

Bab 6

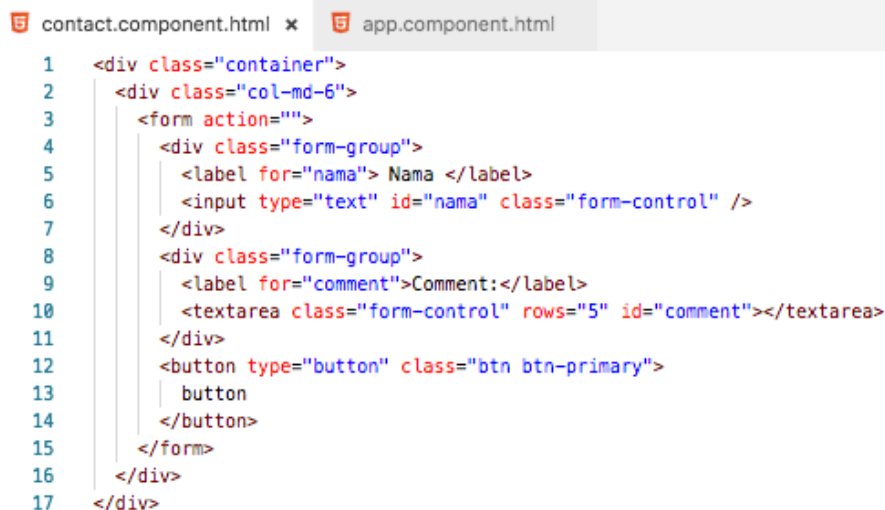
Template-Driven Forms

Pada bab ini kita akan membahas penggunaan form dengan berbagai macam inputan, menampilkan validasi error, dan perbedaan antara template-driven forms dengan reactive form.

6.1 Membuat Form Bootstrap

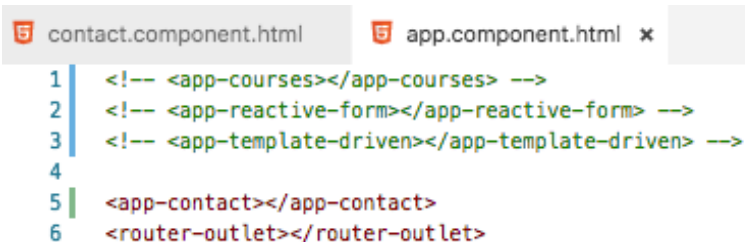
Pada sub bab ini kita akan membuat sebuah form dengan menggunakan bootstrap. Untuk lebih jelasnya ikuti langkah berikut

- buatlah sebuah component dengan nama contact
- buka file contact.component.html dan tambahkan code berikut:



```
1 <div class="container">
2   <div class="col-md-6">
3     <form action="">
4       <div class="form-group">
5         <label for="nama"> Nama </label>
6         <input type="text" id="nama" class="form-control" />
7       </div>
8       <div class="form-group">
9         <label for="comment">Comment:</label>
10        <textarea class="form-control" rows="5" id="comment"></textarea>
11      </div>
12      <button type="button" class="btn btn-primary">
13        button
14      </button>
15    </form>
16  </div>
17 </div>
```

- buka file app.component.html dan tambahkan code berikut



```
1 <!-- <app-courses></app-courses> -->
2 <!-- <app-reactive-form></app-reactive-form> -->
3 <!-- <app-template-driven></app-template-driven> -->
4
5 <app-contact></app-contact>
6 <router-outlet></router-outlet>
```

- jalankan servernya maka hasilnya seperti berikut

6.2 Macam-Macam Form

Dalam penggunaan form pada angular dapat menggunakan template-driven dan reactive form. perbedaan antara template-driven dan reactive form adalah pada reactive form biasanya juga disebut model-driven, penggunaan reactive form menghindari penggunaan directive seperti ngModel, required dsb. Pada reactive form ini menggunakan reactive-pattern, testing dan validasi yang bersifat synchronous. Untuk lebih jelasnya kita buat sebuah contoh menggunakan reactive form.

- generate 2 buah component dengan nama reactive-form dan template-driven
- buka file reactive-form.component.html dan buatlah code seperti berikut

```

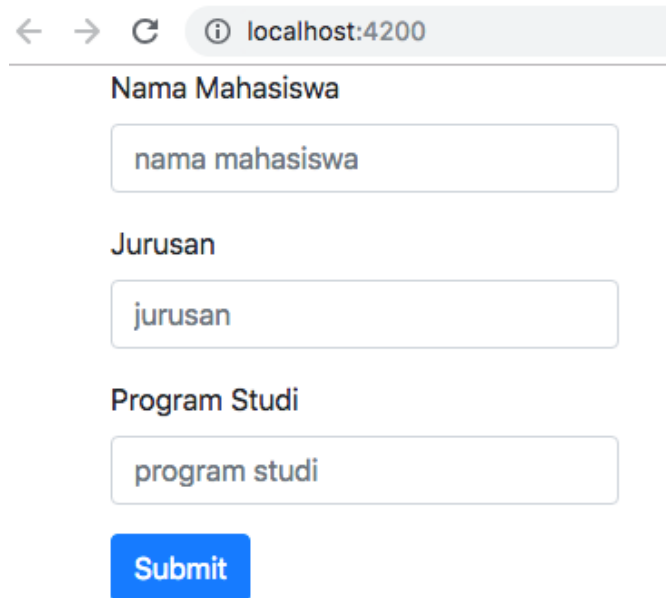
1  <div class="container">
2      <div class="col-md-4">
3          <form novalidate >
4              <div class="form-group">
5                  <label for="nama_mahasiswa">Nama Mahasiswa</label>
6                  <input
7                      type="text"
8                      class="form-control"
9                      id="nama_mahasiswa"
10                     aria-describedby="namamhs"
11                     placeholder="nama mahasiswa"
12                 />
13             </div>
14             <div>
15                 <div class="form-group">
16                     <label for="jurusan">Jurusan</label>
17                     <input
18                         type="text"
19                         class="form-control"
20                         id="nama_jurusan"
21                         placeholder="jurusan"
22                     />
23                 </div>
24                 <div class="form-group">
25                     <label for="ProgramStudi">Program Studi</label>
26                     <input
27                         type="text"
28                         class="form-control"
29                         id="nama_program_studi"
30                         placeholder="program studi"
31                     />
32                 </div>
33             </div>
34             <button type="submit" class="btn btn-primary">Submit</button>
35         </form>
36     </div>
37 </div>

```

- buka file app.component.html dan tambahkan code berikut

```
app.component.html x reactive-form.component.html
1 <!-- <app-courses></app-courses> -->
2 <app-reactive-form></app-reactive-form>
3 <router-outlet></router-outlet>
```

- sehingga hasilnya seperti berikut



A screenshot of a web browser at localhost:4200. The browser window displays a form with three input fields and a submit button. The first input field is labeled 'Nama Mahasiswa' and contains the text 'nama mahasiswa'. The second input field is labeled 'Jurusan' and contains the text 'jurusan'. The third input field is labeled 'Program Studi' and contains the text 'program studi'. Below the input fields is a blue button labeled 'Submit'.

