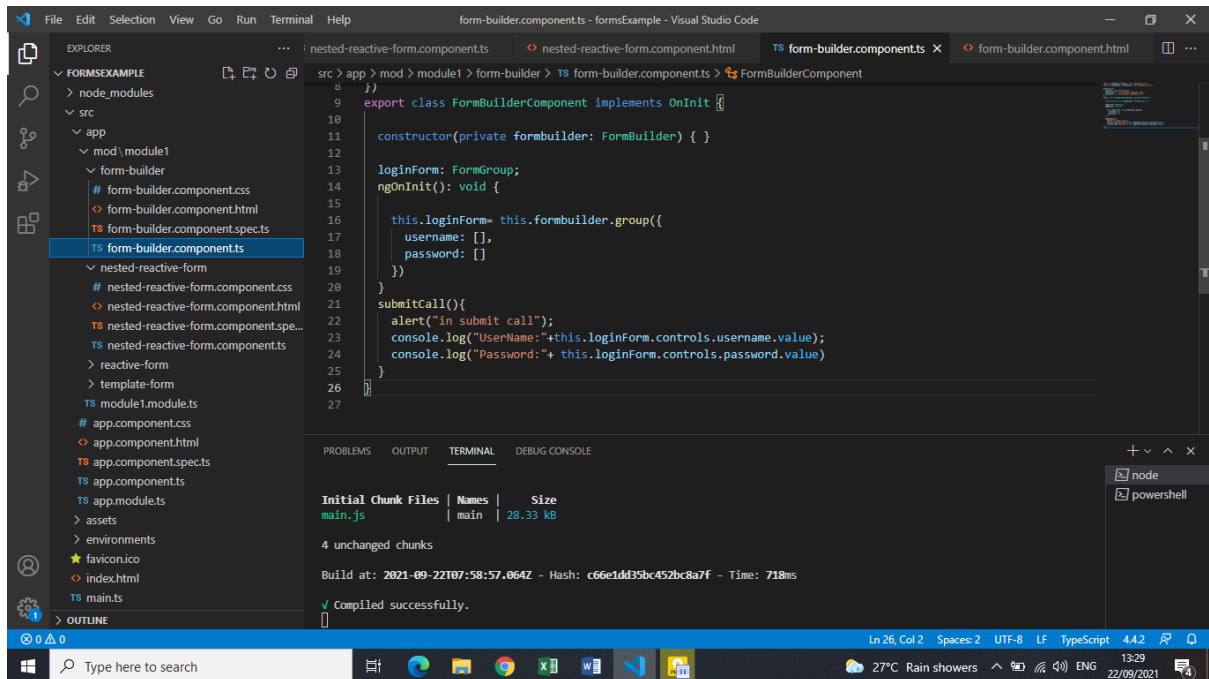


FORMBUILDER

The FormBuilder provides syntactic sugar that shortens creating instances of a FormControl, FormGroup, or FormArray. It reduces the amount of boilerplate needed to build complex forms.



