**Questions & Assignments List**

1 Introduction To Component Lifecycle

the component lifecycle refers to the **various stages** a component goes through from its creation to its destruction.

The lifecycle hooks provided by Angular allow developers to **tap into these stages** and **execute custom logic**.

1. **ngOnChanges**: This hook is called **when** any of the component's input properties **change**. It receives a **SimpleChanges** object containing previous and current values of the input properties.

2. **ngOnInit**: This hook is called once, **after** the component has been **initialized** and its input properties have been set. It's often used **for initialization logic, such as fetching data from a backend service.**

ngDoCheck: This hook is called during every change detection cycle. It's used for custom change detection and performance optimization.

ngAfterContentInit: This hook is called after Angular projects external content (such as content passed into a component via <ng-content>). It's often used for initialization that relies on the component's content being fully initialized.

ngAfterContentChecked: This hook is called after Angular checks the content projected into the component.

ngAfterViewInit: This hook is called after the component's view (and child views) has been initialized. It's often used for initialization that relies on the component's view being fully initialized.

ngAfterViewChecked: This hook is called after Angular checks the component's view (and child views).

**ngOnDestroy**: This hook is called just before the component is destroyed. It's used for cleanup logic, such as unsubscribing from observables and detaching event listeners, to prevent memory leaks.

2 Component Creation Hook ngOnChanges

2.1 What is the purpose of the ngOnChanges hook in Angular?

When you implement the ngOnChanges hook in your component class, Angular will call this method whenever any of the input properties bound to the component change. The hook receives a SimpleChanges object as an argument, which contains information about each changed property.

3 Understanding SimpleChanges Object

3.1 What is SimpleChanges in Angular, and what is its purpose?

ngOnChanges ( changes : SimpleChange) {}

4 Component Creation Hook ngOnInIt

4.1 Explain the difference between constructor() and ngOnInit() hook

5 Detecting Custom Changes With ngDoCheck hook

5.1 Explain how the ngOnChanges() is different from ngDoCheck() hook.

6 Understanding the ngAfterContentInit() hook

7 Checking The Content With ngAfterContentChecked() hook

8 Loading Content With ngAfterViewInit() hook

8.1 What will be the difference if we apply the same code logic inside the ngOnInit() hook instead of ngAfterViewInit()?

9 Checking The Component View With ngAfterViewChecked() hook

10 Destroying Component With ngDestroy() hook