known element:
[ng] 1. If 'ion-card' is an Angular component, then verify that it is included in the '@Component.imports' of this co
mponent.
```

Füge dies hier hinzu:



Ergebnis

Nachdem im Sidemenü auch Änderungen vorgenommen wurden:



Sidemenü ändern in der „app.components.ts“:

```
public appPages = [
  { title: 'Team', url: '/team', icon: 'people' },
  { title: 'Standort', url: '/standort', icon: 'home' },
  { title: 'Shop', url: '/shop', icon: 'shopping-cart' },
];
// public labels = ['Family', 'Friends', 'Notes', 'Work', 'T
```

Beschreibung bei den Bildern anzeigen

Zurück in die „shop.page.html“:

Erstelle innerhalb der Card aber unter dem Thumbnail ein <ion-label> für den Textbereich.

```
<ion-label>
    <ion-text color="dark"><strong>
     {{item?.name}}
    </strong></ion-text>
   </ion-label>
```

```
      </ion-thumbnail>
      <ion-label>
        <ion-text color="dark"><strong>
          {{item?.name}}
        </strong></ion-text>
      </ion-label>
    </ion-card>
    </ion-col>
```

Darunter, aber noch innerhalb des labels, der Preis:

```
  <p>
    <ion-text color="dark">
     <strong>
      {{item?.preis}}
     </strong>
    </ion-text>
   </p>
```

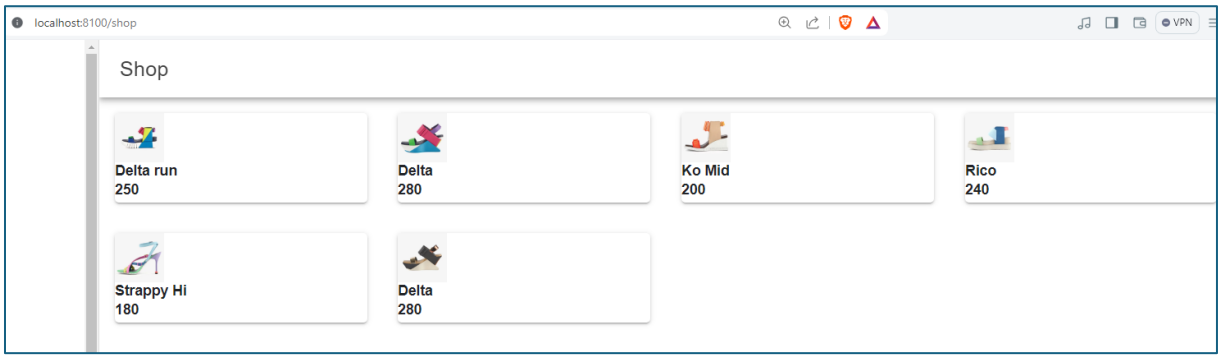
```
      </strong></ion-text>
      <p>
        <ion-text color="dark">
          <strong>
            {{item?.preis}}
          </strong>
        </ion-text>
      </p>

    </ion-label>
```

Lösung für fehlenden IMPORT:

```
17
18        <ion-label>
19          <ion-text color="dark"><strong>
20            {{item?.name}}
21          </strong></ion-text>
22                <p>
```

Kicke auf „ion-text“ und es wird sich in der dazu passenden „.ts“-Datei der Import automatisch erstellen. Die .ts dann speichern.

Ergebnis:



Innerhalb vom <p> wird nun das Rating angezeigt, das später rechtsbündig gemacht wird:

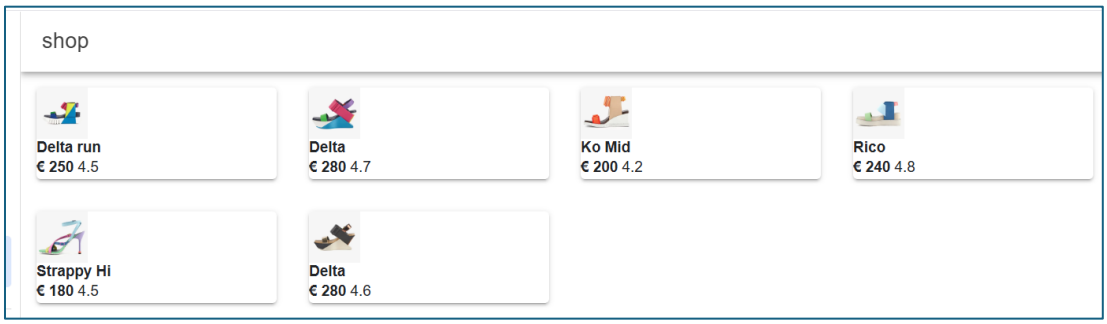
<ion-text class="rating" color="dark">
{{item?.rating}}
</ion-text>