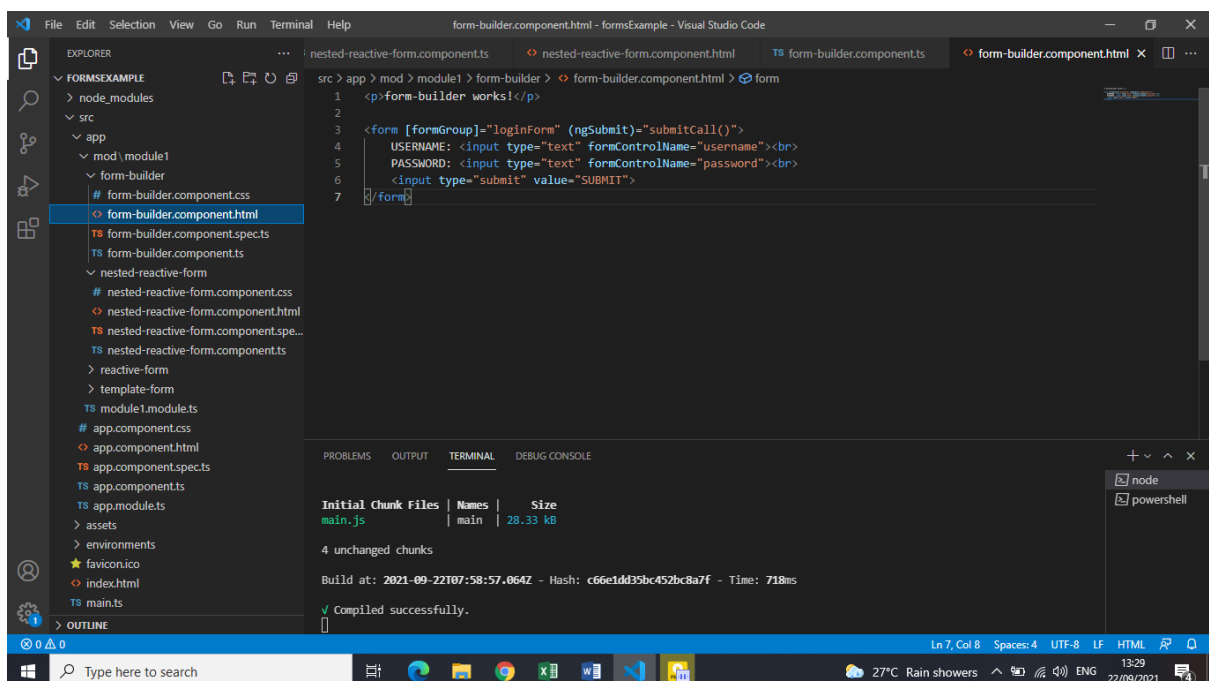
```
8  }
9  export class FormBuilderComponent implements OnInit {
10
11    constructor(private formbuilder: FormBuilder) { }
12
13    loginForm: FormGroup;
14    ngOnInit(): void {
15
16      this.loginForm = this.formbuilder.group({
17        username: [],
18        password: []
19      })
20    }
21    submitCall() {
22      alert("in submit call");
23      console.log("UserName: "+this.loginForm.controls.username.value);
24      console.log("Password: "+ this.loginForm.controls.password.value)
25    }
26  }
27
```

Initial Chunk Files | Names | Size
main.js | main | 28.33 kB

4 unchanged chunks

Build at: 2021-09-22T07:58:57.064Z - Hash: c66e1dd35bc452bc8a7f - Time: 718ms

✓ Compiled successfully.



```
1 <p>form-builder works!</p>
2
3 <form [formGroup]="loginForm" (ngSubmit)="submitCall()">
4   USERNAME: <input type="text" formControlName="username"><br>
5   PASSWORD: <input type="text" formControlName="password"><br>
6   <input type="submit" value="SUBMIT">
7 </form>
```

