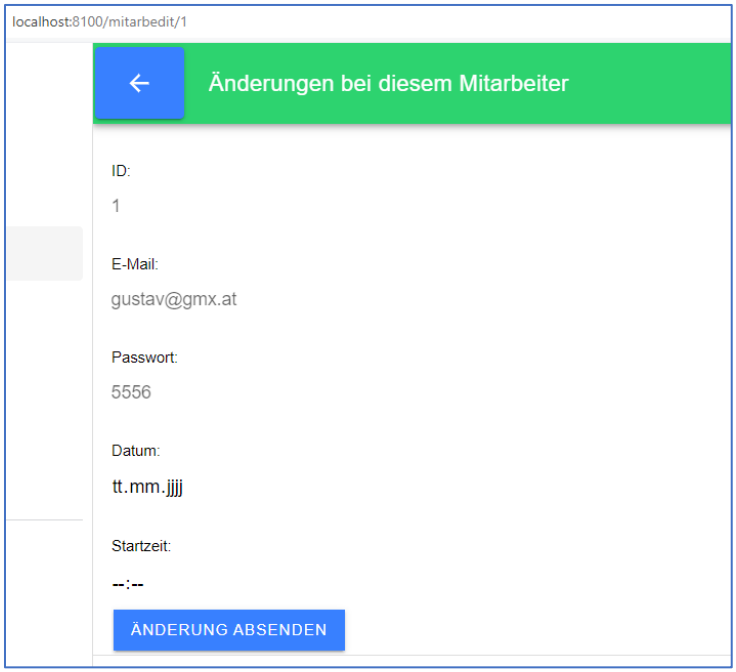
Ionic 13: Ändern (Editieren) eines Mitarbeiters – Abspeichern des Updates in die Datenbank

Keine sessionStorage, nur mit ID aus der Datenbank

Testen und ansehen auf: https://ionic.digbizmistelbach.info/login

Ergebnis:



Noch offen:

- Die Angaben im placeholder werden noch nicht übernommen. **Daher MÜSSEN alle Daten NEU ausgefüllt** werden, damit sie an das apiService übergeben werden. Ansonsten würde der Button nicht klickbar sein.
- Leider ist die Liste danach nicht aktuell und man muss aktualisieren

Von der Seite „mitarbdetail“ wird man hier durch den Klick auf den Button „editieren“ auf die neu zu erstellende Seite „mitarbedit“ geleitet, wo man die Daten einliest und dann ändern kann.

Info:

- Am Beginn ist die neue „mitarbedit“ eigentlich gleich wie die schon vorhandene „mitarbdetails“. Daher kann man viel davon übernehmen (kopieren) und einige „Namens-Anpassungen“ durchführen.
- Erst später muss man dazu hineinarbeiten, dass es auch <input>-Felder eines Formulars aufnehmen soll, damit man, wie beim Registrieren, Daten in die Datenbank senden kann (UPDATE statt INSERT INTO in der zuständigen API (PHP)).

a)Erstelle „mitarbEdit“ im Terminal

```
ionic g page pages/mitarbedit
```

Übernimm aus der „mitarbdetail.pages.html“ den Zurück-Historie Code und ändere gleichzeitig die Farbe und den Titel:

```
<ion-button slot="start">
    <ion-back-button></ion-back-button>
  </ion-button>
```

mitarbedit.page.html U mitarbdetail.page.html U

src > app > pages > mitarbedit > mitarbedit.page.html > ion-header > ion-toolbar

Go to component

```
1  <ion-header [translucent]="true">
2    <ion-toolbar color="success">
3      <ion-title>Änderungen bei diesem Mitarbeiter</ion-title>
4      <ion-button slot="start">
5        <ion-back-button></ion-back-button>
6      </ion-button>
7    </ion-toolbar>
8  </ion-header>
```

In den Content füge eine Kopie des schon vorhandenen Contents aus „mitarbeiter.page.html“ ein und ändere ein paar Elemente. Nimm auch die „iduser“ auf.

