SSGMCE	SHRI SANT GAJANAN MAHARAJ COLLEGE OF ENGG.			**LABORATORY MANUAL**	
	PRACTICAL EXPERIMENT INSTRUCTION SHEET				
	EXPERIMENT TITLE: UNDERSTANDING AND USE OF NGSTYLE ANGULAR DIRECTIVES.				
EXPERIMENT NO.: **SSGMCE-WI-**IT-5IT09-06			ISSUE NO.: 00	ISSUE DATE: 30-07-2025	
REV. DATE:		REV. NO.:	DEPTT.: INFORMATION TECHNOLOGY		
LABORATORY: Computer Laboratory- 01				SEMESTER: V	PAGE :1 OF 3

1.0) AIM: To study and implement different control flow statements such as structural directives and attribute in Angular for dynamic rendering of HTML elements.

2.0) SCOPE:

The purpose of this experiment is to demonstrate how Angular directives can control the rendering of DOM elements based on conditions, loops, and multiple choice logic.

3.0) FACILITIES/ APPARATUS:

i)Hardware: I3 Processor, 8GB RAM, HD Monitor, Windows 10, Internet connectivity

ii) Software: Updated Web browser, Node js, Angular CLI, VS Code.

4.0) THEORY:

In Angular, **directives** are special markers in the DOM that extend the behaviour of HTML elements. They help manipulate the DOM, apply styles, and add dynamic behaviour to applications. Angular provides three main types of directives:

- **Component Directives**:
  These are the most common directives in Angular. Every component is essentially a directive with a template.
  Example: <app-header></app-header>
- **Structural Directives**:
  These modify the DOM structure by adding or removing elements. They usually have a * prefix.
  - *ngIf: Conditionally adds or removes elements.
  - *ngFor: Iterates over a list and creates elements dynamically.
  - *ngSwitch: Displays elements based on a switch-case expression.
- **Attribute Directives**:
  These modify the appearance or behaviour of existing elements.
  - ngClass: Dynamically adds or removes CSS classes.
  - ngStyle: Dynamically applies inline CSS styles.
  - Custom attribute directives can also be created to implement specific behaviours.

5.0) PROGRAM CODE:

1. **Create a new Angular project**
   a. ng new directives_demo
   b. cd directives_demo
2. **Generate a new component**

   ng g c directives
3. **Implement different directives in the component HTML**

   Use **attribute directives** ([ngClass], [ngStyle]) to apply dynamic styling.

   directives.html

```
<div class="container mt-4">
   <h2 class="mb-3 text-primary">Angular 20 - ngClass and ngStyle Demo</h2>
   <!-- ngClass Example -->
   <p [ngClass]="{'text-success': isActive, 'text-danger': !isActive}">
      This text changes color based on <strong>isActive</strong>.
   </p>
   <button class="btn btn-info me-2" (click)="toggleActive()">Toggle Active</button>
   <hr>
   <!-- ngStyle Example -->
   <p [ngStyle]="{'font-size': isHighlighted ? '24px' : '16px',
            'background-color': isHighlighted ? 'yellow' : 'white',
            'padding': '10px'}">
      This text changes style dynamically using <strong>ngStyle</strong>.
   </p>
   <button class="btn btn-warning" (click)="toggleHighlight()">Toggle Highlight</button>
</div>
```

4. **Add logic in the TypeScript component**

```
import { CommonModule } from '@angular/common';
import { Component } from '@angular/core';
@Component({
  selector: 'app-directives',
  imports: [CommonModule],
  templateUrl: './directives.html',
  styleUrl: './directives.css'
})
export class Directives {
  isActive: boolean = true;
  isHighlighted: boolean = false;
  toggleActive() {
    this.isActive = !this.isActive;
  }
  toggleHighlight() {
    this.isHighlighted = !this.isHighlighted;
  }
}
```

5. **Run the Project**

```
cd directives
ng serve
```

6.0) CONCLUSION:

By using directives in Angular 20, we can control the structure, style, and behavior of DOM elements dynamically.

- **Attribute directives** like ngClass and ngStyle enable dynamic styling and behavior.
- **Component directives** act as reusable building blocks of the application.

Thus, directives make Angular applications more **interactive, reusable, and maintainable**, providing flexibility in both UI design and functionality