**Part 7 - Reactive Forms**

Here we’re going to learn about

1. What is reactive forms
2. How to use reactive forms
3. Validation and Dynamic Error Message and Setup Global css
4. Submit forms
5. WHAT IS REACTIVE FORMS

Reactive forms provide a model driven approach to handle form inputs. In template driven forms we have to put the logic of forms such a business validation in the template / html. Whereas in reactive forms we just put the business logic forms in component. So, in case we have a larger form then it will more readable and maintainable since in the template is cleaner from code. On the one hand, by using reactive form, developer can be easily to create custom or cross validation. For instance, we have 2 password fields (Password and Confirm Password). Here we only need create custom method for custom or cross validation inside the component.

<https://blog.angular-university.io/introduction-to-angular-2-forms-template-driven-vs-model-driven/#:~:text=Here%20are%20the%20differences%20between,level%20of%20the%20component%20class>

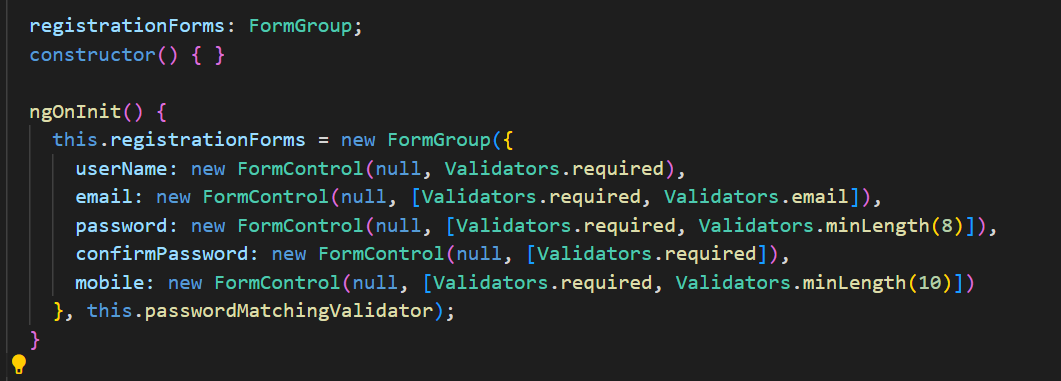
1. HOW TO USE IT



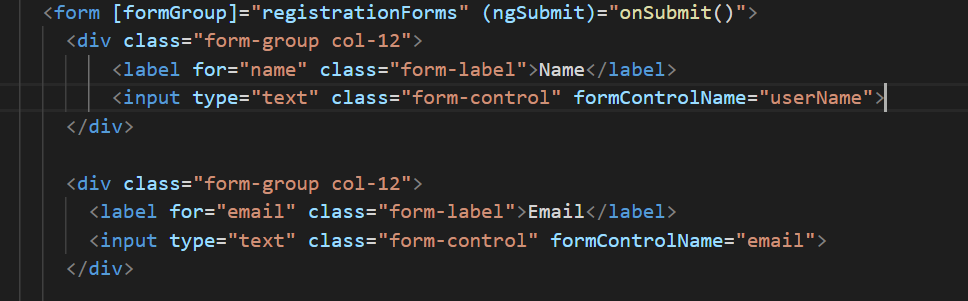
Use ReactiveFormsModule instead of FormModule

1. VALIDATION AND ERROR MESSAGE

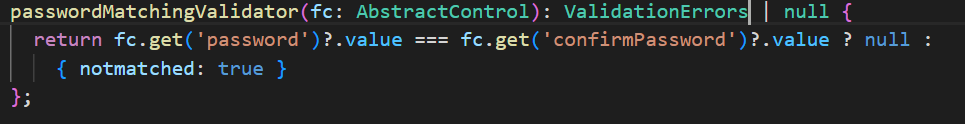
Create User-Registration Component



1. Create registrationForms variable then instance it from ngOnInit method
2. Put your business validation
3. Basically form inputs will be wrapped by FormGroup

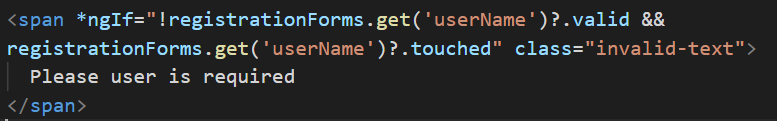


1. Fill the FormGroup Directive using registrationForms variable
2. Just need the form control name on the formControlName property

