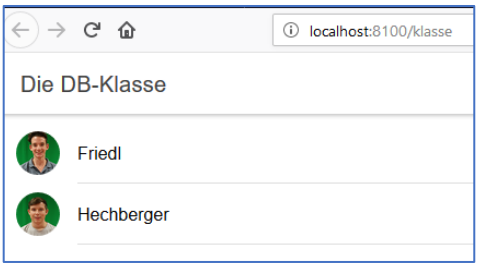
Ionic 4 – Liste mit Bildern anlegen

Ziel:



Erstes Beispiel

1)Projekt im Terminal anlegen: Öffne VS-Code und das Terminal. Befehle sind z.B.:

- dir zeigt den Inhalt eines Ordners an
- mkdir erstellte einen neuen Ordner
- cd wechselt in einen Ordner, z.B. cd desktop (change directory)
- cd .. geht einen Pfad zurück

```
PS C:\Users\edi\Desktop\ionic\erstesapp> cd..
PS C:\Users\edi\Desktop\ionic> cd..
PS C:\Users\edi\Desktop>
```

Hier wurde der Pfad zurück zum Desktop mittels 2 Befehlen hergestellt (zweimal cd..). Dann kann man ein neues Projekt erstellen:

Erstelle auf dem Desktop einen neuen Ordner:

- mkdir ionicangular1

```
PS C:\Users\edi\Desktop> mkdir ionicangular1
```

Man ist nun nicht automatisch darin platziert, sondern muss hinein wechseln mit

- cd ionicangular1

```
PS C:\Users\edi\Desktop> cd ionicangular1
PS C:\Users\edi\Desktop\ionicangular1>
```

Nun starte ein neues Projekt mittels

- ionic start
- Namen: zweites-app und drücke auf Enter. Dann wähle mit den Pfeiltasten „blank“ aus.

```
? Project name: zweites app

Let's pick the perfect starter template!

Starter templates are ready-to-go Ionic apps that come packed with everything you need to bui
prompt next time, supply template, the second argument to ionic start.

? Starter template:
  tabs         | A starting project with a simple tabbed interface
  sidemenu     | A starting project with a side menu with navigation in the content area
> blank        | A blank starter project
  my-first-app | An example application that builds a camera with gallery
  conference   | A kitchen-sink application that shows off all Ionic has to offer
```

Nun wird im Hintergrund das Projekt erstellt.

Nun muss man in dieses Projekt wechseln mit

- cd zweites-app

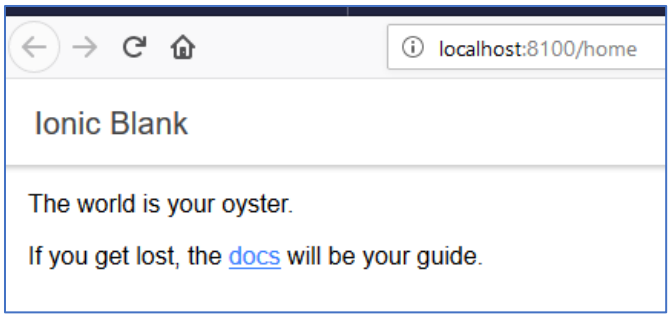
(Name darf kein Leerzeichen enthalten, Bindestrich ist ok).