- langkah berikutnya buatlah sebuah interface dengan nama mahasiswa.interface.ts dan tambahkan code berikut

```
TS mahasiswa.interface.ts x app.component.html
1 export interface mahasiswa{
2   nama_mhs:string;
3   pendidikan:{
4     nama_jurusan:string;
5     nama_prodi:string;
6   }
7 }
```

- Dikarenakan untuk menggunakan reactive-forms diperlukan FormBuilder dan FormGroup dari module ReactiveFormsModule maka kita harus tambahkan ReactiveFormsModule dan FormGroup pada app.module.ts seperti pada gambar berikut

```

TS mahasiswa.interface.ts  app.module.ts x  app.component.html  reactive-form.comp
1  import { BrowserModule } from '@angular/platform-browser';
2  import { NgModule } from '@angular/core';
3  import { ReactiveFormsModule } from '@angular/forms';
4
5  import { AppRoutingModule } from './app-routing.module';
6  import { AppComponent } from './app.component';
7  import { ReactiveFormComponent } from './reactive-form/reactive-form.component';
8  import { TemplateDrivenComponent } from './template-driven/template-driven.component';
9
10 @NgModule({
11   declarations: [
12     AppComponent,
13     ReactiveFormComponent,
14     TemplateDrivenComponent
15   ],
16   imports: [
17     BrowserModule,
18     AppRoutingModule,
19     ReactiveFormsModule
20   ],
21   providers: [],
22   bootstrap: [AppComponent]
23 })
24 export class AppModule { }
25

```

- buka file reactive-form.component.ts, import terlebih dahulu FormBuilder dan FormGroup dan tambahkan beberapa code berikut

```

TS mahasiswa.interface.ts  reactive-form.component.ts x  app.component.html
1  import { FormGroup, FormBuilder } from '@angular/forms';
2  import { mahasiswa } from '../mahasiswa.interface';
3  import { Component, OnInit } from '@angular/core';
4
5
6
7  @Component({
8    selector: 'app-reactive-form',
9    templateUrl: './reactive-form.component.html',
10   styleUrls: ['./reactive-form.component.css']
11 })
12 export class ReactiveFormComponent implements OnInit {
13
14   mahasiswa: FormGroup;
15
16   constructor(private mhs: FormBuilder) {
17     {
18       this.createForm();
19     }
20
21     ngOnInit() {
22     }
23
24     // method untuk generate form
25     createForm(){
26       this.mahasiswa= this.mhs.group({
27         nama_mhs:'',
28         pendidikan:this.mhs.group({
29           nama_jurusan:'',
30           nama_prodi:''
31         })
32       })
33     }
34
35     onSubmit(f:FormGroup){
36       console.log("hasil dari inputan adalah");
37       console.log(f.value);
38     }
39   }

```

Penjelasan code

- line 14 adalah sebuah property yang merupakan sebuah FormGroup
- line 16-19 pada constructor terdapat parameter mhs yang merupakan FormBuilder dan dimana didalam constructor tersebut terdapat method untuk generate form
- line 25-33 adalah method yang digunakan untuk mengenerate form yang sesuai dengan struktur form

```

1  export interface mahasiswa { //FormGroup
2      nama_mhs:string; //FormControl
3      pendidikan:{ //FormGroup
4          nama_jurusan:string; //FormControl
5          nama_prodi:string; //FormControl
6      }
7  }

```

- line 35-38 method yang digunakan untuk menampilkan isi dari inputan user ke dalam console
- buka file reactive-form.component.html dan modifikasi codenya menjadi seperti berikut



```

1  <div class="container">
2      <div class="col-md-4">
3          <form novalidate [formGroup]="mahasiswa" (ngSubmit)="onSubmit(mahasiswa)">
4              <div class="form-group">
5                  <label for="nama_mahasiswa">Nama Mahasiswa</label>
6                  <input
7                      type="text"
8                      formControlName="nama_mhs"
9                      class="form-control"
10                     id="nama_mahasiswa"
11                     aria-describedby="namamhs"
12                     placeholder="nama mahasiswa"
13                 />
14             </div>
15             <div formGroupName="pendidikan">
16                 <div class="form-group">
17                     <label for="jurusan">Jurusan</label>
18                     <input
19                         type="text"
20                         formControlName="nama_jurusan"
21                         class="form-control"
22                         id="nama_jurusan"
23                         placeholder="jurusan"
24                     />
25                 </div>
26                 <div class="form-group">
27                     <label for="ProgramStudi">Program Studi</label>
28                     <input
29                         type="text"
30                         formControlName="nama_prodi"
31                         class="form-control"
32                         id="nama_program_studi"
33                         placeholder="program studi"
34                     />
35                 </div>
36             </div>
37             <button type="submit" class="btn btn-primary">Submit</button>
38         </form>
39     </div>
40 </div>

```

penjelasan code

- pada line 3 dan 15 terdapat FormGroup yaitu mahasiswa dan pendidikan seperti pada mahasiswa.interface.ts

```

1  export interface mahasiswa { //FormGroup
2      nama_mhs:string; //FormControl
3      pendidikan:{ //FormGroup
4          nama_jurusan:string; //FormControl
5          nama_prodi:string; //FormControl
6      }
7  }

```

- pada line 3 terdapat event-binding yaitu ngSubmit dimana jika button submit diklik maka method onSubmit dengan parameter mahasiswa pada reactive-form.component.ts akan berjalan
- line 9, 20, 30 adalah FormControlName sesuaikan dengan interface mahasiswa
- jalan server localhost maka hasilnya seperti berikut

The screenshot shows a web browser window with the title 'BelajarAngular' and the address 'localhost:4200'. The form contains three text inputs: 'Nama Mahasiswa' (filled with 'DIKA'), 'Jurusan' (filled with 'Informatika'), and 'Program Studi' (filled with 'Teknik Informatika'). A blue 'Submit' button is located below the inputs. Below the browser window, the Chrome DevTools Console is open, displaying the following JSON object:

```

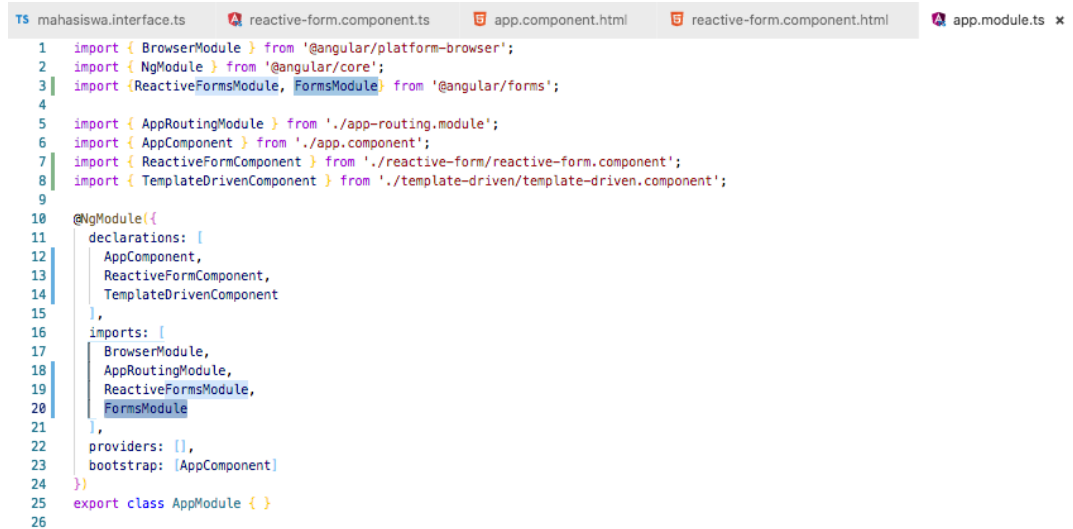
{
  nama_mhs: "DIKA",
  pendidikan: {
    nama_jurusan: "Informatika",
    nama_prodi: "Teknik Informatika"
  }
}

```

The console also shows a message: 'Angular is running in the development mode. Call enableProdMode() to enable'.

