7. Design a Student Registration Form in Angular with validation (template-driven and reactive)

For Angular version less than 17, import FormsModule inside app.module.ts

```
// src/app/app.module.ts
import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';
import { FormsModule } from '@angular/forms'; // <-- MUST be included

import { AppComponent } from './app.component';

@NgModule({
  declarations: [
    AppComponent
  ],
  imports: [
    BrowserModule,
    FormsModule // <-- Added here
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

For Angular version 17 and higher, import FormsModule directly inside into the component's import array.

```
import { Component } from '@angular/core';
import { FormsModule } from '@angular/forms'; // <-- Import here

@Component({
  standalone: true, // <-- This is key
  selector: 'app-student-form',
  templateUrl: './student-form.component.html',
  imports: [FormsModule] // <-- Add to the component's imports array
})
```

src/app/app.component.ts

```
import { Component } from '@angular/core';

interface Student {
  studentName: string;
  usn: string;
}

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {

  studentModel: Student = {
    studentName: '',
    usn: ''
 };

  onSubmit(): void {
    // TypeScript now knows studentModel has 'studentName' and 'usn'
    console.log('Form Submitted Data:', this.studentModel);
    alert(`Registration successful for: ${this.studentModel.studentName}`);
    this.studentModel = { studentName: '', usn: '' };
  }
}
```

src/app/app.component.html

```
<div class="container">
  <h3>Student Registration</h3>

  <form #studentForm="ngForm" (ngSubmit)="onSubmit()">

    <div class="form-group">
      <label for="studentName">Student Name</label>

      <input type="text"
             id="studentName"
             name="studentName"
             [(ngModel)]="studentModel.studentName"
             required
             #name="ngModel">

      <div *ngIf="name.invalid && name.touched" class="error-message">
        Student Name is required.
      </div>
    </div>

    <div class="form-group">
      <label for="usn">USN Number</label>

      <input type="text"
             id="usn"
             name="usn"
             [(ngModel)]="studentModel.usn"
             required
```

FULL

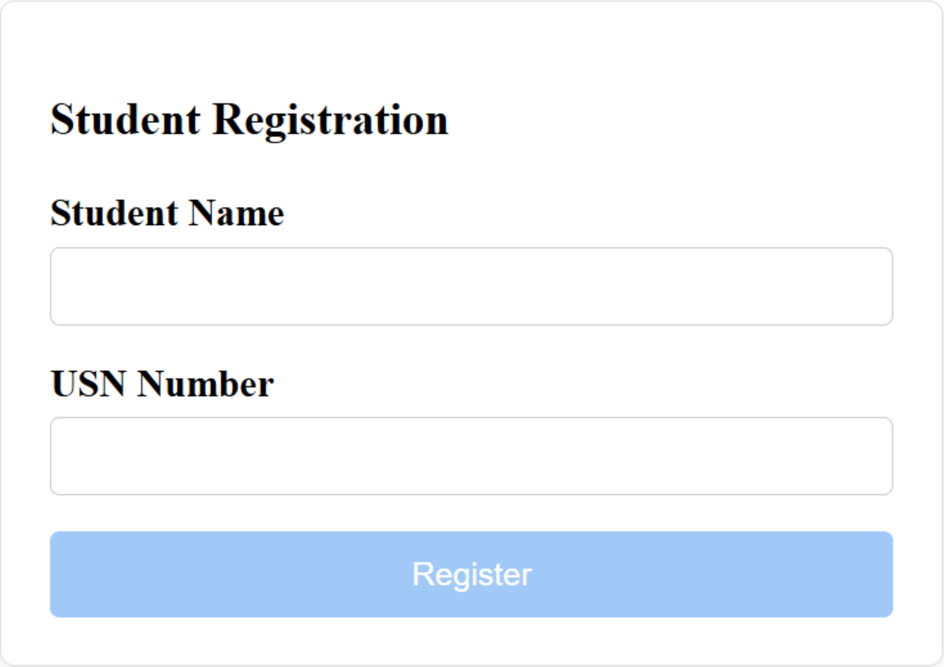
```
             #usn="ngModel">

      <div *ngIf="usn.invalid && usn.touched" class="error-message">
        USN Number is required.
      </div>
    </div>

    <button type="submit" [disabled]=" studentForm.invalid">
      Register
    </button>

  </form>
</div>
```

src/app/app.component.css:

```
.container {
  max-width: 350px;
  margin: 50px auto;
  padding: 20px;
  border: 1px solid #ddd;
  border-radius: 6px;
  box-shadow: 0 2px 4px rgba(0, 0, 0, 0.05);
}

.form-group {
  margin-bottom: 15px;
}

label {
  display: block;
  margin-bottom: 5px;
  font-weight: 600;
}

input {
  width: 100%;
  padding: 8px;
  border: 1px solid #ccc;
  border-radius: 4px;
  box-sizing: border-box;
}

/* Angular's automatic class styling for invalid fields */
input.ng-invalid.ng-touched {
  border-color: #dc3545; /* Red border */
}
```