Der Kunde soll gleich im „placeholder“ die Daten angezeigt bekommen.

Datum: Type = date
Startzeit: Type = time
Wird so dann abgefragt:

Datum:
TT . MM . JJJJ

Startzeit:
-- : --

```
10  <ion-content [fullscreen]="true">
11    <ion-list>
12      <ion-item lines="insert" *ngFor="let item of mitarbedit">
13        <h5><ion-label position="stacked">ID: </ion-label></h5>
14        <ion-input type="text"  placeholder="{{item.iduser}}"  [(ngModel)]="id"></ion-input>  <!--nichts us DB-
15        <h5><ion-label position="stacked">E-Mail: </ion-label></h5>
16        <ion-input type="text"  placeholder="{{item.mail}}"   [(ngModel)]="email"></ion-input>
17        <h5><ion-label position="stacked">Passwort:  </ion-label></h5>
18        <ion-input type="text"  placeholder="{{item.passwort}}"  [(ngModel)]="passwort"></ion-input>
19        <h5><ion-label position="stacked">Datum:  </ion-label></h5>
20        <ion-input type="date"  placeholder="{{item.datum}}"  [(ngModel)]="datum"></ion-input>
21        <h5><ion-label position="stacked">Startzeit:  </ion-label></h5>
22        <ion-input type="time"  placeholder="{{item.startzeit}}"  [(ngModel)]="startzeit"></ion-input>
23
24        <ion-button color="primary" size="full" (click)="speichern()" >Änderung absenden</ion-button>
25      </ion-item>
26    </ion-list>
27  </ion-content>
```

```
  <ion-item lines="insert" *ngFor="let item of mitarbedit">
    <h5><ion-label position="stacked">ID: </ion-label></h5>
    <ion-input type="text"  placeholder="{{item.iduser}}"  [(ngModel)]="id"></ion-input>  <!--nichts
us DB-->
    <h5><ion-label position="stacked">E-Mail: </ion-label></h5>
    <ion-input type="text"  placeholder="{{item.mail}}"   [(ngModel)]="email"></ion-input>
    <h5><ion-label position="stacked">Passwort:  </ion-label></h5>
    <ion-input type="text"  placeholder="{{item.passwort}}"  [(ngModel)]="passwort"></ion-input>
    <h5><ion-label position="stacked">Datum:  </ion-label></h5>
    <ion-input type="date"  placeholder="{{item.datum}}"  [(ngModel)]="datum"></ion-input>
    <h5><ion-label position="stacked">Startzeit:  </ion-label></h5>
```

```
    <ion-input type="time"  placeholder="{{item.startzeit}}"  [(ngModel)]="startzeit"></ion-input>

    <ion-button color="primary" size="full" (click)="speichern()" >Änderung absenden</ion-button>
  </ion-item>
</ion-list>
```

Damit der Kunde die Änderungen bequem vornehmen kann, sollen die Daten nicht nebeneinander, sondern untereinander angezeigt werden.

Nach jedem Element soll ein <input>-Feld des Formulars folgen, um Änderungen eintragen zu können.

Dazu nutze das Attribut **position="stacked":**

Das Attribut position="stacked" typischerweise innerhalb des <ion-item>-Elements verwendet. Dadurch wird

- das Label über das Input-Feld platziert, anstatt es danebenzustellen
- vertikale Ausrichtung: das Label wird über dem Input platziert

Natürlich wird die Seite noch nicht anzeigefähig sein, daher muss man noch in der Export-Klassen einfügen, die TS herrichten usw.

2)mitarbedit.page.ts

Öffne die „mitarbedit.page.ts“ für die Eingabe des Control-Codes:

Als erstes füge in die Klasse den Begriff ein, der in der *ngFor-Schleife nach dem „item“ angeführt ist:

mitarbedit: any;

```
})
export class MitarbeditPage implements OnInit {
  mitarbedit: any;

  constructor() { }
```

Speichern. Nun sollte die Website wieder verwendbar sein, ohne von einem Fehler zu melden.

