FULLSTACK WEBANWENDUNG

ANGULAR UND NODE.JS-KONZEPT

ALEXANDER TEICHMANN
KEVIN HAYES
LUKAS LOKJANOW
GIANFRANCO MANGIONE
BUSRANUR HINISLIOGLU

COMPONENTS:

Pages:

Cart-Page Home-page Success-page

Product-Page Add-page Admin-Page

Checkout

Partials:

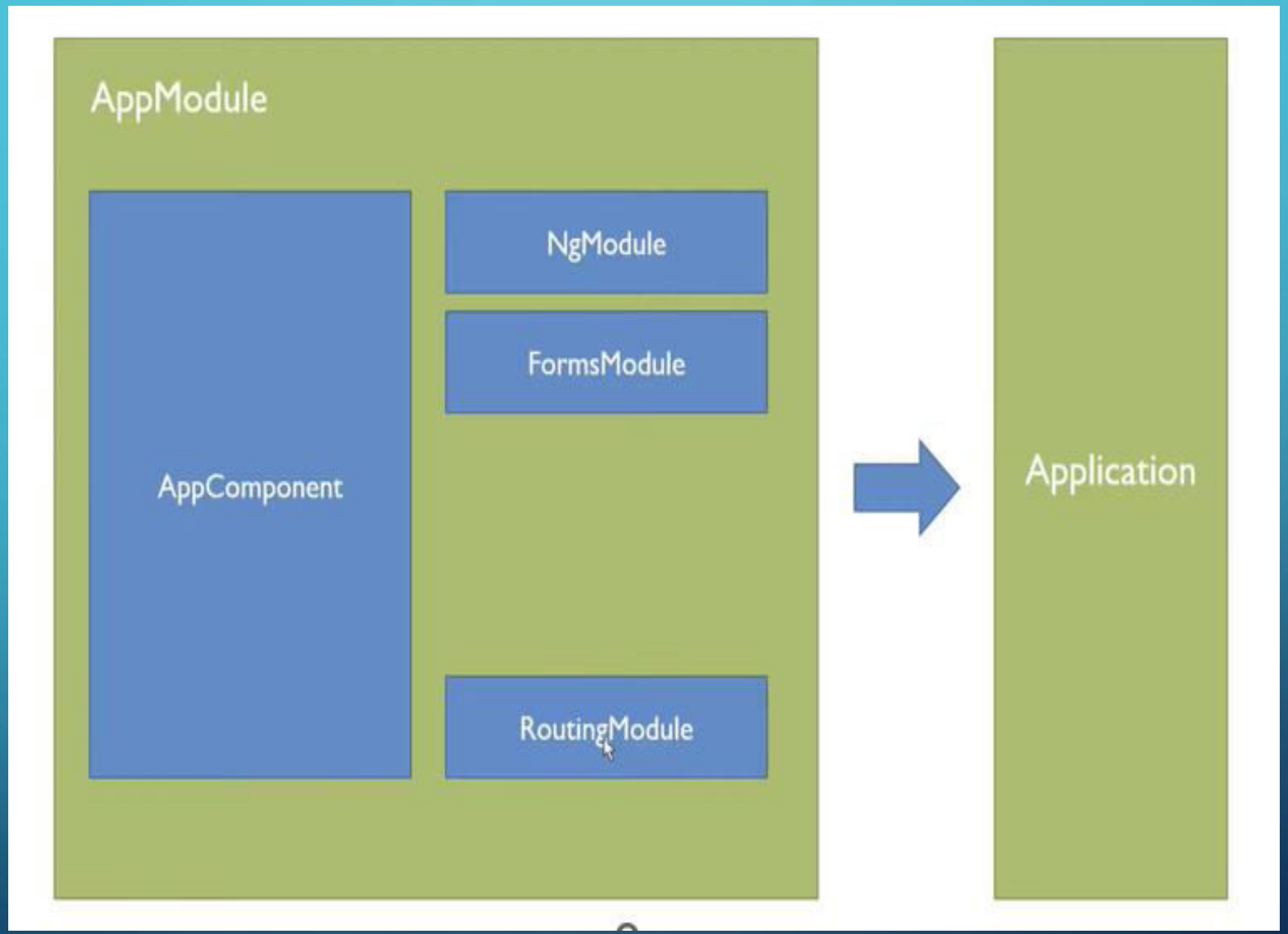
Category Header Image-Container Not-Found

Search Title Input-Container Input-Validation

Order-items-list Text-inputs

```
@NgModule({
  declarations: [
    AppComponent,
    HeaderComponent,
    HomeComponent,
    SearchComponent,
    CategoryComponent,
    ProductPageComponent,
    CartPageComponent,
    TitleComponent,
    NotFoundComponent,
    ImageContainerComponent
  ],
```

- AppModule: Dieser Befehl initialisiert das AppModule und lädt alle damit verbundenen Module.
- AppComponent: Hier wird das AppComponent als Startkomponente angegeben, die die gesamte Anwendung lädt.
- Declaration in app.module.ts: Der Angular-Code für die Deklaration der Module und Komponenten wird in der app.module.ts-Datei geschrieben


AppModule
AppComponent
NgModule
FormsModule
RoutingModule
Application

ERSTELLUNG VON MODELLEN, SERVICES UND KOMPONENTEN IN ANGULAR

- Erstellen Sie das CartItem-Modell und das Product-Modell.
- Fügen Sie alle Methoden hinzu, die in ... verwendet werden, zum Product Service oder Cart Service.
- Generieren Sie die gewünschte Seite oder Komponente.
  - Route
  - TypeScript-Code
  - HTML-Code
  - CSS-Code

```
@Component({
  selector: 'app-cart-page',
  templateUrl: './cart-page.component.html',
  styleUrls: ['./cart-page.component.scss']
})
```

VERBINDUNG ZUM BACKEND-PROJEKT

- Fügen Sie APIs zum Backend-Projekt hinzu, um die Verarbeitung von HTTP-Anfragen (GET, POST, DELETE, PUT, etc.) und den Zugriff auf Daten zu ermöglichen.

