Services in Angular

A **service** in Angular is a reusable piece of code that performs a specific task and is typically used to **share logic** or **data** across different components.

Think of it like a helper or a "middleman" that centralizes functionality in your app. Instead of writing the same logic in multiple places, you put it in a service and use it wherever needed.

For example:

- Fetching data from an API.
- Storing user information.
- Logging errors or debugging info.

Services are usually **singletons**, meaning one instance is shared throughout the app.

Dependency Injection (DI)

Dependency Injection is a design pattern Angular uses to provide **dependencies (services or objects)** to components or other parts of the app when they need them.

In simple terms:

- Instead of creating a new object or service manually (e.g., `new MyService()`), Angular **injects** the instance automatically wherever required.
- This makes your app more flexible, reusable, and easier to test.

How It Works in Angular

Create a Service
You define a service (a class) with reusable logic:

```
@Injectable({
  providedIn: 'root', // Makes the service available app-wide
})
export class MyService {
  getData() {
    return 'Hello from the service!';
  }
```

```
}
```

Inject It into a Component
Angular provides this service to components that need it:

```
import { MyService } from './my-service.service';

@Component({
  selector: 'app-example',
  template: '<p>{{ message }}</p>',
})
export class ExampleComponent {
  message: string;

  constructor(private myService: MyService) {
    this.message = this.myService.getData();
  }
}
```

1. **Result**
   The service handles the logic, and the component just "asks" the service for what it needs.

Why Use Services and Dependency Injection?

- **Reusability**: Write logic once in a service, use it anywhere in the app.
- **Separation of Concerns**: Keep components clean by moving heavy logic to services.
- **Testability**: Mock services in tests without worrying about implementation details.
- **Efficiency**: Angular ensures only one instance of a service is created and shared (singleton).

So, **services handle logic** and **dependency injection makes it easy to use them wherever needed.**