

Angular Anwendung mit GPT erstellen

Im folgenden habe ich den Chatverlauf zwischen mir und ChatGPT exportiert. Anweisungen von mir sind *kursiv*. Als JSON habe ich das Beispiel-JSON für einen Patienten von der FHIR-Dokumentationsseite genommen. Es funktioniert mäßig gut mit so großen Datenstrukturen. Mittendrin hört die KI einfach auf Text zu generieren. Auch macht sich der etwas ältere Datenstand bemerkbar. Als ich am Schluss das package.json generiert habe verwendete es Angular 13.

Use the latest Angular Version and build a form with the FormBuilder that is able to represent the JSON I will provide for you. It is an example of a FHIR Patient. For each list in the JSON add a add and remove button. Also please use angular material. Here is the JSON:

```
{
  "resourceType": "Patient",
  "id": "example",
  "text": {
    "status": "generated",
    "div": "html goes here"
  },
  "identifier": [
    {
      "use": "usual",
      "type": {
        "coding": [
          {
            "system": "http://terminology.hl7.org/CodeSystem/v2-0203",
            "code": "MR"
          }
        ],
        "system": "urn:oid:1.2.36.146.595.217.0.1",
        "value": "12345",
        "period": {
          "start": "2001-05-06"
        },
        "assigner": {
          "display": "Acme Healthcare"
        }
      },
      "active": true,
      "name": [
        {
          "use": "official",
          "family": "Chalmers",
          "given": ["Peter", "James"]
        },
        {
          "use": "usual",
          "given": ["Jim"]
        },
        {
          "use": "maiden",
          "family": "Windsor",
          "given": ["Peter", "James"],
          "period": {
            "end": "2002"
          }
        }
      ],
      "telecom": [
        {
          "use": "home",
          "system": "phone",
          "value": "(03) 5555 6473",
          "use": "work",
          "rank": 1
        },
        {
          "system": "phone",
          "value": "(03) 3410 5613",
          "use": "mobile",
          "rank": 2
        },
        {
          "system": "phone",
          "value": "(03) 5555 8834",
          "use": "old",
          "period": {
            "end": "2014"
          }
        }
      ],
      "gender": "male",
      "birthDate": "1974-12-25",
      "_birthDate": {
        "extension": [
          {
            "url": "http://hl7.org/fhir/StructureDefinition/patient-birthTime",
            "valueDateTime": "1974-12-25T14:35:45-05:00"
          }
        ],
        "deceasedBoolean": false
      },
      "address": [
        {
          "use": "home",
          "type": "both",
          "text": "534 Erewhon St PeasantVille, Rainbow, Vic 3999",
          "line": ["534 Erewhon St"],
          "city": "PleasantVille",
          "district": "Rainbow",
          "state": "Vic",
          "postalCode": "3999",
          "period": {
            "start": "1974-12-25"
          },
          "contact": [
            {
              "relationship": [
                {
                  "coding": [
                    {
                      "system": "http://terminology.hl7.org/CodeSystem/v2-0131",
                      "code": "N"
                    }
                  ],
                  "name": {
                    "family": "du Marché",
                    "_family": {
                      "extension": [
                        {
                          "url": "http://hl7.org/fhir/StructureDefinition/humanname-own-prefix",
                          "valueString": "VV"
                        }
                      ],
                      "given": ["Bénédicte"]
                    },
                    "telecom": [
                      {
                        "system": "phone",
                        "value": "+33 (237) 998327"
                      }
                    ],
                    "address": [
                      {
                        "use": "home",
                        "type": "both",
                        "line": ["534 Erewhon St"],
                        "city": "PleasantVille",
                        "district": "Rainbow",
                        "state": "Vic",
                        "postalCode": "3999",
                        "period": {
                          "start": "1974-12-25"
                        },
                        "gender": "female",
                        "period": {
                          "start": "2012"
                        }
                      }
                    ],
                    "managingOrganization": {
                      "reference": "Organization/1"
                    }
                  }
                }
              ]
            }
          ]
        }
      ]
    }
  ]
}
```

To build a form with Angular version 13 and Angular Material, we will need to create a new Angular project and install the necessary dependencies. Afterward, we will create a new component and use the FormBuilder class to create a reactive form that represents the provided JSON object.