Percobaan diatas adalah penggunaan form menggunakan reactive-form untuk berikutnya kita akan membuat form menggunakan template-driven. Untuk lebih jelas ikuti langkah berikut ini:

- untuk template-driven kita membutuhkan FormsModule pada app.module.ts, jadi buka file app.module.ts dan tambahkan FormsModule

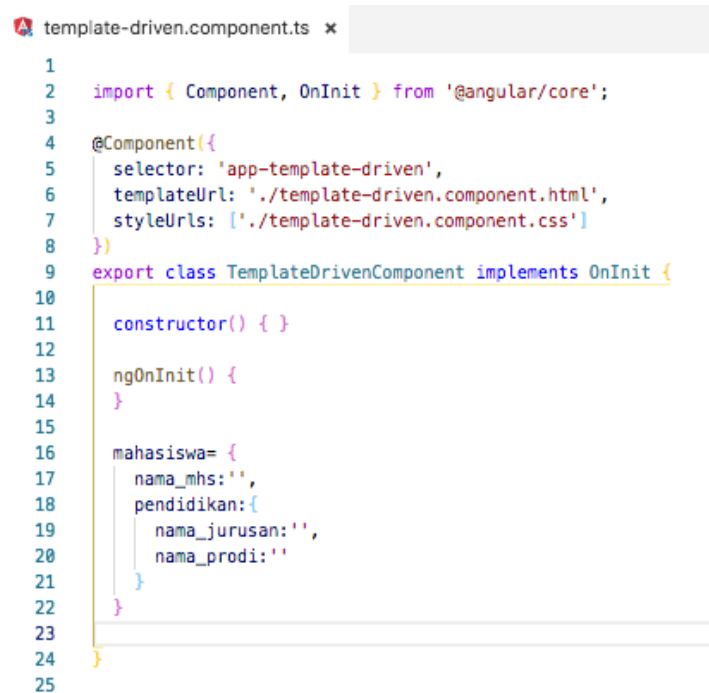


```

1 import { BrowserModule } from '@angular/platform-browser';
2 import { NgModule } from '@angular/core';
3 import { ReactiveFormsModule, FormsModule } from '@angular/forms';
4
5 import { AppRoutingModule } from './app-routing.module';
6 import { AppComponent } from './app.component';
7 import { ReactiveFormComponent } from './reactive-form/reactive-form.component';
8 import { TemplateDrivenComponent } from './template-driven/template-driven.component';
9
10 @NgModule({
11   declarations: [
12     AppComponent,
13     ReactiveFormComponent,
14     TemplateDrivenComponent
15   ],
16   imports: [
17     BrowserModule,
18     AppRoutingModule,
19     ReactiveFormsModule,
20     FormsModule
21   ],
22   providers: [],
23   bootstrap: [AppComponent]
24 })
25 export class AppModule {}
26

```

- buka file template-driven.component.ts dan tambahkan struktur form seperti berikut



```

1
2 import { Component, OnInit } from '@angular/core';
3
4 @Component({
5   selector: 'app-template-driven',
6   templateUrl: './template-driven.component.html',
7   styleUrls: ['./template-driven.component.css']
8 })
9 export class TemplateDrivenComponent implements OnInit {
10
11   constructor() {}
12
13   ngOnInit() {}
14
15   mahasiswa = {
16     nama_mhs: '',
17     pendidikan: {
18       nama_jurusan: '',
19       nama_prodi: ''
20     }
21   }
22 }
23
24
25

```

- buka file template-driven.component.html dan tambahkan code berikut

```
template-driven.component.ts  template-driven.component.html x  app.component.html
1 <div class="container">
2   <div class="col-md-4">
3     <form novalidate (ngSubmit)="onSubmit()">
4       <div class="form-group">
5         <label for="nama_mahasiswa">Nama Mahasiswa</label>
6         <input
7           type="text"
8           [(ngModel)]="mahasiswa.nama_mhs"
9           [ngModelOptions]="{standalone:true}"
10          class="form-control"
11          id="nama_mahasiswa"
12          aria-describedby="namamhs"
13          placeholder="nama mahasiswa"
14        />
15      </div>
16      <div>
17        <div class="form-group">
18          <label for="jurusan">Jurusan</label>
19          <input
20            type="text"
21            [(ngModel)]="mahasiswa.pendidikan.nama_jurusan"
22          />
23          [ngModelOptions]="{standalone:true}"
24          class="form-control"
25          id="nama_jurusan"
26          placeholder="jurusan"
27        />
28      </div>
29      <div class="form-group">
30        <label for="ProgramStudi">Program Studi</label>
31        <input
32          type="text"
33          [(ngModel)]="mahasiswa.pendidikan.nama_prodi"
34        />
35        [ngModelOptions]="{standalone:true}"
36        class="form-control"
37        id="nama_program_studi"
38        placeholder="program studi"
39      />
40      </div>
41    </div>
42    <button type="submit" class="btn btn-primary">Submit</button>
43  </form>
44 </div>
45 </div>
46
```

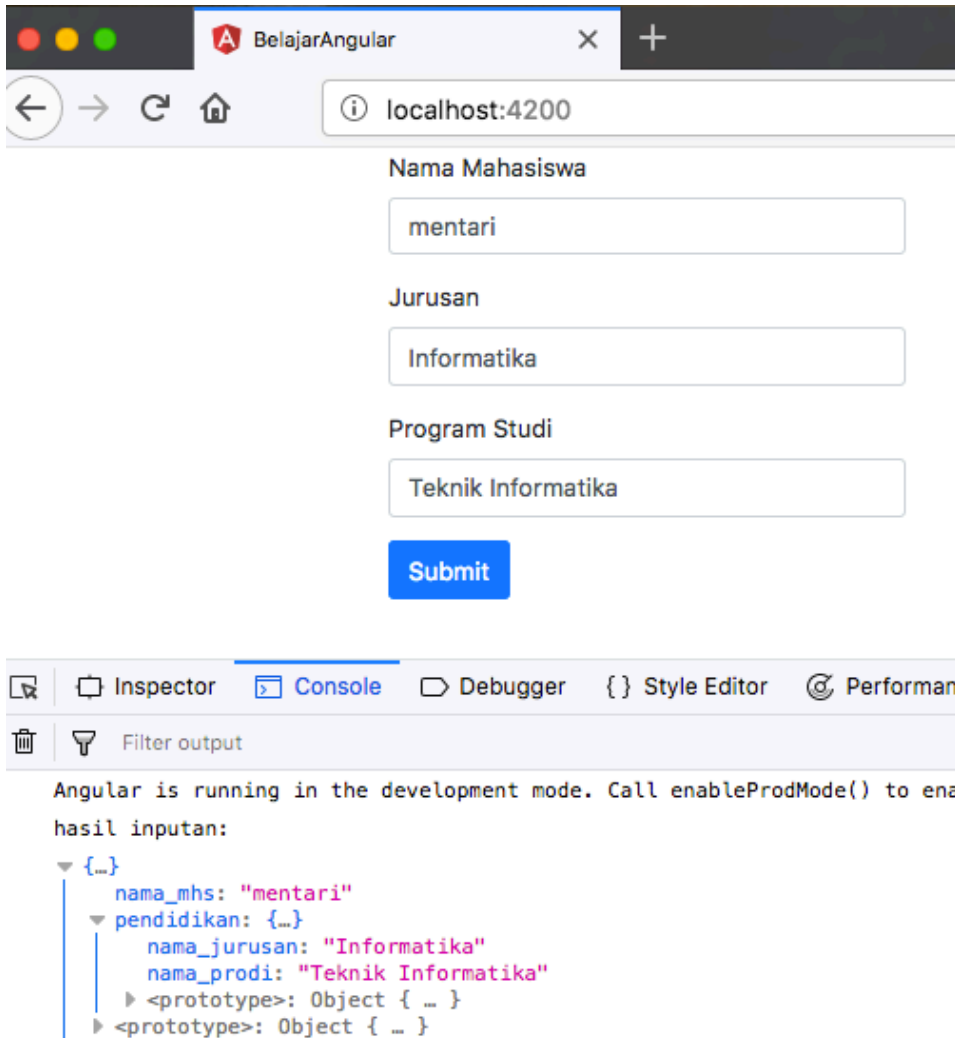
- tambahkan method onSubmit pada template-driven.component.ts

```
template-driven.component.ts x
21   }
22 }
23
24 onSubmit(){
25   console.log("hasil inputan:");
26   console.log(this.mahasiswa);
27 }
28
29 }
```