Gib danach

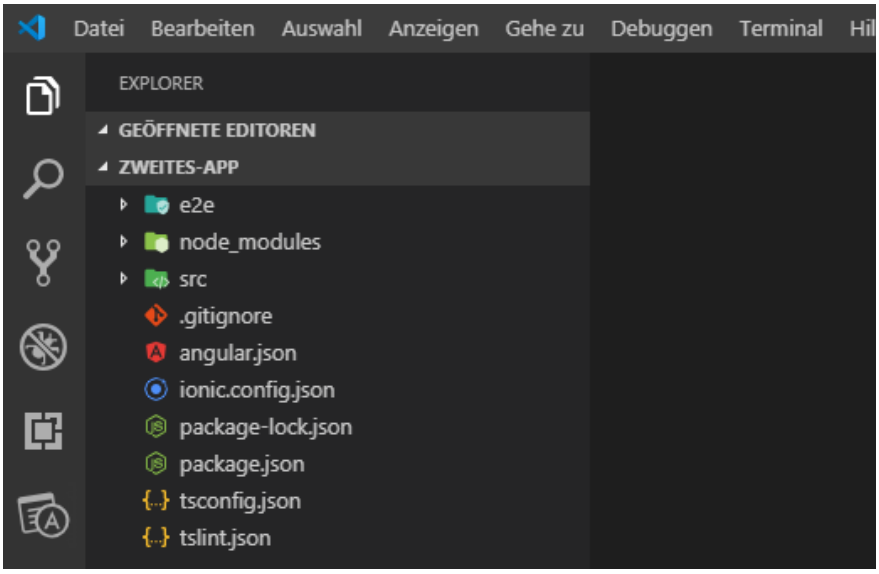
- ionic serve“ ein.

```
PS C:\Users\edi\Desktop\ionicangular1> cd zweites-app
PS C:\Users\edi\Desktop\ionicangular1\zweites-app> ionic serve
```

Ergebnis:

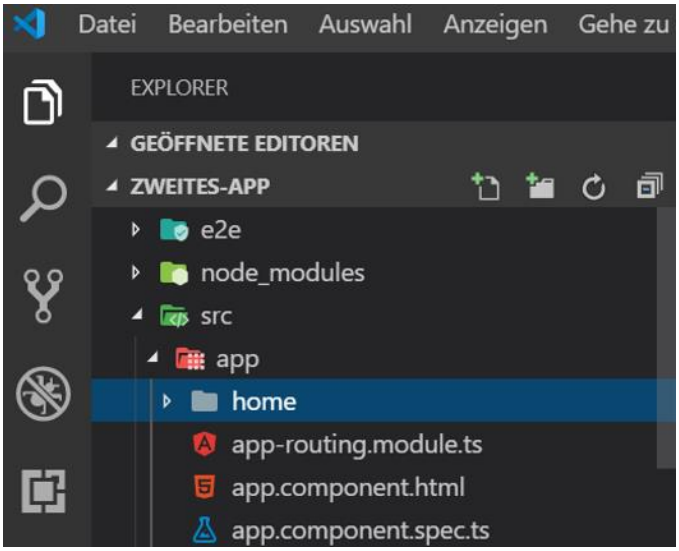


Dann muss man noch in VS-Code den neuen Ordner auf dem Desktop öffnen:

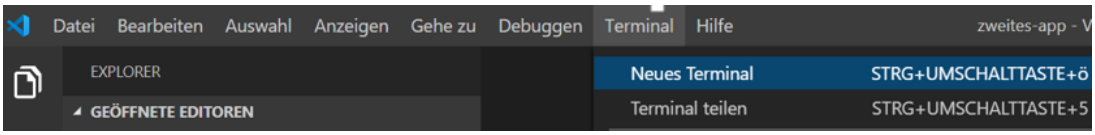


2)neue Seiten anlegen

Öffne „src“ und „app“ und lösche „home“.



Danach wird im Terminal eine neue Page erstellt: Klicke dazu oben im Menü auf „Terminal“



Folgender Code:

- ionic generate

Es werden einige Optionen angeboten. Wähle „page“

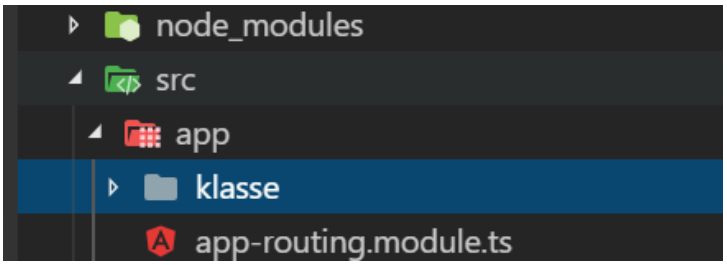
```
PS C:\Users\edi\Desktop\ionicangular1\zweites-app> ionic generate
? What would you like to generate? (Use arrow keys)
> page
  component
  service
  module
```

Und gib den Namen

- klasse

```
? What would you like to generate? page
? Name/path of page: klasse
```

Nun wird ein neuer Ordner erstellt:



Öffne „app-routing.module.ts“.

