To setup a template driven form we need the following pieces :

* In the form tag add # to assign the form to a variable and ngsubmit(#) to handle the submit  
  <form #form="ngForm" (submit)="onsubmit(form)">
* For each input field the following will mark the input as a part of the object  
  [(ngModel)]="Name" #Name="ngModel"
* The submit method in the ts file can be  
  onsubmit(form:NgForm)

  {

    console.log(form.valid);

    console.log(form.controls);

    console.log(form.value);

  }

Classes and properties that define the form state

|  |  |
| --- | --- |
| Ng-valid  Ng-invalid | Valid  Invalid |
| Ng-touched  Ng-untouched | Touched  Untouched |
| Ng-dirty  Ng-pristine | Dirty  pristine |
|  | submitted |

To add a border in case of the field touched and invalid

input.ng-invalid.ng-touched,

select.ng-invalid.ng-touched

{

    border: 1px solid red;

}

Or

A nested ngif div can be built in the form to handle the errors

Can be used in the submit function to prevent submitting

The inputs can be grouped to better form the resulted form object

Surround multiple inputs with a div and give it the following

ngModelGroup="UserData" #UserData="ngModelGroup"

and you can also validate the whole div

<div class="row" \*ngIf="!UserData.valid">

<span>user Data is Invalid</span>

    </div>

-Getting access to the form outside the submit method

@viewchiled can be used to access the form from anywhere at the typescript code

@ViewChild("f") signupform:NgForm;

onsuggest()

  {

    this.signupform.form.patchValue({

      UserData:{username:"superman"}

    })

  }

this.signupform.form.reset() will reset the form