- buka file app.component.html tambahkan code berikut


```
template-driven.component.ts | template-driven.component.html | app.component.html x
1 <!-- <app-courses></app-courses> -->
2 <!-- <app-reactive-form></app-reactive-form> -->
3 <app-template-driven></app-template-driven>
4 <router-outlet></router-outlet>
```

- hasil



jika kita lihat hasilnya sama tetapi penggunaan directivenya yang berbeda

6.3 ngModel

ngModel adalah sebuah directive untuk binding data atau value ke variable, semisalnya terdapat kode seperti berikut :

- `[(ngModel)]="nama"`

berarti value/nilai yang ada pada input yang memiliki attribute tersebut akan masuk kedalam variable nama. Sebagai pembuktian lakukan beberapa percobaan berikut :

- buka file contact-form.component.html dan tambahkan ngmodel seperti berikut

```

1 <div class="container">
2   <div class="col-md-6">
3     <form action="">
4       <div class="form-group">
5         <label for="nama"> Nama </label>
6         <input type="text" id="nama" class="form-control" [(ngModel)]="nama" />
7       </div>
8       <div class="form-group">
9         <label for="comment">Comment:</label>
10        <textarea class="form-control" rows="5" id="comment" [(ngModel)]="comment"></textarea>
11      </div>
12      <button type="button" class="btn btn-primary">
13        button
14      </button>
15    </form>
16  </div>
17 </div>
18

```

- buka file contact-form.component.ts dan tambahkan decorator input (line 9 dan 10)

```

1 import { Component, OnInit, Input } from '@angular/core';
2
3 @Component({
4   selector: 'app-contact',
5   templateUrl: './contact.component.html',
6   styleUrls: ['./contact.component.css']
7 })
8 export class ContactComponent implements OnInit {
9   @Input() nama:string;
10  @Input() comment:string;
11
12  constructor() { }
13
14  ngOnInit() {
15  }
16
17 }
18

```

- coba jalan servernya dan lihat terdapat pesan error seperti pada gambar berikut

```

✖ ERROR Error: If ngModel is used within a form tag, either the name attribute must be set
or the form
control must be defined as 'standalone' in ngModelOptions.
    at Function.push../node_modules/@angular/forms/fesm5/forms.js.TemplateDrivenErrors.missingNameException (forms.
js:4498)
    at NgModel.push../node_modules/@angular/forms/fesm5/forms.js.NgModel._checkName (forms.js:4867)
    at NgModel.push../node_modules/@angular/forms/fesm5/forms.js.NgModel._checkForErrors (forms.js:4852)
    at NgModel.push../node_modules/@angular/forms/fesm5/forms.js.NgModel.ngOnChanges (forms.js:4760)
    at checkAndUpdateDirectiveInline (core.js:21997)
    at checkAndUpdateNodeInline (core.js:23265)
    at checkAndUpdateNode (core.js:23227)
    at debugCheckAndUpdateNode (core.js:23861)
    at debugCheckDirectivesFn (core.js:23821)
    at Object.eval [as updateDirectives] (ContactComponent.html:6)

```

penjelasan error : hal ini disebabkan jika kita menggunakan ngModel maka kita harus menambahkan atribut name pada tag tersebut

- tambahkan atribut name pada tag input

```

1 <div class="container">
2   <div class="col-md-6">
3     <form action="">
4       <div class="form-group">
5         <label for="nama"> Nama </label>
6         <input type="text" id="nama" class="form-control" [(ngModel)]="nama" name="nama" />
7       </div>
8       <div class="form-group">
9         <label for="comment">Comment:</label>
10        <textarea class="form-control" rows="5" id="comment" [(ngModel)]="comment" name="comment"></textarea>
11      </div>
12      <button type="button" class="btn btn-primary">
13        button
14      </button>
15      <!-- menampilkan hasil inputan -->
16      <p> Hasil Inputan:</p>
17      <h1>Nama = {{nama}}</h1>
18      <h1>Comment = {{comment}}</h1>
19    </form>
20  </div>
21 </div>

```

- jalankan servernya (cat:tidak perlu diklik button untuk menampilkan data karena tidak menggunakan event binding jadi inputan akan langsung tampil saat user memasukan data)

The screenshot shows a web browser window titled 'BelajarAngular' at 'localhost:4200'. The form contains a text input for 'Nama' with the value 'mentari' and a text area for 'Comment:' with the value 'belajar angular'. Below the inputs is a blue 'button'. Underneath the button, the text 'Hasil Inputan:' is followed by two lines of output: 'Nama = mentari' and 'Comment = belajar angular'.

kita juga dapat melihat properties pada ngModel, Untuk lebih jelasnya ikuti langkah bertikut

- buka file contact.component.html modifikasi code berikut ini:

penjelasan code :

- line 31 ngModel adalah directive
- line 32 #firstname="ngModel" adalah property dari ngModel atau template variable (**#namavariabel**)
- line 33 (change)="log(firstName)" adalah untuk melakukan perubahan pada method log dengan parameter firstName

```
contact.component.html x contact.component.ts app.component.html
22 <div class="container">
23   <div class="col-md-6">
24     <form action="">
25       <div class="form-group">
26         <label for="nama"> Nama </label>
27         <input
28           type="text"
29           id="nama"
30           class="form-control"
31           ngModel
32           #firstName="ngModel"
33           (change)="log(firstName)"
34           name="nama"
35         />
36       </div>
37       <div class="form-group">
38         <label for="comment">Comment:</label>
39         <textarea
40           class="form-control"
41           rows="5"
42           id="comment"
43           name="comment"
44         ></textarea>
45       </div>
46       <button type="button" class="btn btn-primary">
47         button
48       </button>
49     </form>
50   </div>
51 </div>
```

Pada textarea tambahkan juga ngModel seperti pada gambar berikut line 43

```
37 <div class="form-group">
38   <label for="comment">Comment:</label>
39   <textarea
40     class="form-control"
41     rows="5"
42     id="comment"
43     ngModel
44     name="comment"
45   ></textarea>
46 </div>
```