- in Zeile 7 sieht man das Laden dieser neuen Page. Das funktioniert hier automatisch „out of the box“ im Modus „lazy loading“. Das bedeutet, dass der Code nur geladen wird, wenn der User die Site besucht, also wenn er wirklich gebraucht wird.

```
4   const routes: Routes = [
5     { path: '', redirectTo: 'home', pathMatch: 'full' },
6     { path: 'home', loadChildren: './home/home.module#HomePageModule' },
```

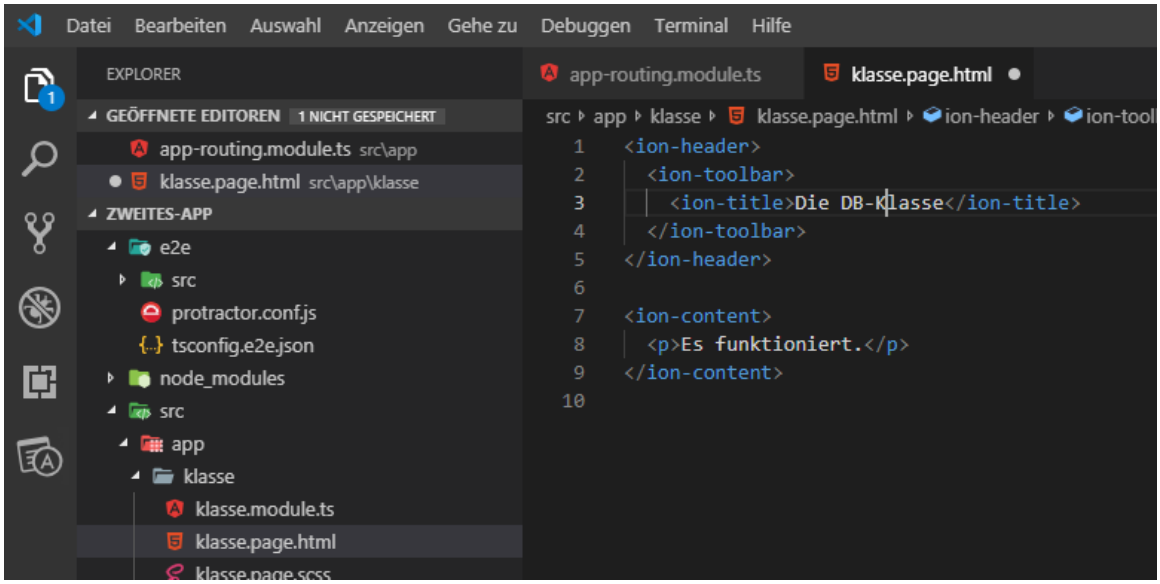
- in der Zeile 6 steht noch der „path“ zur „home“-page. Diese Zeile soll gelöscht werden.
- In Zeile 5 ändere das „redirectTo“ auf die neue Page „klasse“.

```
5     { path: '', redirectTo: 'klasse', pathMatch: 'full' },
6
7     { path: 'klasse', loadChildren: './klasse/klasse.module#KlassePageModule' },
8   ];
```

Speichere die Änderungen.

Neue Site bearbeiten. Öffne dazu „recipes.page.html“.

- Ändere den <ion-title> in „Die DB-Klasse“ und s
- schreibe in den Content einen Paragraf z.B. „Es funktioniert“.

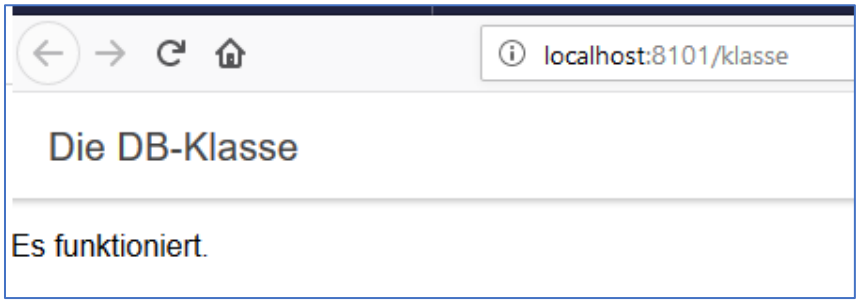


Speichern und die Seite ansehen.

