



Angular Aufgaben

Kleines Projekt

Datum: 22.12.2021

Übersicht

- Git → Projekt herunterladen
- Angular aufsetzen
- Layout & Design
- Routing
- Login/Logout erstellen
- Tabellen auf separate Seite anzeigen

Verfügbare Materialien

1. NodeJS
2. Visual Studio Code
3. Git
4. Internet-Zugang

Übung

Angular 13 mit Material - Erste Schritte

Lade das Projekt herunter und erstelle einen neuen Zweig mit deinen Namen:

<https://github.com/AndyNope/simple-angular-material.git>

Im Projekt ist bereits ein Layout und Angular Material Design eingebaut.

Für das Projekt fehlen jedoch noch Module, welches mit Node.js (in der Konsole) noch aufgesetzt werden muss:

```
projekt-pfad> npm i
```

Nun muss die Angular-Library noch installiert werden (Falls nicht vorhanden) :

```
projekt-pfad> npm install -g @angular/cli
```

Angular-Projekt starten

Um das Projekt zu starten, kannst du folgende Befehle eingeben:

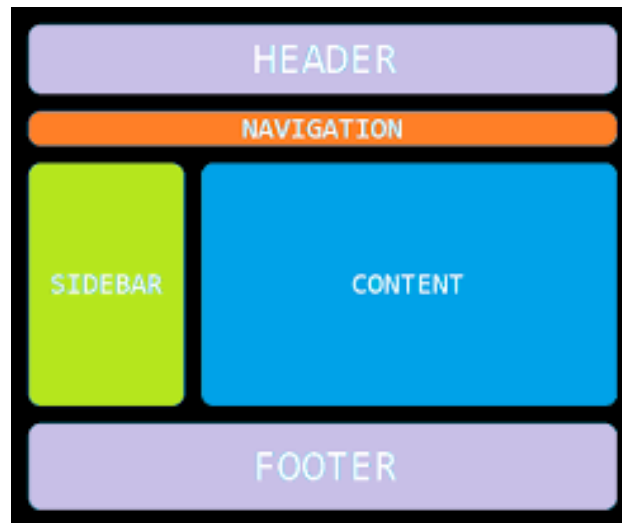
```
projekt-pfad> npm start
```

oder

```
projekt-pfad> ng serve
```

Aktuelles Layout:

- Header
- Navigation
- Slidenbar
- Main-Container / Body
- Footer



Design-Werkzeuge

- CSS / SCSS
- Material-Design
- Bootstrap (optional)

Bootstrap (optional)

Falls du dich mit Bootstrap auskennst und damit arbeiten möchtest, kannst du es wie folgt einbauen:

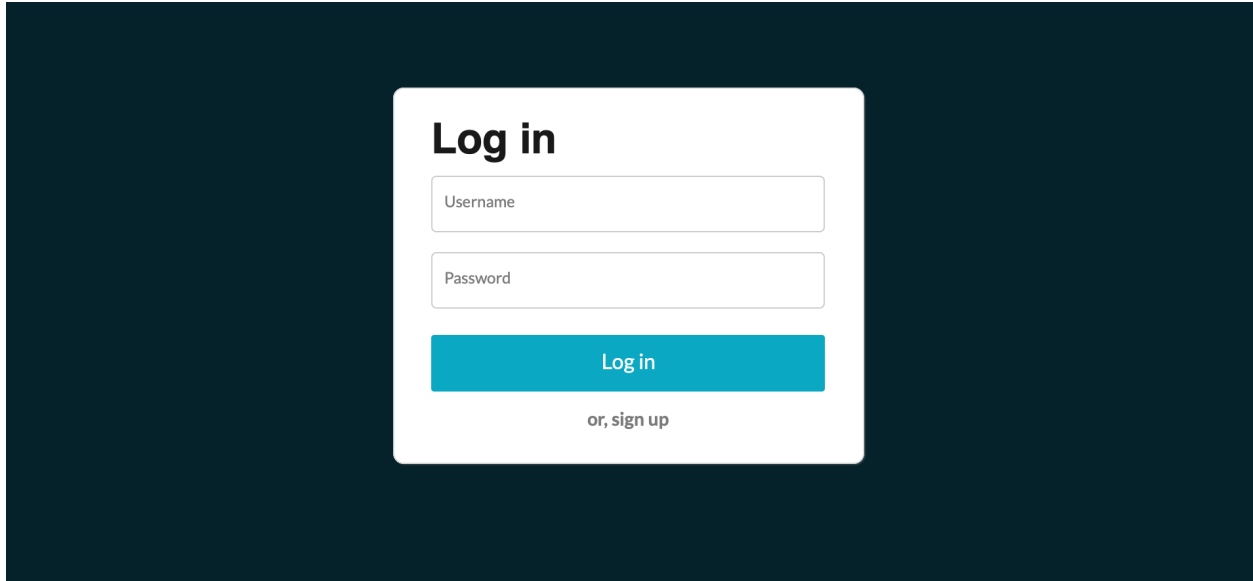
```
projekt-pfad> npm i bootstrap
```

In angular.json muss du noch unter Style folgenden Pfad hinzufügen:

```
"styles": [  
  "node_modules/bootstrap/dist/css/bootstrap.min.css", ...  
],
```

Login-Seite

Erstellen Sie hier eine Login-Seite. Clients sollte ohne Anmeldung nicht erreichbar sein.



Die Logik sollte als Beispiel bereits eingebaut sein. Styles wurden hier noch nicht hinzugefügt. Hier kannst du es selbst gestalten oder Material Design verwenden:

<https://material.angular.io/components/categories>

Um dem Zugriff auf der Hauptseite zu verhindern, kannst du CanActivate verwenden:

<https://angular.io/api/router/CanActivate>

Wir nennen dies meistens einen "AuthGuard", da dies den Zugang beschützt, bevor die Komponente geladen werden kann.

Clients

Auf dieser Seite sollte eine Tabelle (Paginator) dargestellt werden.

Clients-Seite sollte nur eine bestimmte Anzahl auflisten.

Du kannst die Anzahl Einträge dynamisch gestalten:

Variablen:

```
@ViewChild('datatable') elementDatatable: ElementRef;  
ROW_HEIGHT = 37;  
HEIGHT_SPACE = 200;
```

Berechnung:

```
this.pageSize = Math.round(  
(innerHeight - this.HEIGHT_SPACE - this.elementDatatable.nativeElement.offsetTop)  
/ this.ROW_HEIGHT  
);
```

Element kennzeichnen (für @ViewChild('datatable')):

```
<div #datatable>  
...  
</div>
```

Der Inhalt sollte in einem Modal/Popup angezeigt werden.

Hierzu kannst du Bootstrap Modal, Angular Dialog, oder Falls die Zeit nicht reicht, einfach einen alert einbauen.

Man kann eine Person aus der Datenbank auswählen und dessen Name soll im Header anzeigen. Hierzu sollte an Angular Input() und Output() verwenden:

(Data-Binding)

<https://angular.io/guide/inputs-outputs>

Diagramme (Optional)



Stelle die verschiedenen Länder der Person grafisch dar. Die Darstellung ist frei wählbar.

Hier kannst du ein Algorithmus schreiben, um die Länder auszusortieren und zu filtern.

Daten JSON & API

API GET: <https://api.andynope.com/clients/>

Response:

```
[{
  "id": string,
  "name": string",
  "phone": string,
  "email": string,
  "address": string,
  "postalZip": string,
  "region": string,
  "country": string
}]
```

API POST: <https://api.andynope.com/login/>

Request:

```
{
  "username": "KreativMedia"
  "password": "b3E!13hy"
}
```

API: <https://api.andynope.com/logout/>

API GET: <https://api.andynope.com/session/>

API GET: <https://api.andynope.com/paginator/>

Request:

```
{  
  "pageSize": number,  
  "pageIndex": number  
}
```

Response:

```
[{  
  "id": string,  
  "name": string,  
  "phone": string,  
  "email": string,  
  "address": string,  
  "postalZip": string,  
  "region": string,  
  "country": string  
},  
  "pageSize": number,  
  "pageIndex": number,  
  "quantity": number  
]
```