```
.error-message {
  color: #dc3545; /* Red text */
  font-size: 0.85em;
  margin-top: 3px;
}

button[type="submit"] {
  padding: 10px 15px;
  background-color: #007bff;
  color: white;
  border: none;
  border-radius: 4px;
  cursor: pointer;
  width: 100%;
}

button[type="submit"]:disabled {
  background-color: #a0c9f8;
  cursor: not-allowed;
}
```

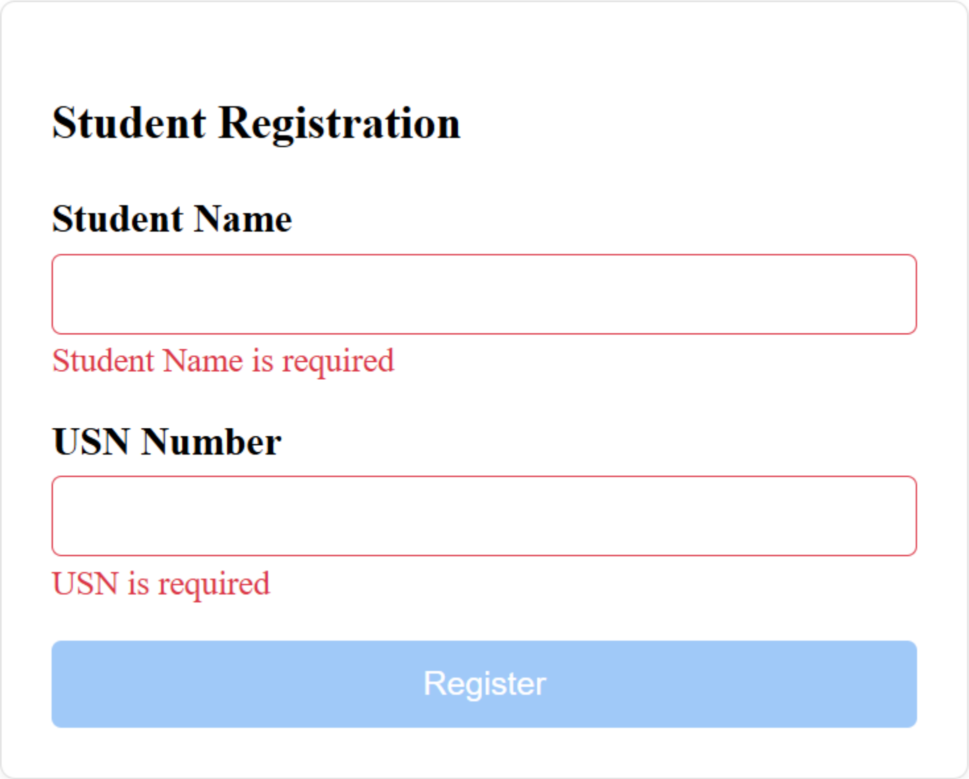
Output:



Student Registration

Student Name

USN Number



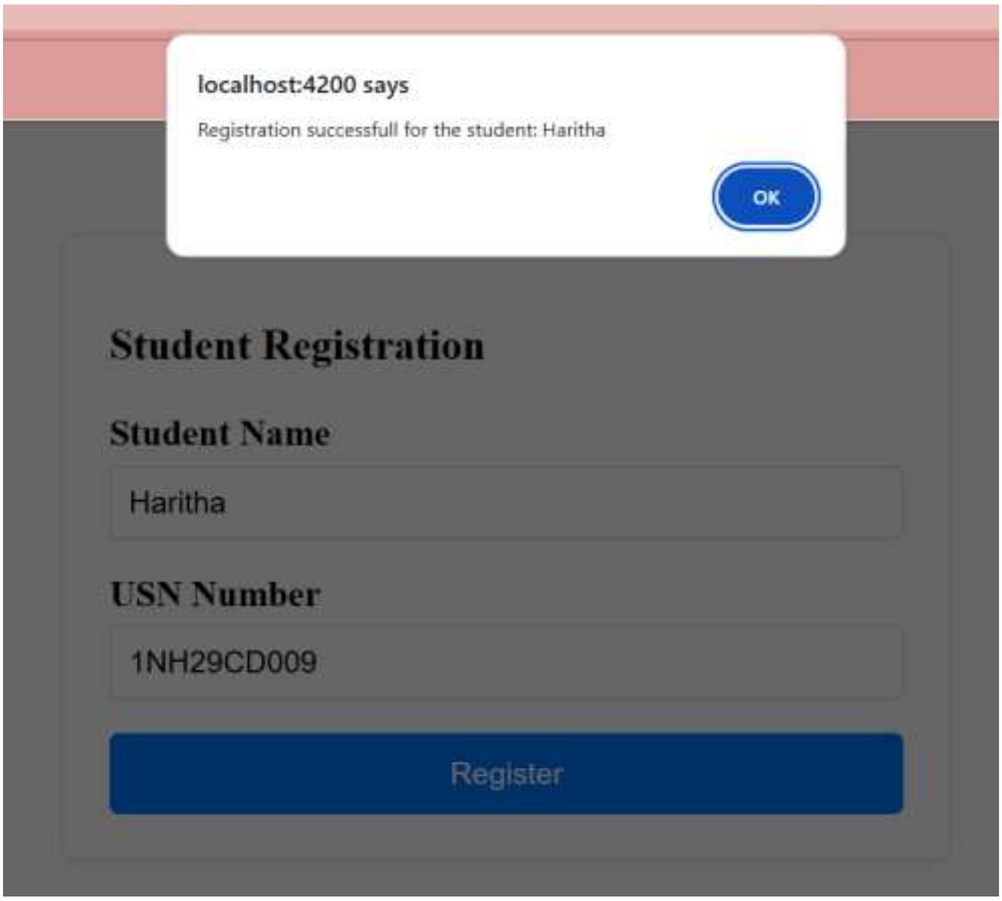
Student Registration

Student Name

Student Name is required

USN Number

USN is required



localhost:4200 says

Registration successfull for the student: Haritha

Student Registration

Student Name

Haritha

USN Number

1NH29CD009