Gib im Terminal ein „ionic serve“ und warte bis die Site kompiliert ist:

```
[ng] chunk {styles} styles.js, styles.js.map
[ng] chunk {vendor} vendor.js, vendor.js.map
[INFO] ... and 96 additional chunks
[ng] i [wdm]: Compiled successfully.
```

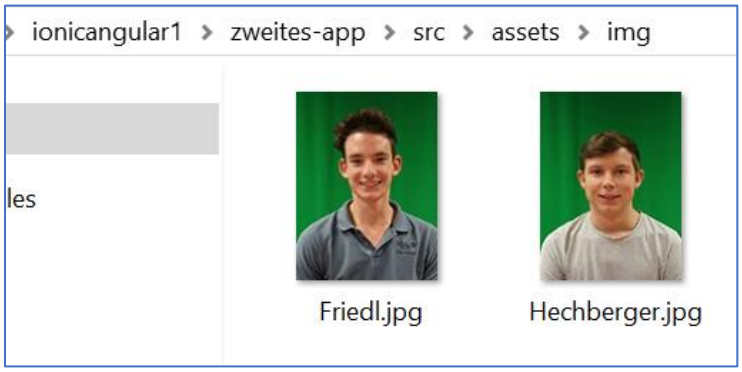
Ergebnis:



Liste in der Site erstellen

Vorbereitung: Bilder im richtigen Ordner ablegen:

Im Ordner „assets“ einen Ordner „img“ anlegen und dort die Bilder deponieren:



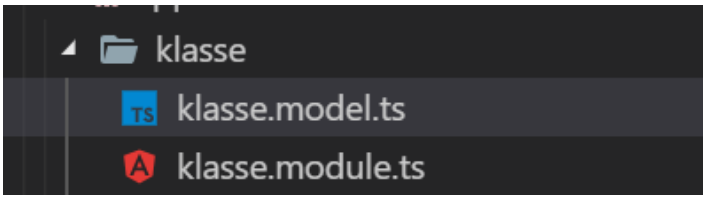
Lösche den Paragrafen und erstelle eine <ion-list>:

```
7   <ion-content>
8     <ion-list></ion-list>>
9   </ion-content>
```

Erstelle eine Datei „klasse.model.ts“

Diese gibt ein Muster vor, wie die Liste aussehen soll. Darin wird stehen, dass z.B. die „id“ einen Typ String bekommt, der Titel ebenfalls usw.

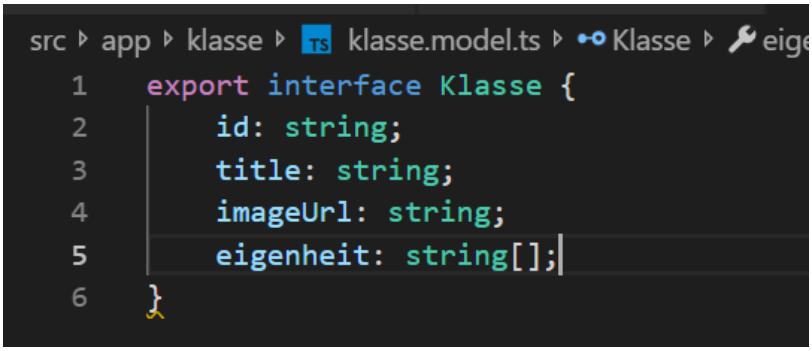
Klicke auf den Ordner „klasse“ und rechter Mausklick um eine neue Datei einzufügen:



Hier folgt ein TypeScript-Feature, das sie Struktur der Daten festlegt.

Darin soll ein „interface“ genutzt werden, welches die verschiedenen Item klären soll, die in der Liste angezeigt werden sollen.

