# Angular Forms - Lab

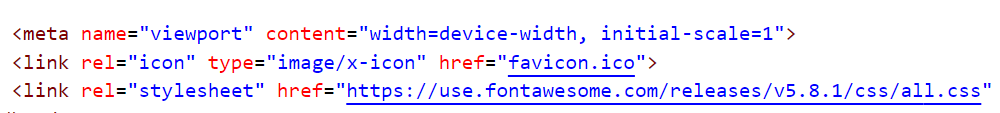
## Initial Setup

Create a new angular application. You are provided with an **HTML file** containing the form itself with **bootstrap** and **font awesome** icons.

Install **bootstrap** through **NPM** or link it using the CDN inside **index.html** (your choice). Import it in **angular.json**:



Go to [fontawesomesome.com](https://fontawesome.com/start) and add the provided HTML link inside **index.html:**



## Template-Driven Form

Generate a **register-form** component and use the provided HTML to create a **template-driven** form. Attach all the needed directives like (**ngForm**, **ngModel**).

Use **@ViewChild** to reference the **NgForm** inside the component.

Handle **submitting** of the form. Inside the handler function **reset** the form.

Define the options for both select lists as an **array of strings** and loop through the arrays inside the template. Add a **default value** to both select lists.

Define the following CSS inside **register-form.component.css**:

|  |
| --- |
| input.ng-valid {  border-left: 5px solid #42A948; /\* green \*/  }  input.ng-invalid.ng-touched {  border-left: 5px solid #A94442; /\* red \*/  } |

### Validation

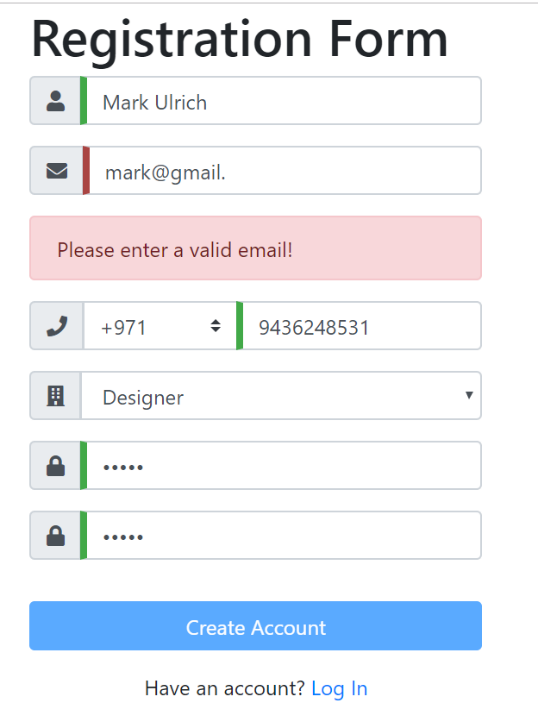
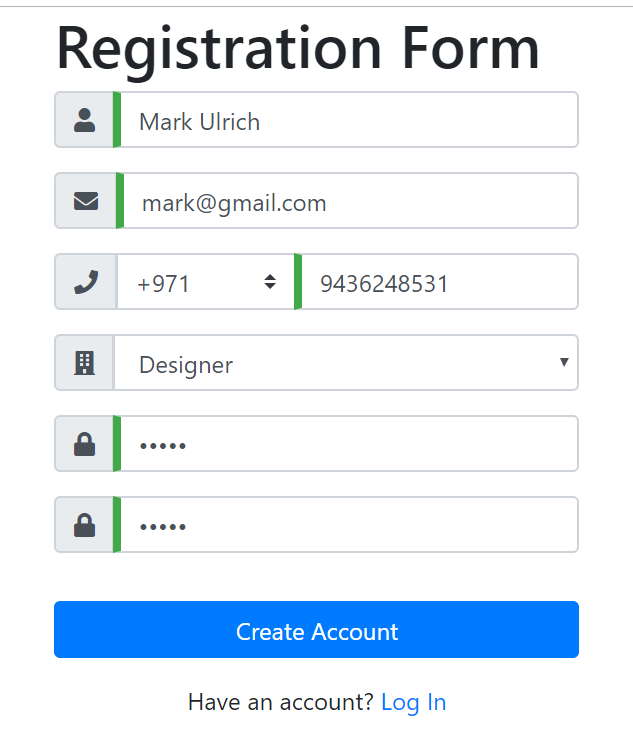
* The full name input field must contain **two names** (letters only) separated by a **space**. Both should start with a **capital** letter.
* The email input field must be a **valid email** address.
* The phone number input field must be a **9 digit** number.
* The password input field must be between **3** and **16** **symbols** and must contain only **letters** and **digits**.
* Both passwords must **match**.

Output error messages inside a div with class **"alert alert-danger"** when an input field is **invalid** and **touched,** also **disable** the **submit button** if an input field is invalid.

### Grouping

Wrap both password fields inside a common **ngModelGroup** and add the **password validation** after the group definition.

At the **end** your form should look like this:

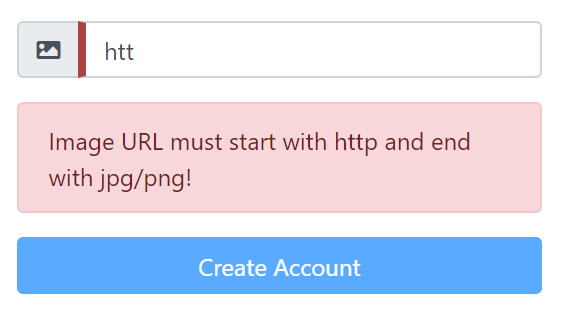


## Create a Custom Attribute Directive

Let's add another input field containing an **image URL** to a profile picture (use '**fa fa-image**' for the font awesome icon). Validate this input field by writing your own **attribute directive**.

The directive should listen to **input** events and validate if the input value **starts with http** and **ends with jpg/png**.

Use **Renderer2** to set the border-left **styling** of the field to either red or green. To manually change the state of an input field to **invalid** you must inject the **NgForm** inside the directive and use the **setErrors({ 'incorrect': true })** method on the control to mark it as invalid. Clearing the errors is done with **setErrors(null)**



## Reactive Form

Create a new component **register-form-reactive** and now build the same form using the reactive approach. Create your own **form groups** and **form controls** (optionally you can inject the **form builder**). Validate the fields using **Validators** and etc.