Here are the steps to follow:

Create a new Angular project and navigate to the project directory:

```
ng new angular-fhir-form
cd angular-fhir-form
```

Install the required dependencies:

```
npm install @angular/material @angular/cdk @angular/animations
npm install --save-dev @angular-devkit/build-angular
```

Import the BrowserAnimationsModule and MatInputModule modules in the app module:

```
import { BrowserAnimationsModule } from '@angular/platform-browser/animations';
import { MatInputModule } from '@angular/material/input';

@NgModule({
  imports: [
    BrowserAnimationsModule,
    MatInputModule
  ],
  declarations: [AppComponent],
  bootstrap: [AppComponent]
})
export class AppModule {}
```

Create a new component using the Angular CLI:

```
ng generate component patient-form
```

In the patient-form.component.ts file, import the FormBuilder class, the FormGroup and FormArray classes from @angular/forms, and the JSON object:

```
import { Component } from '@angular/core';
import { FormBuilder, FormGroup, FormArray } from '@angular/forms';
import patientData from './patient.json';
```

Create the form using the FormBuilder class:

```
@Component({
  selector: 'app-patient-form',
  templateUrl: './patient-form.component.html',
  styleUrls: ['./patient-form.component.css']
})
```

```

}))
export class PatientFormComponent {
  patientForm: FormGroup;

  constructor(private fb: FormBuilder) {
    this.patientForm = this.fb.group({
      id: [patientData.id],
      identifier: this.fb.array([]),
      active: [patientData.active],
      name: this.fb.array([]),
      telecom: this.fb.array([]),
      gender: [patientData.gender],
      birthDate: [patientData.birthDate],
      deceasedBoolean: [patientData.deceasedBoolean],
      address: this.fb.array([]),
      contact: this.fb.array([]),
      managingOrganization: [patientData.managingOrganization.reference]
    });
  }
}

```

Add the FormArray for the identifier, name, telecom, address, and contact fields and set their initial values:

```

constructor(private fb: FormBuilder) {
  this.patientForm = this.fb.group({
    id: [patientData.id],
    identifier: this.fb.array([
      this.fb.group({
        use: [patientData.identifier[0].use],
        type: this.fb.group({
          coding: this.fb.array([
            this.fb.group({
              system: [patientData.identifier[0].type.coding[0].system],
              code: [patientData.identifier[0].type.coding[0].code]
            })
          ])
        })
      ]),
      system: [patientData.identifier[0].system],
      value: [patientData.identifier[0].value],
      period: this.fb.group({
        start: [patientData.identifier[0].period.start]
      }),
      assigner: this.fb.group({
        display: [patientData.identifier[0].assigner.display]
      })
    ])
  ),
  active: [patientData.active],
  name: this.fb.array([
    this.fb.group({
      use: [patientData.name[0].use],
      family: [patientData.name[0].family],

```

```
given: this.fb.array([
  [patientData.name[0].given[
```

continue

Now that we have our Angular project set up and the necessary dependencies installed, we can start building our form using the FormBuilder and Angular Material.

First, let's create a new component for our form. Run the following command in your terminal to generate a new component named "patient-form":

```
ng generate component patient-form
```

Next, let's open up the patient-form.component.ts file and start by importing the necessary modules and defining our patient form:

```
import { Component, OnInit } from '@angular/core';
import { FormBuilder, FormGroup, FormArray } from '@angular/forms';
import { MatFormFieldModule } from '@angular/material/form-field';
import { MatInputModule } from '@angular/material/input';
import { MatButtonModule } from '@angular/material/button';

@Component({
  selector: 'app-patient-form',
  templateUrl: './patient-form.component.html',
  styleUrls: ['./patient-form.component.css']
})
export class PatientFormComponent implements OnInit {

  patientForm: FormGroup;

  constructor(private fb: FormBuilder) { }

  ngOnInit() {
    this.patientForm = this.fb.group({
      id: [''],
      active: [''],
      name: this.fb.array([]),
      identifier: this.fb.array([]),
      telecom: this.fb.array([]),
      gender: [''],
      birthDate: [''],
      deceasedBoolean: [''],
      address: this.fb.array([]),
      contact: this.fb.array([]),
      managingOrganization: ['']
    });
  }
}
```

```
}
```

Here we have defined our patientForm as a FormGroup, and included FormArrays for the name, identifier, telecom, address, and contact fields, since each of these fields can have multiple values in the FHIR Patient resource.