Die „eigenheit“ wird als Array geführt, das später enthalten wird z.B. blond, männlich, …



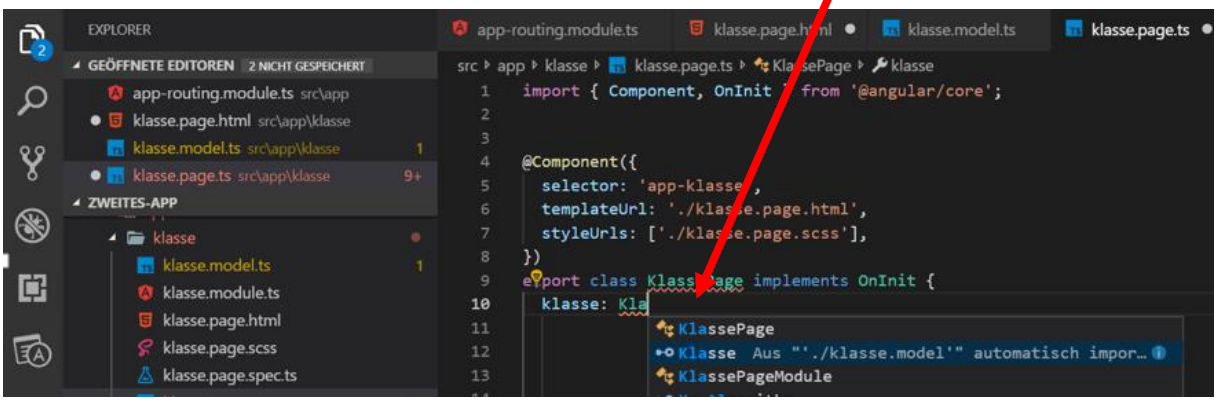
z.B. imageUrl ist String, weil hier erkennbar in der Dokumentation:
https://ionicframework.com/docs/api/img

Speichern.

klasse.page.ts

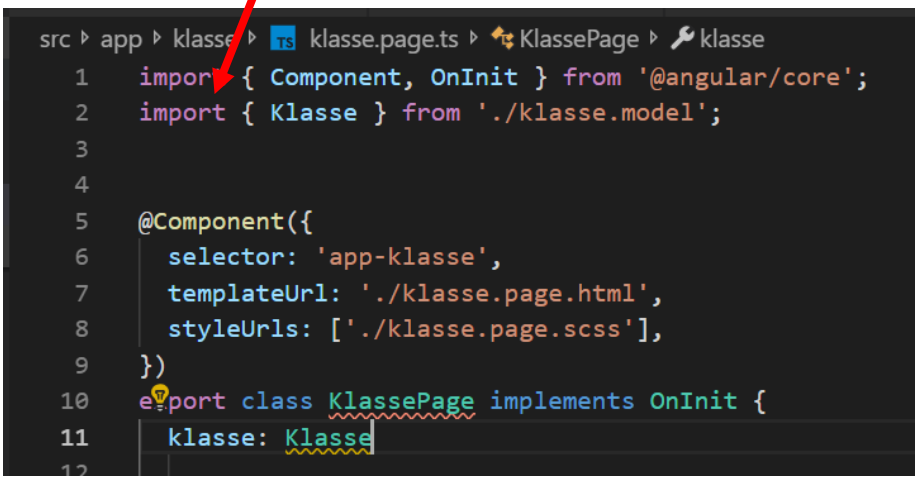
Nun fügt man eine Property ein, die vom Type „Klasse-Array“ ist.

Wichtig: Wenn die Property „klasse:“ eingegeben ist und man dann „Kl“ beginnt zu schreiben, muss man den Vorschlag übernehmen.



Dadurch wird automatisch der Type vom Klassen Model „importiert“.

Siehe Zeile 2:



Danach vervollständige dieses Array, getrennt mittels Beistriche:

```
10 export class KlassePage implements OnInit {
11   klasse: Klasse[] = [
12     {
13       id: 's1',
14       title: 'Friedl',
15       imageUrl: 'assets/img/Friedl.jpg',
16       eigenheit: ['schlank', 'keine Brille']
17     }
18   ];
```

Am Ende, nach der eckigen Klammer ein Strichpunkt.

Kopiere von Zeile 13 bis inkl. Zeile 17 und füge die Kopie in Zeile 16 ein. Füge in Zeile 17 nach der Klammer einen Beistrich ein.

