Experiment No.: 06

Title: Create a custom component using angular.

Objectives:

1. To study creating components using angular during building the web app.

Theory:

Components are basically classes that interact with the .html file of the component, which gets displayed on the browser. The file structure has the app component and it consists of the following files −

- app.component.css
- app.component.html
- app.component.spec.ts
- app.component.ts
- app.module.ts

The above files were created by default when we created new project using the angular-cli command.

Component metadata-

The **@Component** decorator identifies the class immediately below it as a component class, and specifies its metadata. The metadata for a component tells Angular where to get the major building blocks that it needs to create and present the component and its view. In particular, it associates a template with the component, either directly with inline code, or by reference. Together, the component and its template describe a view. In addition to containing or pointing to the template, the @Component metadata configures, for example, how the component can be referenced in HTML and what services it requires.

How to create a Component in Angular-

1. Import Component Decorator-

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

This imports the component decorator from angular/core directory.

2. Use Decorator and Add Metadata-

Use the @Component decorator by passing it some data & then export component class. Decorators are a design pattern. They are an alternative to subclassing.

```
@Component({
            selector: 'demo-root',
            template: `<h1>Hello World</h1>` })
export class MyComponent {}
```

We've informed Angular that the class MyComponent is an *Angular Component*. Now, Angular will treat it like a component. It will use all the metadata we pass to **augment** the MyComponent class. Our class is now an Angular component.

3. Specify Template-

An Angular component is **a directive with a template**. No template, no component. There are two ways to add a template to component –

a) Inline Template

Use this when the template code is less than (or equal to) 3 lines.

b) External Template File

Use this when template code is more than 3 lines. It keeps separates the logic from the presentation making the code more maintainable.

4. Add Styles-

This is optional. Components don't need styles to work. They work a lot like templates. There are two wats to include styles in template-

a) Inline Styles
b) External Style Files

5. Export the Component-

The component cannot be used without exporting it.

```
@Component ({…..})
export class MyComponent{}
```

The component also needs to be a part of a module. So it can be added as a declaration of NgModule

Key Concept: Component, Decorator, Template