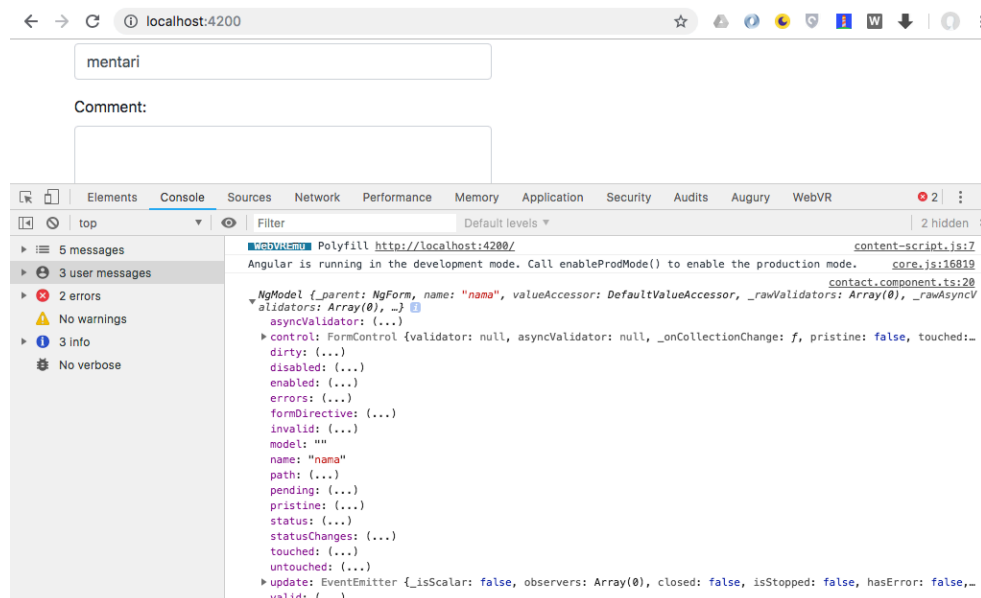
- buka file contact.component.ts modifikasi code berikut ini

```

1  import { Component, OnInit, Input } from "@angular/core";
2
3  @Component({
4    selector: "app-contact",
5    templateUrl: "../contact.component.html",
6    styleUrls: ["../contact.component.css"]
7  })
8  // export class ContactComponent implements OnInit {
9  //   @Input() nama:string;
10  //   @Input() comment:string;
11
12  //   constructor() { }
13
14  //   ngOnInit() {
15  //   }
16
17  // }
18  export class ContactComponent {
19    log(x) {
20      console.log(x);
21    }
22  }

```

- jalankan server dan lihat pada inspect maka akan muncul property dari ngModel



Penjelasan gambar

- jika kita click pada (...) pada control maka akan muncul FormControl dan dimana terdapat instance dari FormControl Class di angular. Sebagai contoh ada property dirty yang digunakan untuk menentukan jika nilai dari inputan field berubah. jika kita click maka nilainya true hal ini dikarenakan kita memasukkan value di input field dan ada perubahan inputan sehingga dirty bernilai true
- sedangkan kebalikan dari property dirty adalah pristine dan bernilai false

6.4 Validasi

Pada sub bab ini kita akan membuat sebuah validasi dengan menggunakan bantuan directive, untuk lebih jelasnya ikuti langkah berikut

- buka file contact.component.html dan tambahkan code berikut

```
contact.component.html x contact.component.ts app.component.html
25 <div class="form-group">
26   <label for="nama"> Nama </label>
27   <input
28     type="text"
29     id="nama"
30     class="form-control"
31     required
32     ngModel
33     #firstName="ngModel"
34     (change)="log(firstName)"
35     name="nama"
36   />
37   <div class="alert alert-danger" *ngIf="!firstName.valid">Nama harus diisi</div>
--
```

Penjelasan code

- line 31 required berarti harus diisi
- line 37 adalah alert, jika kondisi menggunakan directive ngif dimana jika firstName bernilai false atau dengan kata lain tidak ada isinya maka akan muncul alert.
- valid adalah sebuah property pada ngModel seperti pada gambar berikut (cat : nilai valid berupa true dikarenakan ada isinya pada inputan textfield)

```
▼ NgModel {_parent: NgForm, name: "nama", valueAccessor: DefaultValueAccessor, _rawValidators: Array(1), _rawAsyncValidators:
  asyncValidator: (...)}
  ▶ control: FormControl {validator: f, asyncValidator: null, _onCollectionChange: f, pristine: false, touched: true, ...}
    dirty: (...)
    disabled: (...)
    enabled: (...)
    errors: (...)
    formDirective: (...)
    invalid: (...)
    model: ""
    name: "nama"
    path: (...)
    pending: (...)
    pristine: (...)
    status: (...)
    statusChanges: (...)
    touched: (...)
    untouched: (...)
  ▶ update: EventEmitter {_isScalar: false, observers: Array(0), closed: false, isStopped: false, hasError: false, ...}
    valid: true
```

- jalankan localhost dan lihat hasilnya

Nama

Nama harus diisi

Comment:

button

- jika kita jalankan maka secara default alert akan muncul. Untuk itu kita membutuhkan sebuah kondisi lagi. Buka file `contact.component.html` dan tambahkan code berikut (line 39)

```
24 <form action="">
25   <div class="form-group">
26     <label for="nama"> Nama </label>
27     <input
28       type="text"
29       id="nama"
30       class="form-control"
31       required
32       ngModel
33       #firstName="ngModel"
34       (change)="log(firstName)"
35       name="nama"
36     />
37   <div
38     class="alert alert-danger"
39     *ngIf="firstName.touched && !firstName.valid"
40   >
41     Nama harus diisi
42   </div>
43 </div>
```

Penjelasan code:

- pada line 39 terdapat directive `ngIf` terdapat 2 kondisi yaitu `touched` dan `valid`, jika kedua kondisi ini tidak terpenuhi maka akan muncul alert
- jalankan localhost pada saat dijalankan maka secara default alert tidak akan muncul dan pada saat field nama dikosongkan maka akan muncul alert “Nama harus diisi”

<p>Nama</p> <input type="text"/> <p>Comment:</p> <div><div></div></div> <p>button</p>
<p>Nama</p> <div><div></div><div>Nama harus diisi</div></div> <p>Comment:</p> <div><div></div></div> <p>button</p>

6.5 spesifik validasi error

Ada beberapa macam untuk validasi error seperti minlength, maxlength, pattern dan required. Untuk lebih jelasnya ikuti langkah-langkah berikut :

- buka file conctact.component.html dan tambahkan beberapa code berikut


```
contact.component.html x contact.component.ts app.component.html
25 <div class="form-group">
26 <label for="nama"> Nama </label>
27 <input
28   type="text"
29   id="nama"
30   class="form-control"
31   required
32   minlength="3"
33   maxlength="10"
34   pattern="pisang"
35   ngModel
36   #firstName="ngModel"
37   (change)="log(firstName)"
38   name="nama"
39 />
40 <div
41   class="alert alert-danger"
42   *ngIf="firstName.touched && !firstName.valid"
43 >
44   <div *ngIf="firstName.errors['required']">Nama harus diisi</div>
45
46   <div *ngIf="firstName.errors['minlength']">nama min 3 kata</div>
47   <div *ngIf="firstName.errors['pattern']">
48     nama tidak sesuai dengan pattern
49   </div>
50 </div>
51 </div>
```

penjelasan code :

- line 32 minlength adalah min abjad yang harus diisi
- line 33 maxlenth adalah max panjang abjad
- line 34 pattern adalah yang harus diisi
- line 44 ngif (error required) adalah sebuah kondisi jika line 42 tidak terpenuhi
- line 45 ngif (error minlength) adalah sebuah kondisi dimana nama kurang dari 3 abjad
- linr 47 ngif (error pattern) adalah sebuah kondisi dimana jika inputan user tidak sesuai dengan patternnya
- jalankan localhost jika kita masukkan angka 1