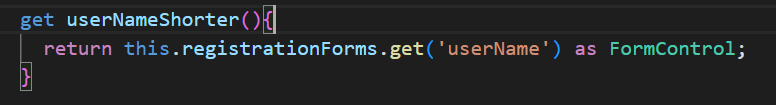


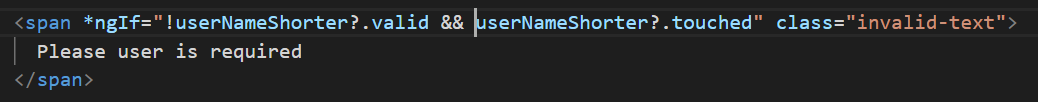
This is a cross and custom validation. In order to check weather password and confirm password are equals or not.

Applying validation in template

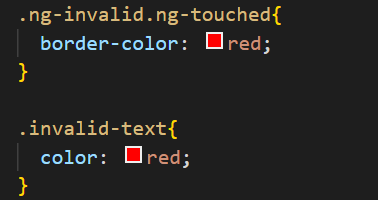


Or

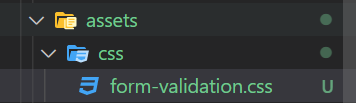




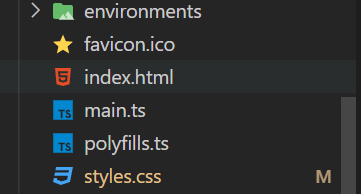
Setup global css

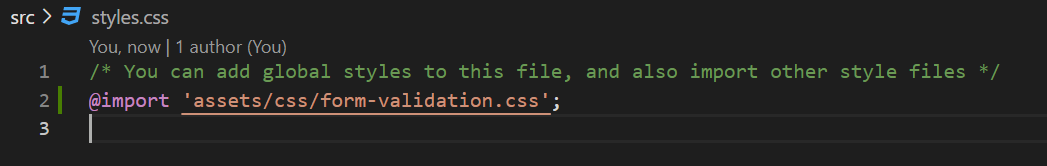


This css class will be used for giving red color at border forms and error message text. Instead of putting this class on each component css, we will put it on “assets” follder

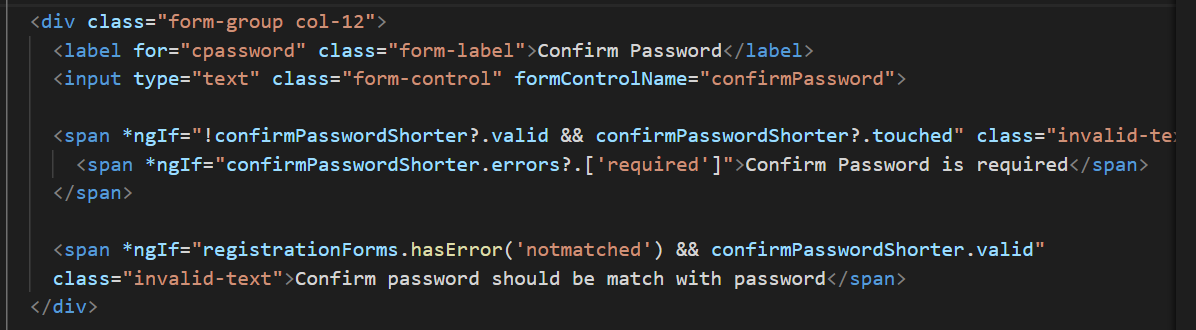


Once you have done created the file, then copy paste the code on this file. After that register your css at style.css





Custom validation implementation



1. Notmatched error is from form level, not control level. So we use registrationForms
2. We put this validation outside the span validation. Since we will check this notmathced validation only if the common validation is valid

<https://www.youtube.com/watch?v=6aOg_IMQjyE&list=PL_NVFNExoAxclqXo9fLAeP0G2Qp56Fu8C&index=16>

Advantages

1. Cleaner template (designer can be focused on html and developer can be focused on logic component), since we separate between template and logic

<https://www.youtube.com/watch?v=WNSjvhJKaIA&list=PL_NVFNExoAxclqXo9fLAeP0G2Qp56Fu8C&index=17>