Next, let's add our HTML template to the patient-form.component.html file:

```
<form [formGroup]="patientForm">
  <mat-form-field>
    <mat-label>ID</mat-label>
    <input matInput formControlName="id">
  </mat-form-field>

  <mat-form-field>
    <mat-label>Active</mat-label>
    <input matInput formControlName="active">
  </mat-form-field>

  <div formArrayName="name">
    <div *ngFor="let name of patientForm.get('name').controls; let i = index">
      <div [formGroupName]="i">
        <mat-form-field>
          <mat-label>Use</mat-label>
          <input matInput formControlName="use">
        </mat-form-field>

        <mat-form-field>
          <mat-label>Family</mat-label>
          <input matInput formControlName="family">
        </mat-form-field>

        <mat-form-field>
          <mat-label>Given</mat-label>
          <div formArrayName="given">
            <div *ngFor="let given of name.get('given').controls; let j = index">
              <input matInput [formControlName]="j">
              <button mat-icon-button color="warn" *ngIf="j != 0"
(click)="removeGiven(i, j)">
                <mat-icon>remove_circle</mat-icon>
              </button>
              <button mat-icon-button color="primary" *ngIf="j ==
name.get('given').length - 1" (click)="addGiven(i)">
                <mat-icon>add_circle</mat-icon>
              </button>
            </div>
          </div>
        </mat-form-field>

        <button mat-icon-button color="warn" *ngIf="i != 0"
```

```
(click)="removeName(i)">
    <mat-icon>remove_circle</mat-icon>
  </button>
</div>
</div>
<button mat-icon-button color="primary" (click
```

whats the contents of the package.json?

The contents of the package.json file typically include metadata about a Node.js project, as well as a list of its dependencies and scripts that can be run to perform various tasks.

Here is an example of what a package.json file might look like for an Angular project:

```
{
  "name": "my-app",
  "version": "0.0.1",
  "scripts": {
    "ng": "ng",
    "start": "ng serve",
    "build": "ng build",
    "test": "ng test",
    "lint": "ng lint",
    "e2e": "ng e2e"
  },
  "dependencies": {
    "@angular/animations": "~13.1.0",
    "@angular/common": "~13.1.0",
    "@angular/compiler": "~13.1.0",
    "@angular/core": "~13.1.0",
    "@angular/forms": "~13.1.0",
    "@angular/platform-browser": "~13.1.0",
    "@angular/platform-browser-dynamic": "~13.1.0",
    "@angular/router": "~13.1.0",
    "@fortawesome/fontawesome-free": "^5.15.4",
    "rxjs": "~7.5.3",
    "tslib": "^2.3.1",
    "zone.js": "~0.11.4"
  },
  "devDependencies": {
    "@angular-devkit/build-angular": "~13.1.0",
    "@angular/cli": "~13.1.0",
    "@angular/compiler-cli": "~13.1.0",
    "@types/jasmine": "~3.10.0",
    "@types/node": "^16.11.15",
    "codemlizer": "^6.0.2",
    "jasmine-core": "~3.10.0",
    "jasmine-spec-reporter": "~7.0.0",
    "karma": "~6.3.8",
    "karma-chrome-launcher": "~3.1.0",
    "karma-coverage": "~2.1.0",
    "karma-jasmine": "~4.0.0",
```

```
"karma-jasmine-html-reporter": "^1.7.0",  
"protractor": "~7.0.0",  
"ts-node": "~9.1.1",  
"tslint": "~6.1.3",  
"typescript": "~4.5.4"  
}  
}
```

This file specifies various dependencies that are required for the project, as well as scripts that can be run with the npm command to start the development server, build the project, run tests, and more.