Auf jeden Fall werden wir im Constructor die beiden Methoden benötigen. Beachte auch einen entsprechenden IMPORT dieser!

```
private route: ActivatedRoute,
private apiService: ApiService
```

```
constructor(
  private route: ActivatedRoute,
  private apiService: ApiService) { }
```

```
import { Component, OnInit } from '@angular/core';
import { ActivatedRoute } from '@angular/router';
import { ApiService } from 'src/app/api.service';

```

3)den Button „editieren“ in „mitarbdetail.page.html“ verlinken

Öffne die mitarbdetail.page.html“ und ändere den Link auf diese neue Seite „mitarbedit“:

```
      <ion-item lines="insert" *ngFor="let item of mitarbdetails">
        <ion-label><h5>E-Mail:{{item.mail}}</h5></ion-label><br>
        <ion-label><h5>Passwort:  {{item.passwort }}</h5></ion-label><br>
        <ion-button color="primary" routerLink="/mitarbedit/{{item.iduser}}">
          <ion-icon name="create"></ion-icon>
```

Hole danach auch den Inhalt aus „ngOnInit“ – also der Startposition dieser Seite – aus der schon vorhandenen „mitarbdetails.page.ts“.
Dort wird schon erfolgreich der Inhalt aus der Datenbank bezüglich des gewählten Mitarbeiters anhand dessen ID ausgewählt und angezeigt. Die API ist dieselbe, daher bleibt der Name die „apiService“ gleich.
Nur geändert werden muss

- this.mitarbdetails = response;
  auf
- this.mitarbedit = response;

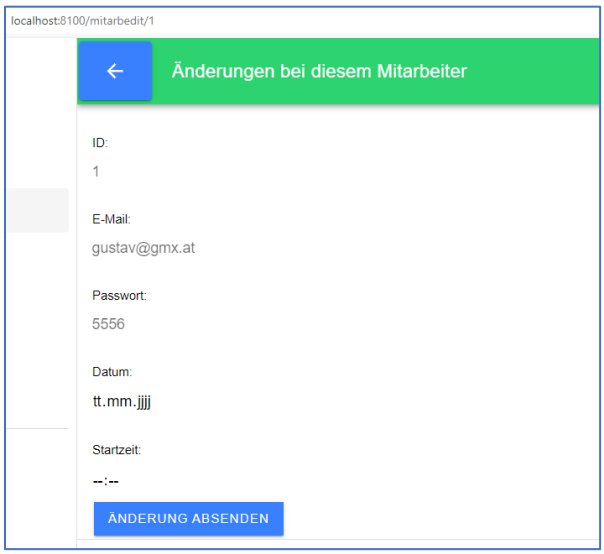
```
  ngOnInit() {
    const id = this.route.snapshot.paramMap.get('id');
  // console.log(id);
    this.apiService.getMitarbDetail(id).subscribe((response: any) => {
    this.mitarbedit = response;
    });
  }
  }

```

4)Pfad in „app-routing.module.ts“ anpassen

Damit der Pfad auch ausgeführt wird, weil ja der „routerink“ mit der ID gesendet wird, muss diese ID auch dem Pfad angehängt werden:

```
  {
    path: 'mitarbedit/:id',
    loadChildren: () => import('./pages/mitarbedit/mitarbedit.module').the
  }
];
```

Ergebnis:



5)Input und Funktion „speichern“

Damit der Kunde Änderungen im Input-Feld eintragen kann, müssen die Felder die Daten weitergeben können. Das erfolgt z.B. mit

- [(ngModel)]="email"

Davor muss man in der Export-Klasse die Begriffe bekannt machen und mit „any“ allgemeingültig machen. Dazu die gleichen Begriffe wie in der login.page.ts

benutzername: any;
email: any;
passwort: any;
id: any;
datum: any;
startzeit: any;

```
11  export class MitarbeditPage implements OnInit {
12    mitarbedit: any;
13    benutzername: any;
14    email: any;
15    passwort: any;
16    id: any;
17    datum: any;
18    startzeit: any;
```