Nama

1

nama min 3 kata
nama tidak sesuai dengan pattern

Comment:

button

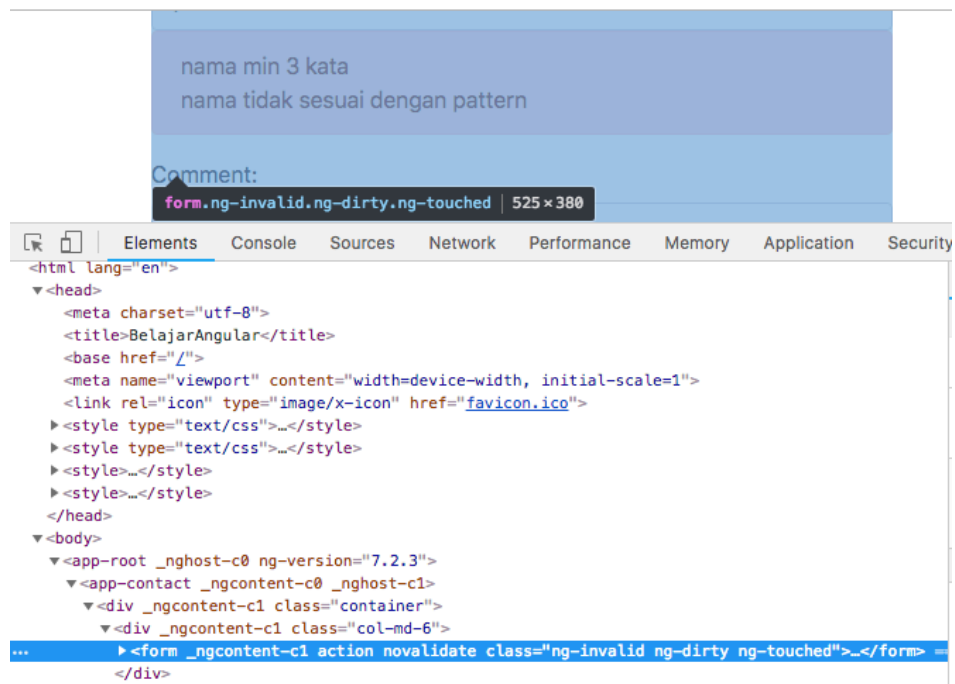
jika kita inspect element maka pada bagian property errors terdapat jenis-jenis validasi yang digunakan seperti pada gambar berikut

```
▼ NgModel {_parent: NgForm, name: "nama", valueAccessor: DefaultValueAccessor, .
  asyncValidator: (...)}
  ▼ control: FormControl
    asyncValidator: null
    dirty: (...)
    disabled: (...)
    enabled: (...)
    ▼ errors:
      ► minlength: {requiredLength: 3, actualLength: 1}
      ► pattern: {requiredPattern: "^pisang$", actualValue: "1"}
      ► __proto__: Object
```

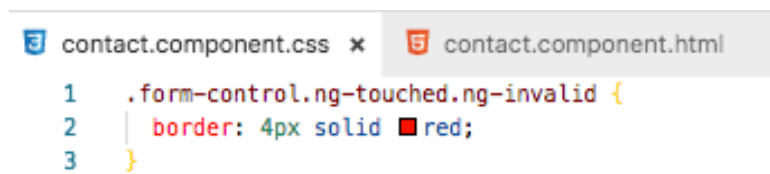
6.6 styling Invalid input field

Kita dapat menambahkan sebuah style pada input fieldnya sebagai contoh jika terdapat error maka selain menampilkan. Untuk lebih jelasnya ikuti langkah berikut

- jika kita inspect element pada tab element dan kita click pada bagian form validation, maka pada element akan menunjukkan sebuah code angular ng-content class ng-invalid, ng-dirty dan ng-touched.
- pada sub bab ini kita akan memodifikasi agar selain muncul alert pada form input akan merah juga



- buka file contact.component.css dan tambahkan code berikut



- jalankan localhost jika kita inputkan salah

Nama

nama min 3 kata
nama tidak sesuai dengan pattern

Comment:

button

6.7 ngForm

ngForm adalah directive Angular yang penting untuk membuat template-driven forms. Mari kita mulai dengan ngForm dulu. Berikut adalah kutipan tentang ngForm dari dokumentasi Angular.

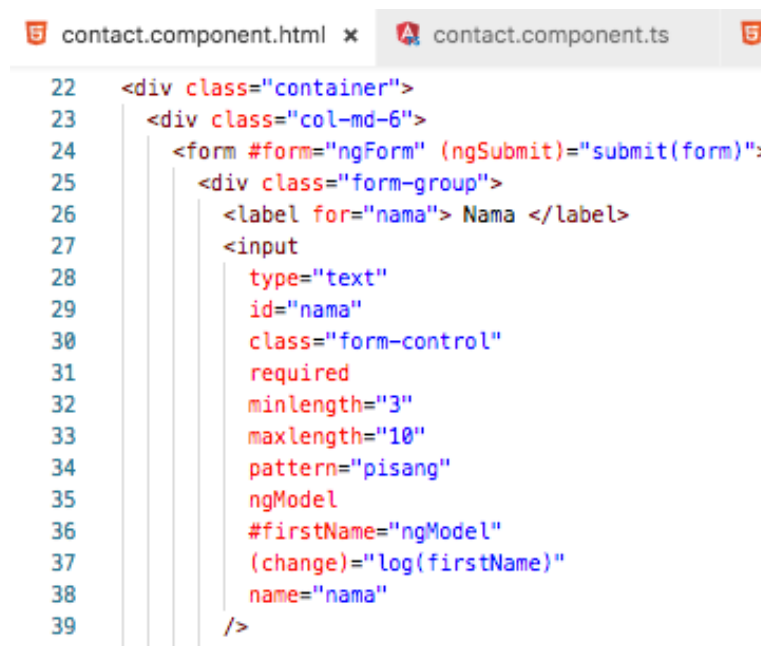
Directive NgForm melengkapi elemen form dengan fitur tambahan. Ini memegang kendali yang kamu buat untuk elemen dengan directive ngModel dan atribut name, dan memonitor properti mereka, termasuk keabsahannya. Ini juga memiliki properti valid sendiri yang mana hanya benar *jika setiap kontrol yang terkandung* itu benar. Untuk lebih jelasnya ikuti langkah-langkah berikut :

- buka file contact.component.ts dan tambahkan sebuah method submit seperti pada gambar berikut (line 23-25)



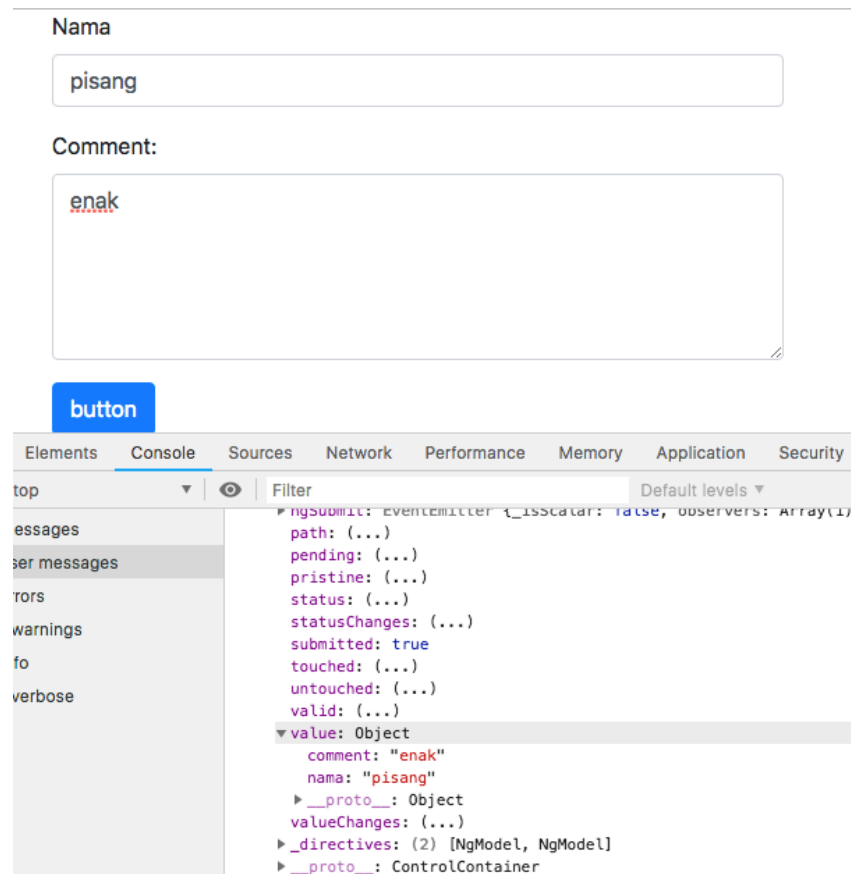
```
18 // }
19 export class ContactComponent {
20   log(z) {
21     console.log(z);
22   }
23
24   submit(form){
25     console.log(form);
26     form.valid;
27   }
28 }
```

