# Create Custom Directives

## 1. Setup Project

### 1.1 Install Dependencies

1. Change directory to calab:

* cd calab

1. Install dependencies by running the following command:

* npm install

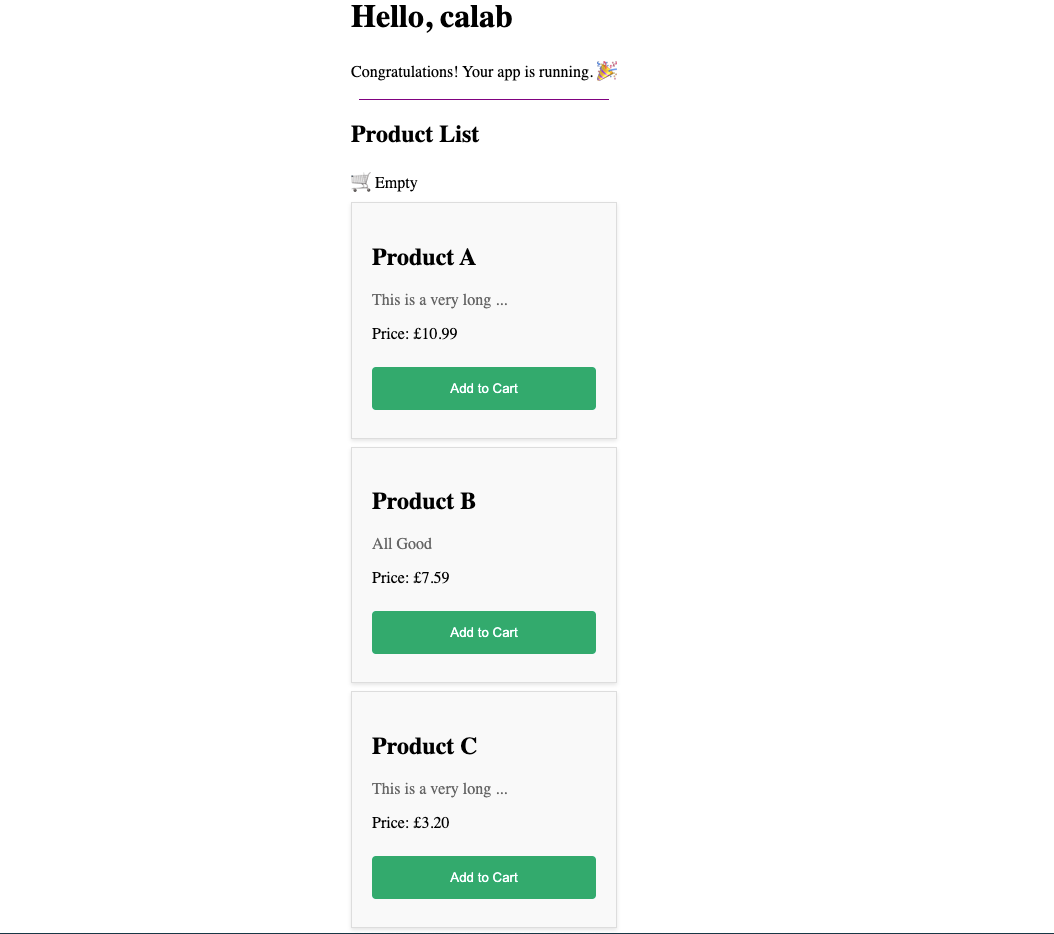
1. You should see a message in your Terminal confirming the npm packages were installed successfully: 

### 1.1 Start The Application

1. Start Angular Development Server if not yet started:

* npx -p @angular/cli ng serve --host 0.0.0.0
* *Otherwise refresh the browser tab to see updated view.*

1. You should see the following getting rendered in your browser:

* 
* *You should see similar view to where you left off in previous lab.*

## 2. Custom Attribute Directive​

### 2.1 Create Custom Attribute Directive

* Create a new drirective called ZoomOnHoover using CLI:
* npx -p @angular/cli ng generate directive directives/zoom-on-hoover
* Open src/app/directives/zoom-on-hoover.directive.ts file and add the following code:
* export class ZoomOnHooverDirective {  
   constructor(private elRef: ElementRef) {}  
    
   @HostListener('mouseover')  
   onMouseOver() {  
   this.elRef.nativeElement.style.transform = 'scale(1.03)';  
   }  
    
   @HostListener('mouseout')  
   onMouseOut() {  
   this.elRef.nativeElement.style.transform = 'scale(1)';  
   }  
  }
* *This directive uses the @HostListener decorator to listen for the mouseover and mouseout events on the host element. When the pointer hovers over the element decorated with this directive, the directive scales it up by 3% using the CSS transform property. When the pointer leaves the card, the directive resets the scale back to its original value.*