Füge nun andere Info ein: Strichpunkt nicht vergessen

```
10  export class KlassePage implements OnInit {
11    klasse: Klasse[] = [];
```

```
10  export class KlassePage implements OnInit {
11    klasse: Klasse[] = [
12      {
13        id: 's1',
14        title: 'Friedl',
15        imageUrl: 'assets/img/Friedl.jpg',
16        eigenheit: ['schlank', 'keine Brille']
17      },
18      {
19        id: 's2',
20        title: 'Hechberger',
21        imageUrl: 'assets/img/Hechberger.jpg',
22        eigenheit: ['schlank', 'runde Brille']
23      }
24    ];
```

<u>Output in der klasse.page.html:</u>

Füge innerhalb der <ion-list> sogenannte <ion-item> ein:

```
7   <ion-content>
8     <ion-list>
9       <ion-item >
10
11      </ion-item>
12    </ion-list>
13  </ion-content>
```

Darin eine For-Schleife über alle Klassen.

<ion-avatar> ist eine Komponente in Zusammenhang mit <ion-img>. In <ion-img> bindet man das Bild mit eckigen Klammern [src] (property binding).

```
7  <ion-content>
8    <ion-list>
9      <ion-item *ngFor="let klasse of klasse">
10       <ion-avatar slot="start">
11         <ion-img [src]="klasse.imageUrl"></ion-img>
12       </ion-avatar>
13     </ion-item>
14   </ion-list>
15 </ion-content>
```

Info: https://ionicframework.com/docs/api/img

In der Dokumentation findet man die Details:

Usage

ANGULAR REACT VUE

HTML

```
<ion-list>
  <ion-item *ngFor="let item of items">
    <ion-thumbnail slot="start">
      <ion-img [src]="item.src"></ion-img>
    </ion-thumbnail>
    <ion-label>{{item.text}}</ion-label>
  </ion-item>
</ion-list>
```

src

Description	The image URL. This attribute is mandatory for the element.
Attribute	`src`
Type	`string | undefined`

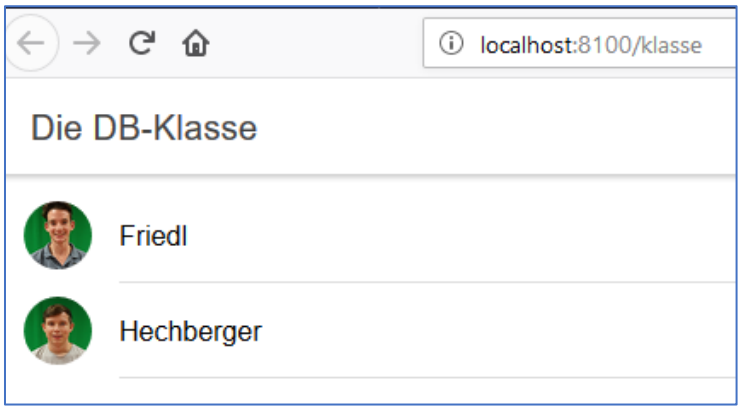
Im Avatar erfolgt nun noch eine Ausrichtung nach links mittels slot="start".

```
10 <ion-avatar slot="start">
11   <ion-img [src]="klasse.imageUrl"></ion-img>
12 </ion-avatar>
```

Unter dem Avatar ein <ion-label> für den Text, hier den Titel.

```
12        </ion-avatar>
13        <ion-label> {{ klasse.title}}  </ion-label>
14      </ion-item>
```

Ergebnis:



3)Zweite Seite anlegen

Ziel: Diese Seite soll per Klick die Details und ein größeres Bild darstellen.

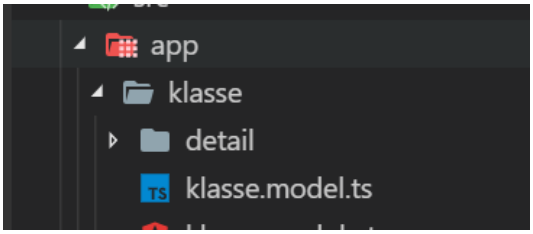
Erstelle eine neue Site. Diese soll im Ordner „klasse“ angelegt werden.