- buka file contact.component.html dan buatlah sebuah template variabel ngForm atau property ngForm dengan nama form (#form) yang nanti digunakan sebagai parameter dari method submit seperti pada gambar berikut (line 24)



```
22 <div class="container">
23   <div class="col-md-6">
24     <form #form="ngForm" (ngSubmit)="submit(form)">
25       <div class="form-group">
26         <label for="nama"> Nama </label>
27         <input
28           type="text"
29           id="nama"
30           class="form-control"
31           required
32           minlength="3"
33           maxlength="10"
34           pattern="pisang"
35           ngModel
36           #firstName="ngModel"
37           (change)="log(firstName)"
38           name="nama"
39         />
40       </div>
41     </form>
42   </div>
43 </div>
```

- pada button rubah codenya menjadi `<button type="submit" class="btn btn-primary">`
- jalankan localhost dan inspect element, masukkan pada field nama adalah pisang dan tambahkan pada text area kata-kata setelah itu tekan button. Maka pada saat di inspect element pada ngform property value:object terdapat nilai dari yang diinputkan user seperti pada gambar berikut



6.8 ngModelGroup

Terkadang kita bekerja dengan banyak form dan kita ingin membuat beberapa group dalam form tersebut sebagai contoh kita ingin inputan nama sebagai modelGroup sendiri maka lakukan hal berikut:

- buka file contact.component.html dan tambahkan code (line 26-27) dan copy paste inputan nama ke dalam div ngModelGroup seperti pada gambar

```

contact.component.html • contact.component.ts app.component.html
26 <div ngModelGroup="contact">
27   <!-- copas -->
28 </div>
29 <div class="form-group">
30   <label for="nama"> Nama </label>
31   <input
32     type="text"
33     id="nama"
34     class="form-control"
35     required
36     minlength="3"
37     maxlength="10"
38     pattern="pisang"
39     ngModel
40     #firstName="ngModel"
41     (change)="log(firstName)"
42     name="nama"
43   />
44   <div
45     class="alert alert-danger"
46     *ngIf="firstName.touched && !firstName.valid"
47   >
48     <div *ngIf="firstName.errors['required']">Nama harus diisi</div>
49     <div *ngIf="firstName.errors['minlength']">nama min 3 kata</div>
50     <!-- <div *ngIf="firstName.errors['minlength']">nama min {{firstName.errors['minlengt
51     <div *ngIf="firstName.errors['pattern']">
52       nama tidak sesuai dengan pattern
53     </div>
54   </div>
55 </div>

```

```

26 <div ngModelGroup="contact">
27   <!-- copas -->
28   <div class="form-group">
29     <label for="nama"> Nama </label>
30     <input
31       type="text"
32       id="nama"
33       class="form-control"
34       required
35       minlength="3"
36       maxlength="10"
37       pattern="pisang"
38       ngModel
39       #firstName="ngModel"
40       (change)="log(firstName)"
41       name="nama"
42     />
43     <div
44       class="alert alert-danger"
45       *ngIf="firstName.touched && !firstName.valid"
46     >
47       <div *ngIf="firstName.errors['required']">Nama harus diisi</div>
48       <div *ngIf="firstName.errors['minlength']">nama min 3 kata</div>
49       <div *ngIf="firstName.errors['pattern']">
50         nama tidak sesuai dengan pattern
51       </div>
52     </div>
53   </div>
54 </div>

```

- jika kita ingin membuat validasi pada ngFormGroup maka kita dapat menambahkan template variabel seperti pada gambar berikut

```

26 <div ngModelGroup="contact" #contact="ngModelGroup">
27   <div *ngIf="!contact.valid">contoh validasi pada ngModelGroup</div>
28   <!-- copas -->
29   <div class="form-group">

```

6.9 Disabling the submit button

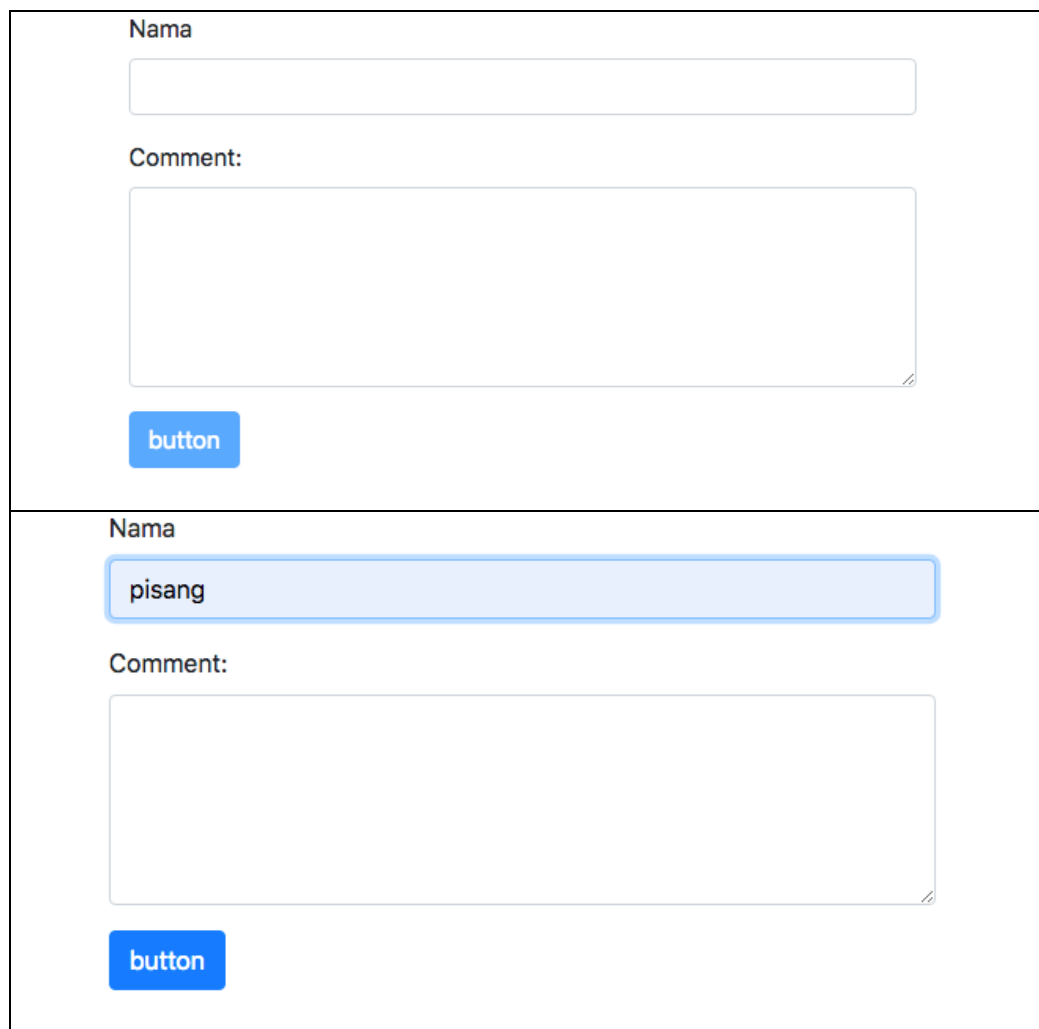
Kita dapat membuat atau mengkondisikan button submit seperti button submit tidak akan dapat diklik jika nilai validnya sama dengan false. Untuk lebih jelasnya ikuti langkah berikut (**kondisikan contact.component.html seperti semula sebelum menggunakan ngFormGroup**)

- buka file contact.component.html dan tambahkan code berikut pada tag button



```
64 </textaread>
65 </div>
66 <button type="submit" [disabled]="!form.valid" class="btn btn-primary">
67   button
68 </button>
```

- jalankan localhost jika berhasil secara default button akan disable, button akan enable saat terdapat inputan



The image shows two states of a web form. The top state shows the form with the 'Nama' field empty and the 'button' disabled. The bottom state shows the form with the 'Nama' field containing the text 'pisang' and the 'button' enabled.

