**STATIC RESOURCES**

Add to the assets folder

**COMPONENTS**

* All Angular apps are made of multiple different components
* Each component is designed to implement one “thing” in the app that is visible on the screen
* A component wraps up all the HTML and code to make one little widget work correctly
* A component can be *reused* multiple times in the same application
* Components can be *nested* or shown inside each other
* Every app has a component called the “App” component and it is always the top parent component
* Each component has its own Component Class, Component Template, Component CSS File and Spec File

**NEW COMPONENT**

ng generate component <name>

**COMPONENT NESTING**

Enter the following inside the parent component:

<app-componentName></app-componentName>

**DATA FROM PARENT TO CHILD COMPONENT**

* Specify what data will be passed to the child in the parent component, e.g. an object array
* Pass the data by Property binding in the child component

<app-componentName

    [childProperty] = "data from parent(expression)"

></app-componentName>

When dealing with multiple objects, use \*ngFor

<app-componentName

    \*ngFor = "let item of items"

    [childProperty] = "item.property"

></app-card>

- Map the properties inside the child component:

import { Input } from '@angular/core';

Inside class:

@Input() parentPropertyName = '';

* Use the properties inside the Child Component Template

**COMPONENT CSS**

The component selector can be used as a css selector only in its parent class.

Inside itself, the ***:host*** selector must be used for self-reference.

**CONTENT**

We can use the <ng-content> syntax inside a component’s html template. When the component is used, the <ng-content> is substituted with any markup that appears inside the component html declaration. The declaration can be used once, though.

<app-componentName>HTML text or markup that replaces the <ng-content> </app-componentName>