```
24              </strong>
25            </ion-text>
26
27            <ion-text class="rating" color="dark">
28          {{item?.rating}}
29          </ion-text>
30          </p>
31      </ion-label>
32      </ion-card>
```

Beim Preis füge ein Eurozeichen hinzu:

```
21              <ion-text color="dark">
22                <strong>
23                € {{item?.preis}}
24                </strong>
```

Ergebnis:

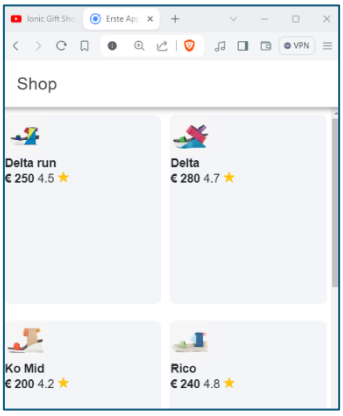


Design per CSS

Öffne die „shop.page.scss“.

Erstelle die ion-card und diese wird nun designed.

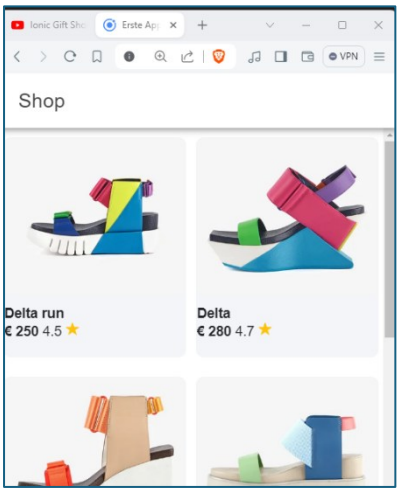

```
ion-card {
  --background: var(--ion-color-light);
  height: 14rem;
  margin: 0.3rem 0;  // top und bottom, null links und rechts
  border-radius: 8px;
  box-shadow: none;
  border: 1px solid var(--ion-color-light);
}
```



Verbesserung, da Bild zu klein:

Innerhalb der Klasse, erstelle:

```
  border: 1px solid var(--ion-color

  ion-thumbnail {
    width: 100%;
    height: 10rem;
    margin-bottom: 10px;
    ion-img {
      height: 100%;
      border-radius: 8px;
    }
  }
}
```




```
ion-thumbnail {
  width: 100%;
  height: 10rem;
  margin-bottom: 10px;
  ion-img {
    height: 100%;
    border-radius: 8px;
  }
}
```

Design vom Label – alles ebenfalls noch in der „card“

Innerhalb der thumbnail: Die font-size und auch den Abstand der beiden Zeile, daher „label UND p“ darunter mit einem Abstand von links und rechts