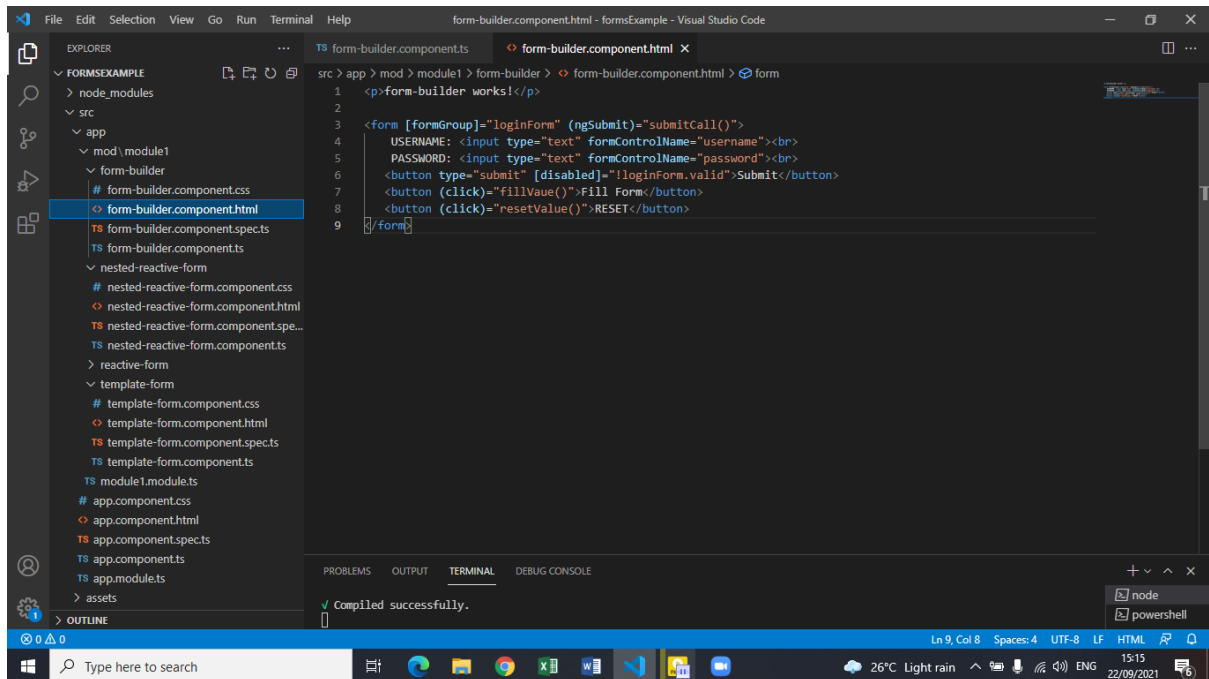
Initial Chunk Files | Names | Size
main.js | main | 28.33 kB

4 unchanged chunks

Build at: 2021-09-22T07:58:57.064Z - Hash: c66e1dd35bc452bc8a7f - Time: 718ms

✓ Compiled successfully.

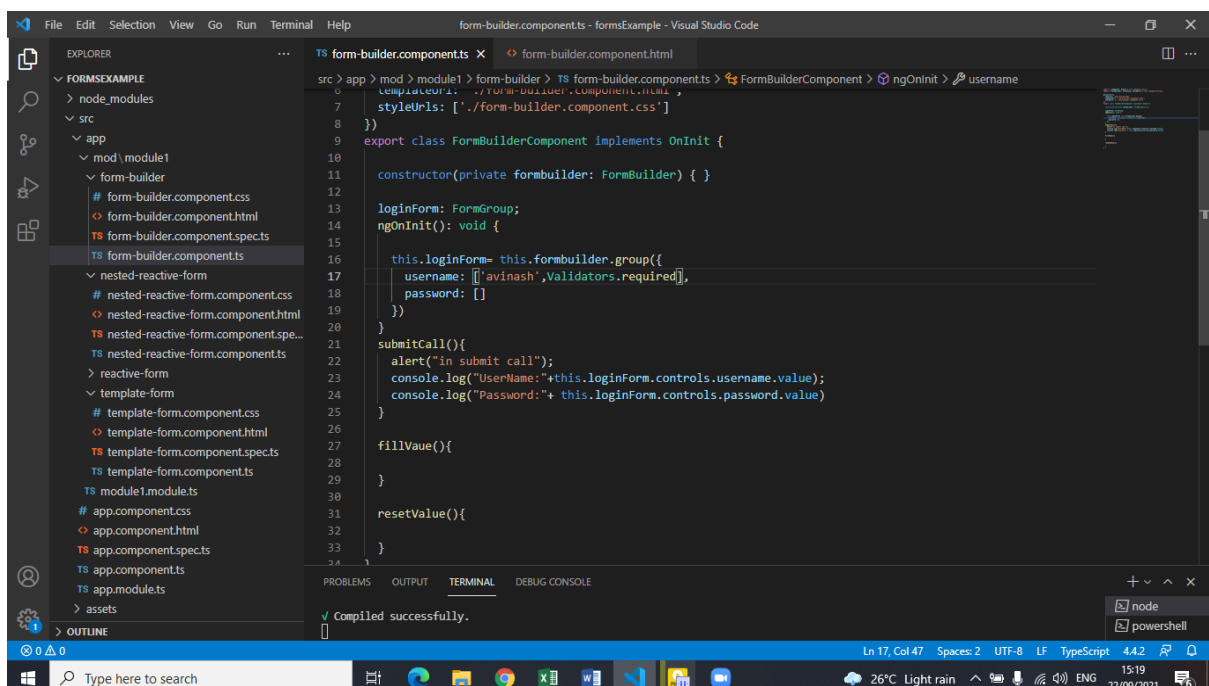
Validators in FormBuilder:



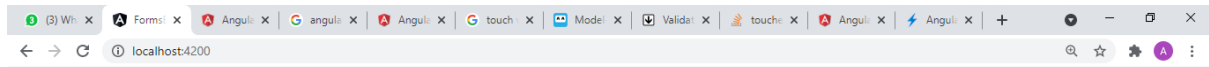
`!loginForm.valid`

This will use for the checking purpose, where the inputs are valid then it returns true or else it will return false.

To activate such validations we have to write validators in .ts file like,



If username is present, then Submit button is enable



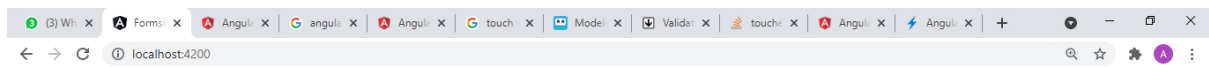
form-builder works!

USERNAME:

PASSWORD:



If we remove data from username field, then



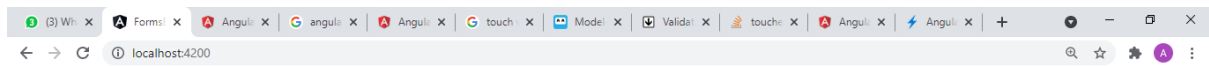
form-builder works!

USERNAME:

PASSWORD:



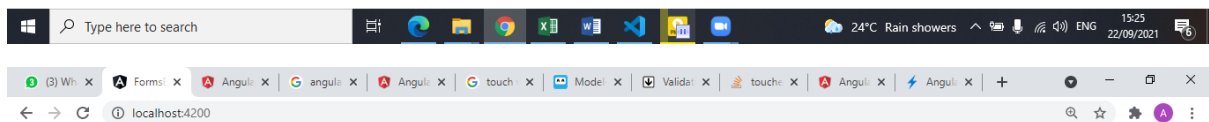
For Both Username and password field:



form-builder works!

USERNAME:

PASSWORD:

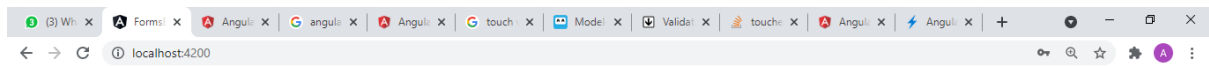


form-builder works!

USERNAME:

PASSWORD:





form-builder works!