Somit kann man dann auch in der HTML-Datei diese in das Input-Feld einfügen:

```
<ion-list>
  <ion-item lines="insert" *ngFor="let item of mitarbedit">
    <h5><ion-label position="stacked">ID: </ion-label></h5>
    <ion-input type="text"  placeholder="{{item.iduser}}"  [(ngModel)]="id"></ion-input>  <!--nichts us DB--
    <h5><ion-label position="stacked">Mitarbeitername: </ion-label></h5>
    <ion-input type="text"  placeholder="{{item.benutzername}}"   [(ngModel)]="benutzername"></ion-input>
    <h5><ion-label position="stacked">E-Mail: </ion-label></h5>
    <ion-input type="text"  placeholder="{{item.mail}}"   [(ngModel)]="email"></ion-input>
    <h5><ion-label position="stacked">Passwort:  </ion-label></h5>
    <ion-input type="text"  placeholder="{{item.passwort}}"  [(ngModel)]="passwort"></ion-input>
    <h5><ion-label position="stacked">Datum:  </ion-label></h5>
    <ion-input type="date"  placeholder="{{item.datum}}"  [(ngModel)]="datum"></ion-input>
    <h5><ion-label position="stacked">Startzeit:  </ion-label></h5>
    <ion-input type="time"  placeholder="{{item.startzeit}}"  [(ngModel)]="startzeit"></ion-input>

    <ion-button color="primary" size="full" (click)="speichern()" >Änderung absenden</ion-button>
  </ion-item>
</ion-list>
```

Beachte:
Wenn die „id“ als „disabled“ gesetzt würde, könnte es nicht später in das input eingefügt werden, und somit würde das apiService blockiert werden. Natürlich sollte die ID vom Kunden auch nicht verändert werden können, aber vorerst müssen wir damit leben, dass sie auch zur Überschreibung dienen kann. Das sollten wir später lösen.

6)Funktion „speicher“ erstellen

In den „Absenden“-Button wird die click-Funktion „speichern“ gestellt.

```
    <ion-button color="primary" size="full" (click)="speichern()" >Änderung absenden</ion-button>
  </ion-item>
```

Das führt vorerst zu einem Fehler, da erst jetzt diese Funktion in der .ts angelegt wird.

In der .ts:

```
  }

speichern(){

}
```

Den Inhalt kopiere aus der registrieren.ts oder der login.ts und ändere sie entsprechend.

Nun folgt die Verknüpfung zur Funktion im API:

this.apiService.speichernmitarb(userdata).subscribe((res: any) => { });

In dieser Funktion

- Erfolgt die Verbindung zum Service mit dem Namen „speichermitarb“
- Diese hat Zugriff auf die userdata
- Dann das Observable mit dem subscribe

Aber das Service „speichernmitarb“ existiert noch nicht, daher die Fehlermeldung.

```
35  speichern(){
36    const userdata = {
37      id: this.id,      //kein Begriff aus der DB nur wenn man alles neu eingibt
38      benutzername: this.benutzername,
39      email: this.email,      //kein Begriff aus der DB
40      passwort: this.passwort,    //kein Begriff aus der DB
41      datum: this.datum,
42      startzeit: this.startzeit
43    };
```

```
43    };
44
45    this.apiService.speichernmitarbd(userdata).subscribe((res: any) => {
46      //console.log('SUCCESS ===',res);
47      // if (res.error !== true) {
48        this.router.navigate(['/mitarbeiter']);   //geht nur, wenn man alles neu eingibt
49      });
50    }
51  }
```

```
speichern(){
 const userdata = {
  id: this.id,     //kein Begriff aus der DB nur wenn man alles neu eingibt
  benutzername: this.benutzername,
  email: this.email,     //kein Begriff aus der DB
  passwort: this.passwort,    //kein Begriff aus der DB
  datum: this.datum,
  startzeit: this.startzeit
 };

 this.apiService.speichernmitarbd(userdata).subscribe((res: any) => {
  //console.log('SUCCESS ===',res);
  // if (res.error !== true) {
    this.router.navigate(['/mitarbeiter']);   //geht nur, wenn man alles neu eingibt
  });
 }
```

7)apiService anpassen und neues Service erstellen

In der index.php der API liegt die Funktion „speichermitarb“. Diese schickt mit PHP-Code (UPDATE) die Daten in die Datenbank.