6.10 bekerja dengan check box

Kita juga dapat menambahkan check box pada angular seperti berikut

- buka file contact.component.html dan tambahkan code check box seperti pada gambar berikut (line 66-73)

```

64 </textarea>
65 </div>
66 <div class="checkbox">
67   <label for="">
68     <input type="checkbox" ngModel name="isSubscribe"> Subscribe jika ingin berlangganan
69   </label>
70 </div>
71 <p>
72   {{form.value.json}}
73 </p>
74 <button type="submit" [disabled]="!form.valid" class="btn btn-primary">
75   button
76 </button>
77 </form>
78 </div>
79 </div>

```

penjelasan code

- line 66-70 adalah checkbox
- line 68 terdapat property ngModel dengan name dengan nama isSubscribe
- line 72 digunakan untuk menampilkan data menggunakan json
- jalankan localhost

Nama

pisang

Comment:

rasanya enak

☒ Subscribe jika ingin berlangganan

{ "nama": "pisang", "comment": "rasanya enak", "isSubscribe": true }

button

6.11 bekerja dengan drop-down list

selain menggunakan check box kita juga dapat menggunakan dropdown list. Untuk lebih jelasnya ikuti langkah berikut

- buka file contact.component.html tambahkan code untuk dropdown list seperti pada gambar berikut (line 72-77)


```

66 <div class="checkbox">
67   <label for="">
68     <input type="checkbox" ngModel name="isSubscribe" /> Subscribe jika
69     ingin berlangganan
70   </label>
71 </div>
72 <div class="form-group">
73   <label for="contactMethod">Contact Method</label>
74   <select ngModel name="contactMethod" id="contactMethod" class="form-control">
75     <option *ngFor="let method of contactMethods" [value]="method.id">{{method.name}}</option>
76   </select>
77 </div>
78 <p>
79   {{ form.value | json }}
80 </p>
81 <button type="submit" [disabled]="!form.valid" class="btn btn-primary">
82   button
83 </button>

```

Penjelasan code :

- line 72-77 adalah check box
- line 74 terdapat property name="contactMethod" dari ngModel
- line 75 terdapat directive ngfor dengan property binding [value] dan menampilkan data dengan string interpolasi {{method.name}}
- buka file contact.component.ts dan tambahkan code berikut (line 21-24)

```

18 // }
19 export class ContactComponent {
20
21   contactMethods=[
22     {id:1,name:'email'},
23     {id:2,name:'phone'}
24   ]
25
26   log(z) {
27     console.log(z);
28   }
29
30   submit(form){
31     console.log(form);
32     form.valid;
33   }
34 }

```

- jalankan localhost maka hasilnya seperti berikut

Contact Method

```
{ "nama": "", "comment": "", "isSubscribe": "", "contactMethod": "1" }
```

button

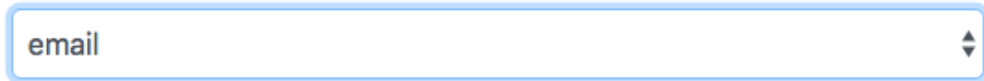
Penjelasan Gambar: jika kita lihat pada gambar diatas saat kita memilih email maka pada json -> contactMethod bernilai 1 seperti pada contact.component.ts

{id:1,name:'email'}. Selain itu kita juga dapat menampilkan property id dan property nama dengan menggunakan property ngValue seperti berikut

```
74 | <select ngModel name="contactMethod" id="contactMethod" class="form-control">
75 |   <option *ngFor="let method of contactMethods" [ngValue]="method">{{method.name}}</option>
76 | </select>
```

sehingga hasilnya seperti berikut

Contact Method



```
{ "nama": "", "comment": "", "isSubscribe": "", "contactMethod": { "id": 1,
"nama": "email" } }
```

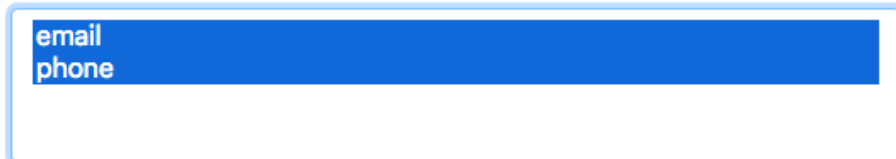
button

selain itu juga kita dapat menggunakan multiple (line 74) jika ingin memilih keduanya seperti pada gambar berikut

```
74 | <select multiple ngModel name="contactMethod" id="contactMethod" class="form-control">
75 |   <option *ngFor="let method of contactMethods" [ngValue]="method">{{method.name}}</option>
76 | </select>
```

sehingga hasilnya seperti berikut (tekan control untuk dapat memilih keduanya)

Contact Method



```
{ "nama": "", "comment": "", "isSubscribe": "", "contactMethod": [ { "id":
1, "nama": "email" }, { "id": 2, "nama": "phone" } ] }
```

button

6.12 bekerja dengan radio button

Selain menggunakan check box dan dropdown list kita juga dapat menggunakan radio button, seperti berikut

- buka file contact.component.html dan tambahkan code berikut (line 83-94)
- catatan jika kita tidak menambahkan ngModel (line 85 dan line 91) pada input maka saat kita memilih salah satu radio button value tidak bisa muncul

```
contact.component.html x contact.component.ts app.component.html
81
82 <!-- radio button -->
83 <div class="radio">
84   <label for="">
85     <input ngModel type="radio" name="contactMethodRadio" value="1">
86     Email
87   </label>
88 </div>
89 <div class="radio">
90   <label for="">
91     <input ngModel type="radio" name="contactMethodRadio" value="2">
92     Phone
93   </label>
94 </div>
95
96 <p>
97   {{ form.value | json }}
98 </p>
```

- jalankan localhost (menggunakan ngModel)

☐ Subscribe jika ingin berlangganan

Contact Method

email
phone

☐ Email

☒ Phone

{ "nama": "", "comment": "", "isSubscribe": "", "contactMethod": "",
"contactMethodRadio": "2" }

button

- jalankan localhost (tanpa menggunakan ngModel)

☐ Email

☒ Phone

{ "nama": "", "comment": "", "isSubscribe": "", "contactMethod": "" }

button

atau kita bisa menggunakan directive ngFor untuk menampilkan value seperti berikut

```
contact.component.html x contact.component.ts app.component.html
81
82 <!-- radio button -->
83 <div *ngFor="let method of contactMethods" class="radio">
84   <label for="">
85     <input ngModel type="radio" name="contactMethodRadio" [value]="method.id">
86       {{method.name}}
87     </label>
88   </div>
89
90 <p>
91   {{ form.value | json }}
92 </p>
93 <button type="submit" [disabled]="!form.valid" class="btn btn-primary">
94   button
95 </button>
96 </form>
97 </div>
98 </div>
99
```

sehingga hasilnya juga sama

☐ Subscribe jika ingin berlangganan

Contact Method

email
phone

☐ Email

☒ Phone

{ "nama": "", "comment": "", "isSubscribe": "", "contactMethod": "",
"contactMethodRadio": "2" }

button