Dazu kann der Ordner und dann mit Slash eine neue Site darin abgelegt werden:

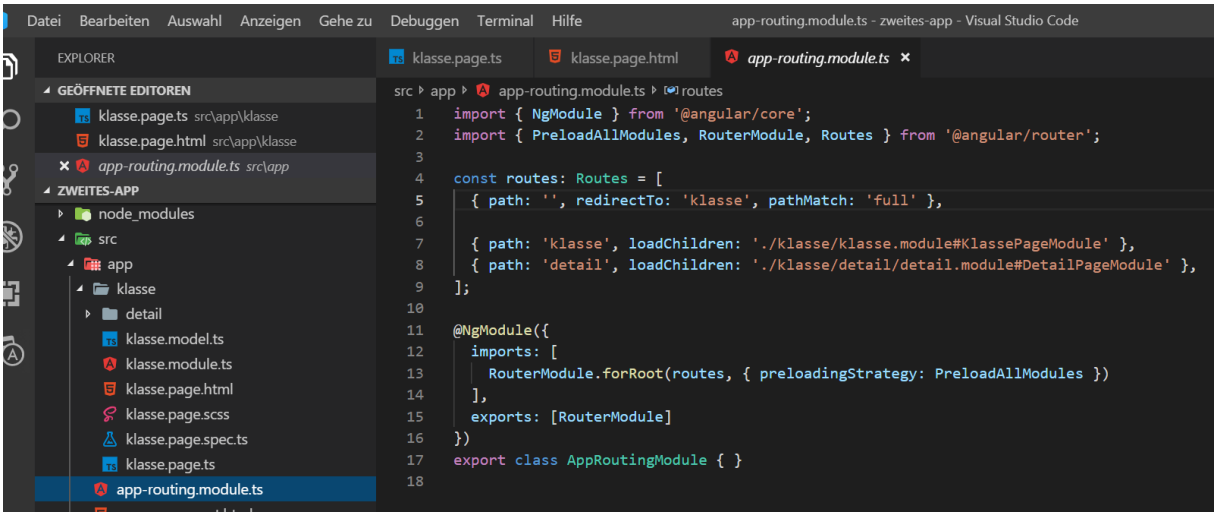
Im Terminal:

```
PS C:\Users\edi\Desktop\ionicangular1\zweites-app> ng generate page klasse/detail
```

Ergebnis:



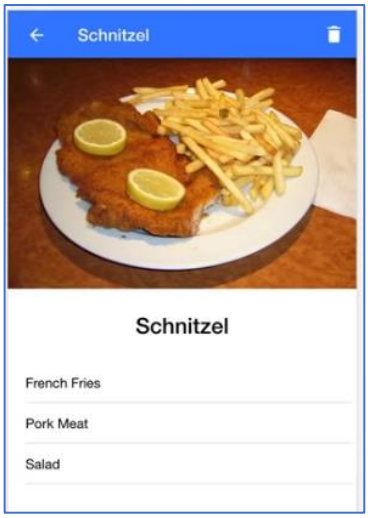
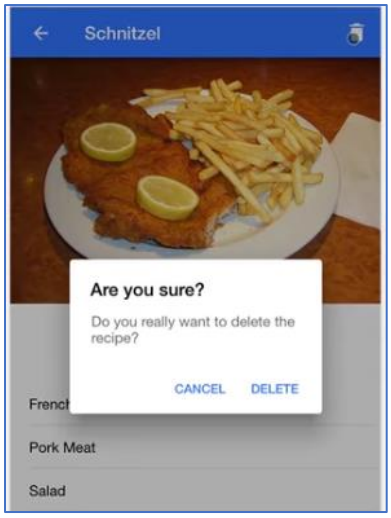
Die Routing-Pfade sind dadurch automatisch angelegt. Siehe dazu in der Datei „app-routing.module.ts“ die Zeile 8:



Weitere Info:

https://www.youtube.com/watch?v=r2ga-iXS5i4

- ab 2:48:30
-
- Inkl. ein Angular-Service nutzen.

Folgende Bereiche werden gezeigt:

- routerLink 3:08:25
- Back-Button 3:09
- Delete-Button 3:14
- Inkl. alert Controller 3:17:30