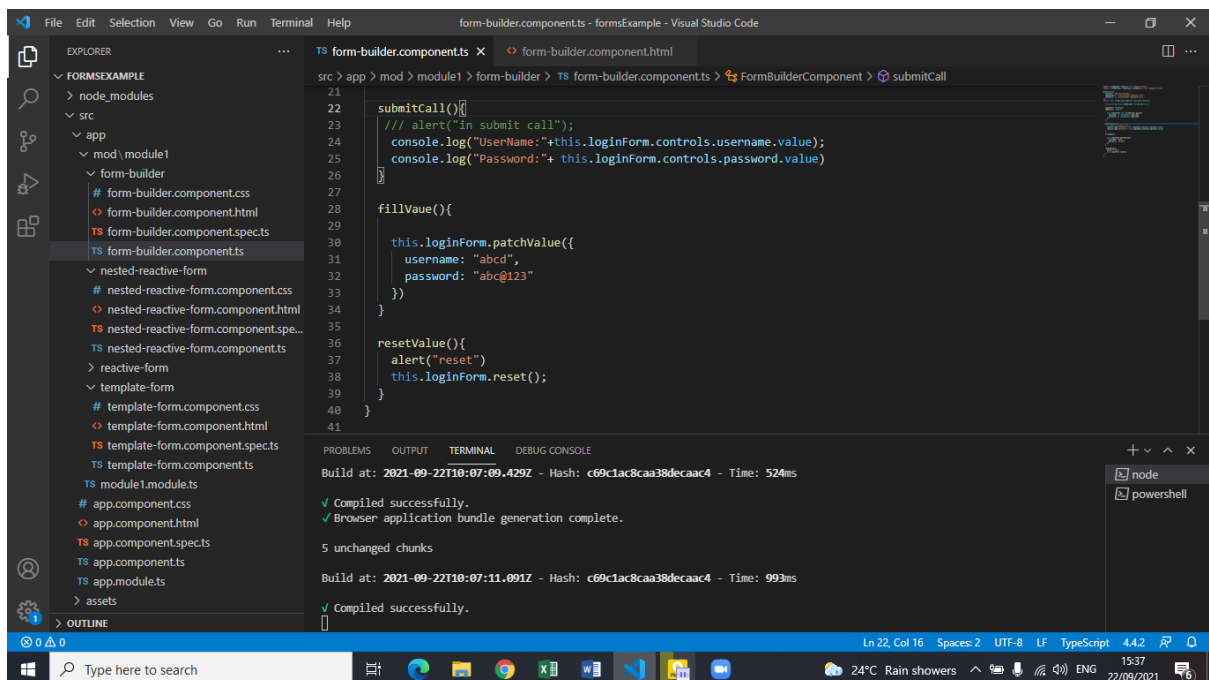
USERNAME:

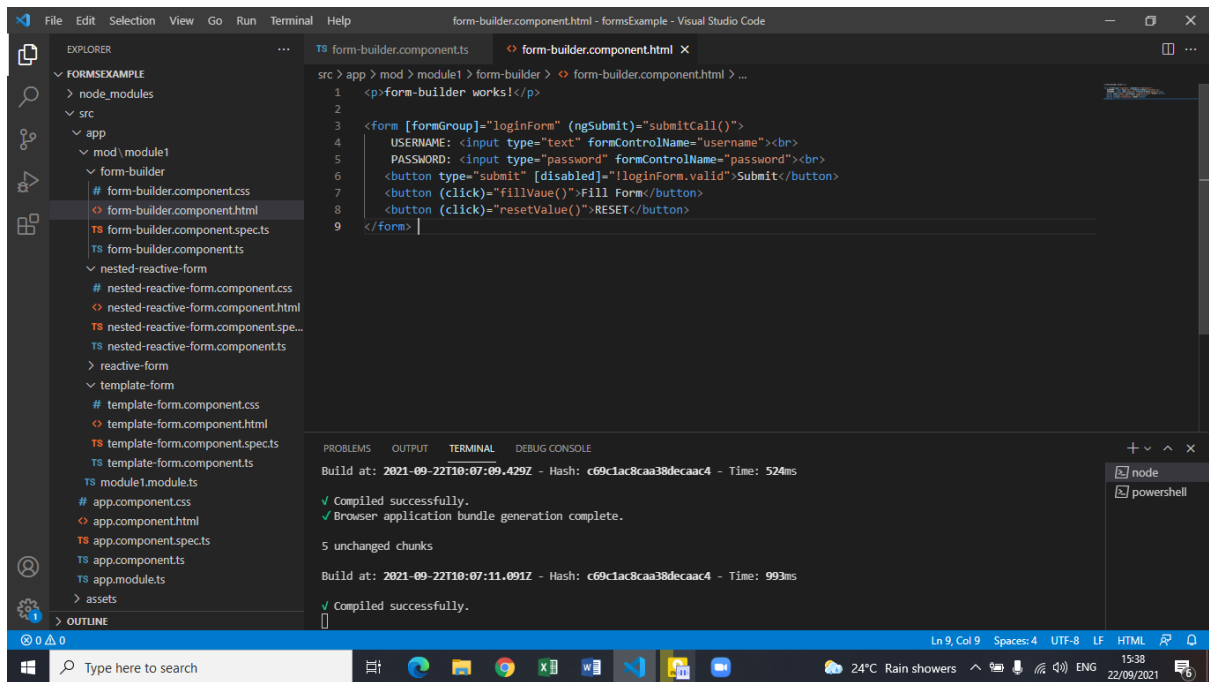
PASSWORD:



FillForm():

This is used for the adding a default value to the fields, like permanent and local address is same, so we fill local address same as permanent address by clicking on checkbox or button

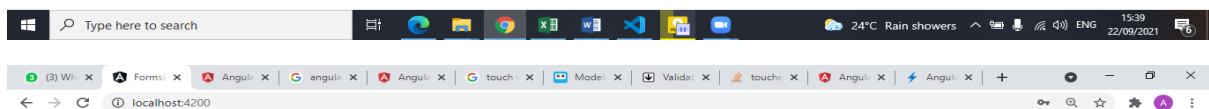




form-builder works!

USERNAME:

PASSWORD:



form-builder works!

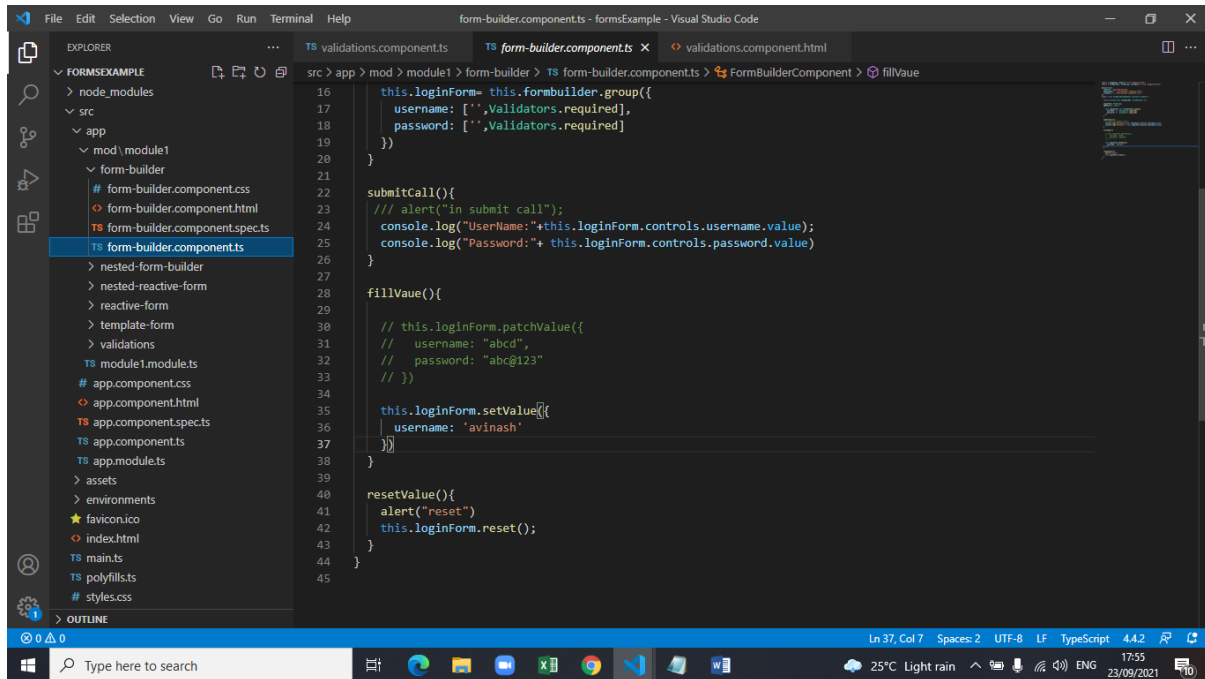
USERNAME:

PASSWORD:



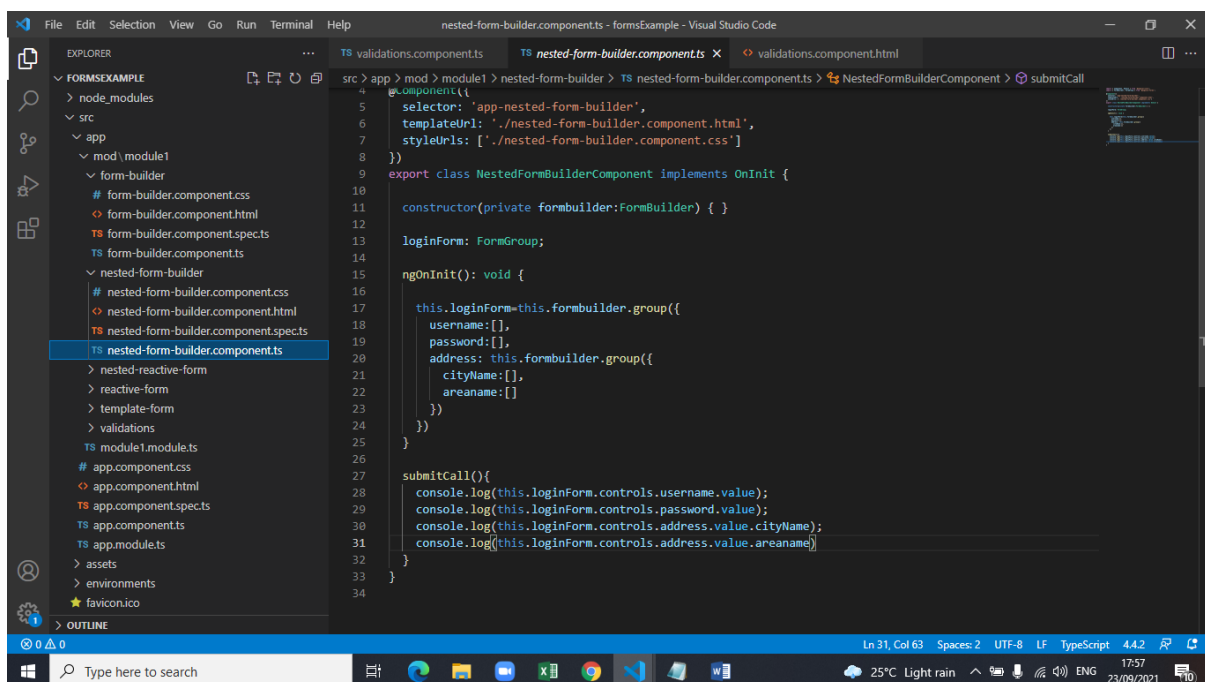
setValue() method:

we need to set compulsory all values on form, then we can go with set value.



```
src > app > mod > module1 > form-builder > TS form-builder.components.ts > FormBuilderComponent > fillVaue
16  this.loginForm=this.formbuilder.group({
17    username: ['',Validators.required],
18    password: ['',Validators.required]
19  })
20
21
22  submitCall(){
23    // alert("in submit call");
24    console.log("UserName:"+this.loginForm.controls.username.value);
25    console.log("Password:"+ this.loginForm.controls.password.value)
26  }
27
28  fillVaue(){
29    // this.loginForm.patchValue({
30    //   username: "abcd",
31    //   password: "abc@123"
32    // })
33
34    this.loginForm.setValue({
35      username: 'avinash'
36    })
37  }
38
39  resetValue(){
40    alert("reset")
41    this.loginForm.reset();
42  }
43
44
45
```

Nested Form Builder:



```
src > app > mod > module1 > nested-form-builder > TS nested-form-builder.components.ts > NestedFormBuilderComponent > submitCall
5  selector: 'app-nested-form-builder',
6  templateUrl: './nested-form-builder.component.html',
7  styleUrls: ['./nested-form-builder.component.css']
8  })
9  export class NestedFormBuilderComponent implements OnInit {
10
11    constructor(private formbuilder:FormBuilder) { }
12
13    loginForm: FormGroup;
14
15    ngOnInit(): void {
16
17      this.loginForm=this.formbuilder.group({
18        username:[],
19        password:[],
20        address: this.formbuilder.group({
21          cityName:[],
22          areaname:[]
23        })
24      })
25    }
26
27    submitCall(){
28      console.log(this.loginForm.controls.username.value);
29      console.log(this.loginForm.controls.password.value);
30      console.log(this.loginForm.controls.address.value.cityName);
31      console.log(this.loginForm.controls.address.value.areaname)
32    }
33  }
34
```