```

  ion-label {
    font-size: 0.9rem;
  }
  ion-label, p {
    margin: 0 10px;
  }
}
```

Die Zeile von „p“ ebenfalls extra designen inkl. „rating und Stern“

Außerhalb vom thumbnail:

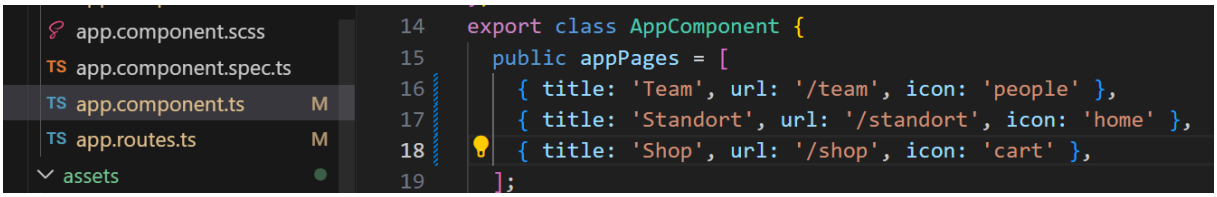
Stern und rating soll rechtsbündig sein

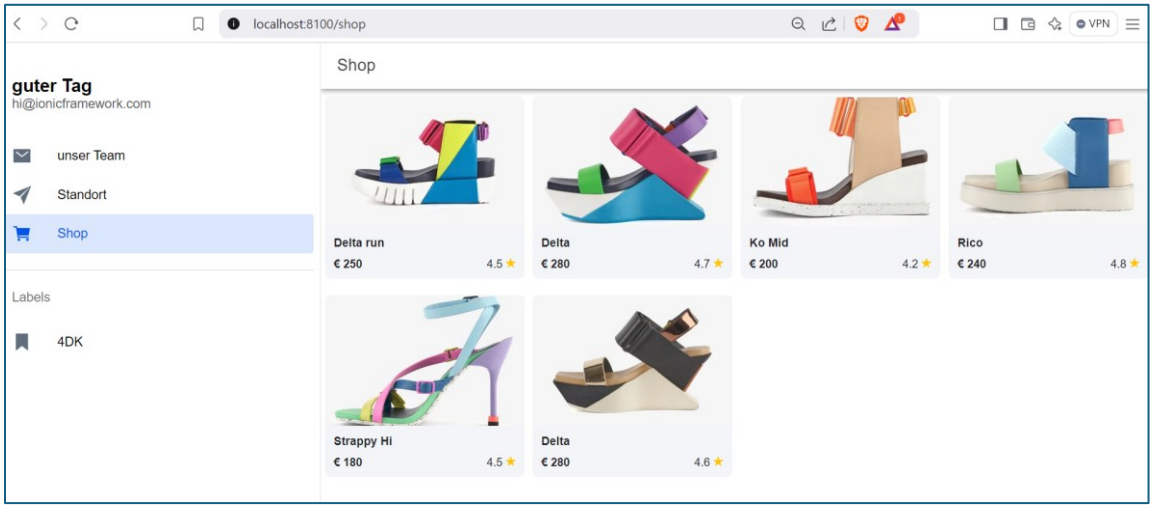
```
25
26    p {
27      font-size: 0.85;
28      margin-top: 8px;
29      font-weight: 500;
30      ion-text.rating {
31        float: right;
32        ion-icon {
33          font-size: 0.8rem;  //damit der Stern kleiner ist
34        }
35      }
36    }
```

```
p {
 font-size: 0.85;
 margin-top: 8px;
 font-weight: 500;
 ion-text.rating {
  float: right;
  ion-icon {
   font-size: 0.8rem;  //damit der Stern kleiner ist
  }
 }
```

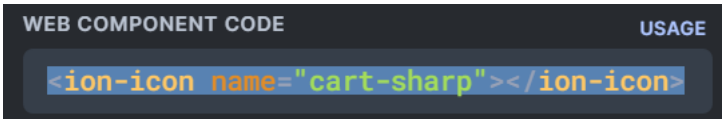
In die Side-Navigation aufnehmen

Öffne die „app.component.ts“ und erstelle einen neuen Eintrag





Als ICON nimm folgende Bezeichnung auf:



Diese muss ebenfalls 2 x aufgenommen werden in:

```
import { addIcons } from 'ionicons';
import { cartSharp, peopleSharp, homeSharp, mailOutl
```

```
constructor() {
  addIcons({cartSharp, peopleSharp, homeShar
}
```

Searchbar

folgt

Shop

https://www.youtube.com/watch?v=tx0ZM6UA464

https://www.youtube.com/watch?v=BHM7_NmX_LA