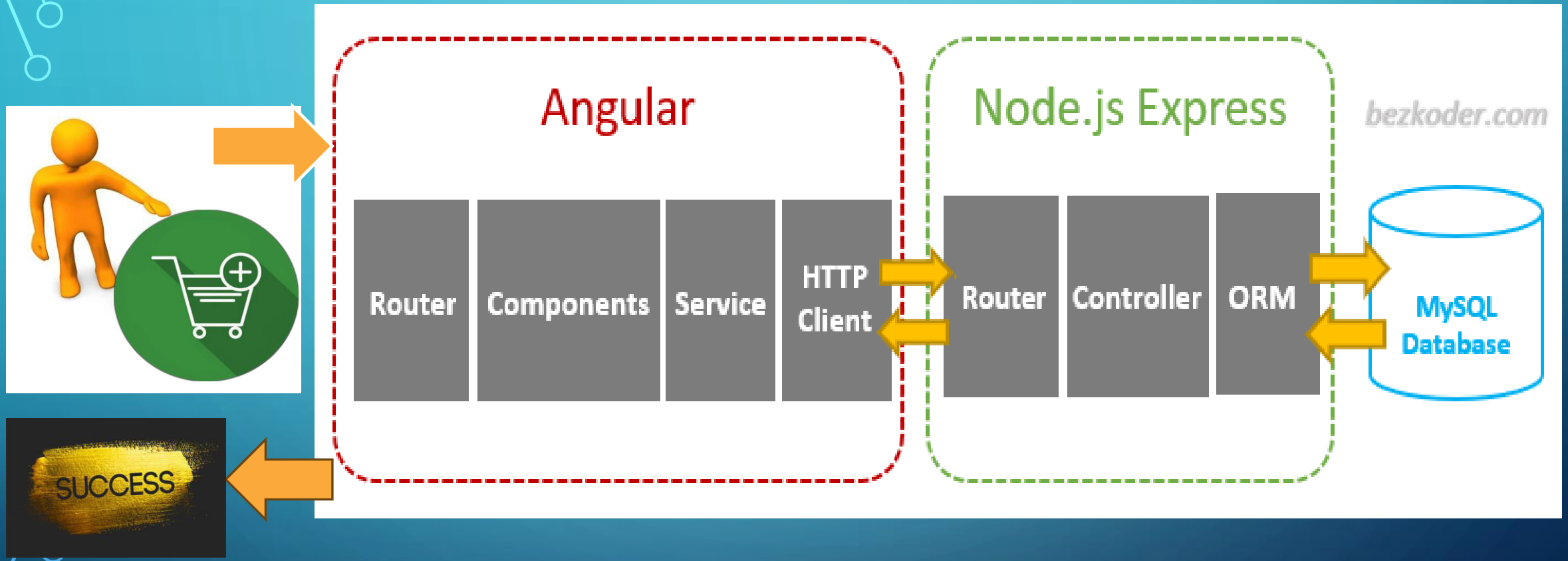
Node.js

- Verwenden Sie Node.js, um auf Client-Anfragen zu reagieren, mit Datenbanken zu kommunizieren und dynamische Webseiten zu erstellen.

Database

- Nutzen Sie MariaDB Database with Docker Container, um die Anwendungen und deren Abhängigkeiten in isolierten Umgebungen zu verpacken und bereitzustellen.

API


Angular
Router
Components
Service
HTTP Client
Node.js Express
Router
Controller
ORM
bezkoder.com
MySQL Database
SUCCESS

CSS - SCSS

```scss
/* Corporate Identity Colors */
$text: #ffffff;
$background: #1E2328;
$primary-color: #0e4452;
$secondary-color: #a09b8c;
$accent: #c89b3c;

$tertiary-color: #CDFAFA;
```

SCSS

- ermöglicht die Verwendung von Variablen ($boxColor, $fontsize)
- Wir definieren einmal, benutzen wir überal

```scss
$ci-highlight: #C89B3C;
$ci-background-dark: #010A13;
$ci-background-normal: #1E232D;
$ci-background-highlight: #3C3C41;
$ci-background-light: #A09B8C;


div {
    color: $ci-background-light;
    border: 5px solid $ci-highlight;
    position: absolute;
    top: 50%;
    left: 50%;
    transform: translate(-55%, -50%);
    padding: 10px;
    font-size: 2rem;
}
```

CSS

- viel redundanten Code

```scss
a{
    font-size: 1.1rem;
    background-color: $accent;
    color: black;
    border-radius: 10rem;
    padding: 0.7rem 1rem;
    margin: 1rem;
    opacity: 0.8;
    font-weight: bold;
    &:hover{
        opacity: 1;
        cursor: pointer;
    }
}
```

@MIXIN

- einfacheres und organisierteres Format
- ermöglicht mathematische Operationen direkt im Code(extra)

```
.article {
    float: left;
    width: 600px / 960px * 100%;
}
```

```
@mixin input-style($border-type:0){
    width: 40vw;
    background-color: transparent;
    @if $border-type == bot {
        border: none;
        border-bottom: 2px solid $accent;
    } @else {
        border: 2px solid $accent;
    }
    outline: none;
    color: $accent;
    background-color: $primary-color;
}
```

```
form > div > textarea {
    @include input-style();
    max-width: 88vw;
}
```