

## Minutes of session Day 20: Angular Module(Directives and Pipes)

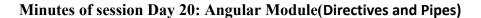
	Advanced Angular Concepts			
	j j			
		Structural directives : *ngIF, *ngFOR		
		Changes the DOM layout by adding/removing elements.		
		Attribute directives : *ngClass, *ngSyle		
		Change the appearance or behavior of an element.		
		Benefits of Built-in directives (nglf, ngFor, ngClass, ngStyle):		
		Increases reusability of UI logic.		
		Reduces boilerplate code.		
		Improves the readability by keeping HTML clean		
		<u>Limitations:</u>		
		Overuse can make template harder to debug		
		Complex logic inside directives may reduce maintainability.		
		Best Practices:		
	Angular Directives and Pipes	Use structural directives for layouts changes, attribute directives for styling and		
	Directives are classes in angular,	behavior.		
		Keep directives logic small and reusable.		
	to elements in DOM.	Avoid putting complex business logic in templates		
Day 20				
		Custom directives and their use:		
		<u>Various scenarios of custom directives are :</u>		
	Custom (user define)directives	Auto-focus on input field.		
	allow us to create reusable	Custom hover effect		
	behavior not available in a	Validation behavior		
	regular built-in set.	Pro :		
		Promote code reusability.		
		Allow adding specific UI/UX behaviour without modifying the component.		
		Cons:		
		Increases complexity if overused.		
		Harder to main compared to simple CSS or service logic.		
		Back and attack		
		Best practices:		
		Use for reusable UI patterns only.  Name clearly to reflect the directive's purpose.		
		Keep them decoupled from specific components.		
		and the second s		
		Using Angular Pipes for data transformation		
		Pipes transform data before displaying it in the view.		
		Ex:		



## Minutes of session Day 20: Angular Module(Directives and Pipes)

Trimutes of session Day 20. Tringular	
	Currency -> formatting like INR, \$
	Date -> MMDDYY, DDMMYY etc
	Json - > pretty prints JSON
	uppercase/lowercase : transformation
	Pros:
	Keep the template clean.
	Improves readability.
	Reusable across templates.
	Comer
	Cons:
	Overuse in template can impact performance.  Pure pipes are more performant, impure pipes can cause re-rendering issues.
	Pure pipes are more performant, impure pipes can cause re-rendering issues.
	Best practices :
	Use pure pipes for static transformations.
	Move heavy computations logic outside pipes.
	Create custom pipes for repetitive data transformations.
	Case study:
	Application product-demo that show list of products
	1. Display products with *ngfor
	2. use*ngclass for marking out of stock items
	3. Create a custom pipe priceFormat to format product prices
	4. Create acustoke directive appHighlight that highlights a product card on
	hover
	5. "Buy" button should be disabled when an item is out of stock.
	Step 1: Create a project adding components like
	Step 2: Genre component product-lis
	Step 3: Generate pipe price-format
	Step 4: Generate Directives app-Highlight
	Step 5: Creating an interface Product in app - module.ts file
	Step 6: So we can reuse them in our <u>component.ts</u> file
	Step 7: Create presentation logic in component.html and css logic in .css file
	Practical exercises: Applying directives and pipes in a real-world example
	Inline, external, and scoped styles
	Best practices for styling Angular components
Component Styling and	Component Lifecycle Methods
Communication	Understanding the lifecycle hooks (ngOnInit, ngOnChanges, etc.)
	Practical exercises: Implementing lifecycle hooks for dynamic behavior
	<u> </u>

- 1. How to implement Jquery with HTML and CSS, differentiating it from bootstrap ??
- 2. Demo based on Design patterns using C# and comparing it with javascript.





## What is JQuery?

- 1. A fast, small and feature rich Javascript library.
- 2. Simplifies DOM manipulation, event handling, AJAX request and animation
- 3. Mainly used for behaviour and logic in web pages.

Feature / Aspect	jQuery	Bootstrap
Туре	JavaScript library	CSS framework (with optional JS)
Main Purpose	DOM manipulation, events, AJAX, animations	Responsive design, styling, UI components
Language Base	JavaScript	CSS, HTML (with JS plugins)
Focus Area	Functionality & behavior	Layout & styling
Learning Curve	Moderate if you know JS	Easy if you know HTML/CSS
Dependencies Pure JS (no dependencies)		Can optionally use jQuery for JS components (Bootstrap 3 & 4) but Bootstrap 5+ does not require it
<b>Example Use</b>	Hide/show elements, form validation, fetch data	Create a navbar, responsive grid, styled buttons
When to Use	You need to handle user interactions and modify DOM dynamically	You need quick, responsive, mobile-friendly design