Wednesday: Form Validation

Form Validation

Users may end up trying to add a new goal without a name or description or a date while a goal is required to have each one. This may lead to our application developing errors. To mitigate this possible probability, we need to perform form validation on our form inputs and then alert users in case we encounter validity problems.

Let's see how we should do this.

src/app/goal-form/goal-form.component.html

```
<div class="container-fluid">
  <h2 class="text-center">Create a new Goal</h2>
  <hr>
  <form #goalForm='ngForm'>
      <div class="form-group">

          <label for="name">Name</label>
          <input type="text" class="form-control" id="name" [(ngModel)]="newGoal.name" name="name" #name='ngModel' required>
          <div [hidden]="name.valid || name.pristine" class="alert alert-danger">
            <p>Name is required</p>
          </div>
      </div>
....
```

We have added a template reference variable `#name` to access Angular's form control on this input from the template. We have then created a div that has an attribute `[hidden]` which makes sure the alert is shown only when the name input is invalid. The `pristine` property allows us to hide the alert when we load a blank page.

If you type anything in the name input and then delete it now, the alert will show. Let's apply this to all our form inputs.

src/app/goal-form/goal-form.component.html

```
<div class="container-fluid">
  <h2 class="text-center">Create a new Goal</h2>
  <hr>
  <form #goalForm='ngForm'>
      <div class="form-group">

          <label for="name">Name</label>
          <input type="text" class="form-control" id="name" [(ngModel)]="newGoal.name" name="name" #name='ngModel' required>
          <div [hidden]="name.valid || name.pristine" class="alert alert-danger">
            <p>Name is required</p>
          </div>
      </div>
      <div class="form-group">

          <label for="description">Description</label>
          <textarea class="form-control" id="description" rows="4" [(ngModel)]="newGoal.description" name="description" #description="ngModel" required></textarea>
          <div [hidden]="description.valid || description.pristine" class="alert alert-danger">
            <p>Description is required</p>
          </div>
      </div>
      <div class="form-group">
```

```
          <label for="complete">Completion</label>
          <input type='date' id="complete" [(ngModel)]="newGoal.completeDate" name="completeDate" #
completeDate="ngModel" required>
          <div [hidden]="completeDate.valid || completeDate.pristine" class="alert alert-danger">
            <p>Date is required</p>
          </div>
      </div>
      <button type="submit" class="btn btn-success">Add Goal</button>
  </form>
</div>
```

If we try adding to an input and then deleting it, the respective error messages will show indicating that the user will now be aware of any invalid inputs which improves user experience and also makes sure our application does not break from invalid data.