Nach einem Beistrich muss noch das Array „userdata“ übergeben werden, damit es weitergeleitet werden kann.

```
59  speichernmitarb(userdata: any){
60  return this.http.post('https://ionic.digbizmistelbach.info/api_einkaufsliste/public/index.php/speichernmitarb',userdata);
61  }
62  }
63
```

```
speichernmitarb(userdata: any){
return
this.http.post('https://ionic.digbizmistelbach.info/api_einkaufsliste/public/index.php/speichernmitarb',userdata);
```

}

8)API

```
176
177 $app->post('/speichernmitarb', function(Request $request, Response $response){
178     // get the parameter from the form submit
179     $id = $request->getParam('id');
180     $bn = $request->getParam('benutzername');
181     $email = $request->getParam('email');
182     $pwd = $request->getParam('passwort');
183     $datum = $request->getParam('datum');
184     $startzeit = $request->getParam('startzeit');
185
186 $sql = "UPDATE user SET benutzername=:bn, mail=:email, passwort=:pwd, datum=:datum,
187         startzeit=:startzeit WHERE iduser=$id";
188     try {
189         $db = new db();
190         $db = $db->connect();
191         $stmt = $db->prepare($sql);
192         $stmt->bindParam(':bn', $bn);
193         $stmt->bindParam(':email', $email);
194         $stmt->bindParam(':pwd', $pwd);
195         $stmt->bindParam(':datum', $datum);
196         $stmt->bindParam(':startzeit', $startzeit);
197
198
199         $stmt->execute();
200
201         $result = array();
202           $result['success'] = true;
203           $result['message'] = 'Registrierung abgeschlossen';
204           echo json_encode($result);
205
206     }
207     catch(PDOException $e) {
208        echo '{"error": {"msg": ' . $e->getMessage() . '}'; //Liefert gute Fehlermeldung
209     }
210 });
211
```

```
$app->post('/speichernmitarb', function(Request $request, Response $response){
        // get the parameter from the form submit
        $id = $request->getParam('id');
        $bn = $request->getParam('benutzername');
        $email = $request->getParam('email');
        $pwd = $request->getParam('passwort');
        $datum = $request->getParam('datum');
        $startzeit = $request->getParam('startzeit');

$sql = "UPDATE user SET benutzername=:bn, mail=:email, passwort=:pwd, datum=:datum,
                startzeit=:startzeit WHERE iduser=$id";
   try {
      $db = new db();
      $db = $db->connect();
      $stmt = $db->prepare($sql);
                $stmt->bindParam(':bn', $bn);
                $stmt->bindParam(':email', $email);
      $stmt->bindParam(':pwd', $pwd);
                $stmt->bindParam(':datum', $datum);
```

```
    $stmt->bindParam(':startzeit', $startzeit);

    $stmt->execute();

                    $result = array();
                     $result['success'] = true;
                     $result['message'] = 'Registrierung abgeschlossen';
                     echo json_encode($result);
  }
  catch(PDOException $e) {
    echo '{"error": {"msg": ' . $e->getMessage() . '}'; //Liefert gute Fehlermeldung
  }
});
```

9)zur Info: Datenbank:

	iduser	benutzername	mail	passwort	datum	startzeit
en	1	vanessa	berta@gmail.com	1234	2024-02-06	*NULL*
en	4	test	biene@gmx.at	4444	*NULL*	*NULL*

#	Name	Typ	Kollation	A
1	**iduser** 🔑	int(11)		
2	**benutzername**	varchar(50)	utf8mb3_general_ci	
3	**mail**	varchar(50)	utf8mb3_general_ci	
4	**passwort**	varchar(100)	utf8mb3_general_ci	
5	**datum**	date		
6	**startzeit**	time		