### 2.2 Import Attribute Directive Into Component

1. Open src/app/components/product/product.component.ts file and import ZoomOnHoover directive :

* imports: [CurrencyPipe, TruncatePipe, ZoomOnHooverDirective],

1. Open src/app/components/product/product.component.html file and and apply ZoomOnHoover directive to an element.​:

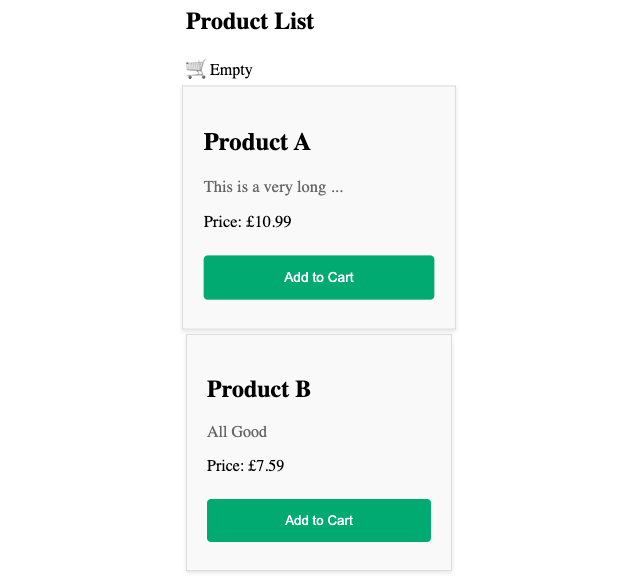
* <div class="product-details" appZoomOnHoover>

### 2.2 Instpect Changes

1. Start Angular Development Server if not yet started:

* npx -p @angular/cli ng serve --host 0.0.0.0
* *Otherwise refresh the browser tab to see updated view.*

1. You should see the following getting rendered in your browser:

* 

## 3. Custom Structural Directive​

### 3.1 Create Custom Attribute Directive

* Create a new drirective called Tooltip using CLI:
* npx -p @angular/cli ng generate directive directives/tooltip
* Open src/app/directives/tooltip.directive.ts file and add the following code: ```.js @Directive({ selector: ‘[appTooltip]’, standalone: true }) export class TooltipDirective {
* @Input(‘appTooltip’) tooltipText!: string; tooltipElement = this.renderer.createElement(‘span’);
* constructor(private el: ElementRef, private renderer: Renderer2) { this.renderer.listen(this.el.nativeElement, ‘mouseover’, () => { this.showTooltip(); }); this.renderer.listen(this.el.nativeElement, ‘mouseout’, () => { this.hideTooltip(); }); }
* showTooltip() { this.renderer.appendChild(this.el.nativeElement, this.tooltipElement); this.renderer.setProperty(this.tooltipElement, ‘textContent’, this.tooltipText);
* const tooltipStyle: any = {  
   'position': 'absolute',  
   'padding': '10px',  
   'border-radius': '5px',  
   'background-color': '#333',  
   'color': '#fff',  
   'font-size': '14px'  
   };  
   Object.keys(tooltipStyle).forEach(style => {  
   this.renderer.setStyle(this.tooltipElement, style, tooltipStyle[style]);  
   });
* }
* hideTooltip() { this.renderer.removeChild(this.el.nativeElement, this.tooltipElement); } } ```
  + The @Input(‘appTooltip’) tooltipText!: string; decorator allows the tooltip text to be passed in from the template.
  + In the constructor, two event listeners are set up for the element: mouseover and mouseout. When these events occur, the showTooltip() or hideTooltip() methods are called respectively.
  + The showTooltip() method creates a new HTML span element (this.tooltipElement) and appends it to the host element using the Renderer2’s appendChild() method. It then sets the text content of the tooltip element using setProperty(), and applies some styles to the tooltip element using the setStyle() method.
  + The hideTooltip() method removes the tooltip element from the host element using the Renderer2’s removeChild() method.

### 3.2 Import Structural Directive Into Component

1. Open src/app/components/product/product.component.ts file and import Tooltip directive :

* imports: [CurrencyPipe, TruncatePipe, TooltipDirective],

1. Open src/app/components/product/product.component.html file and and apply Tooltip directive to an element.​:

* <p [appTooltip]="product.description" ...

### 3.3 Instpect Changes

1. Start Angular Development Server if not yet started:

* npx -p @angular/cli ng serve --host 0.0.0.0
* *Otherwise refresh the browser tab to see updated view.*

1. You should see the following getting rendered in your browser:

* 