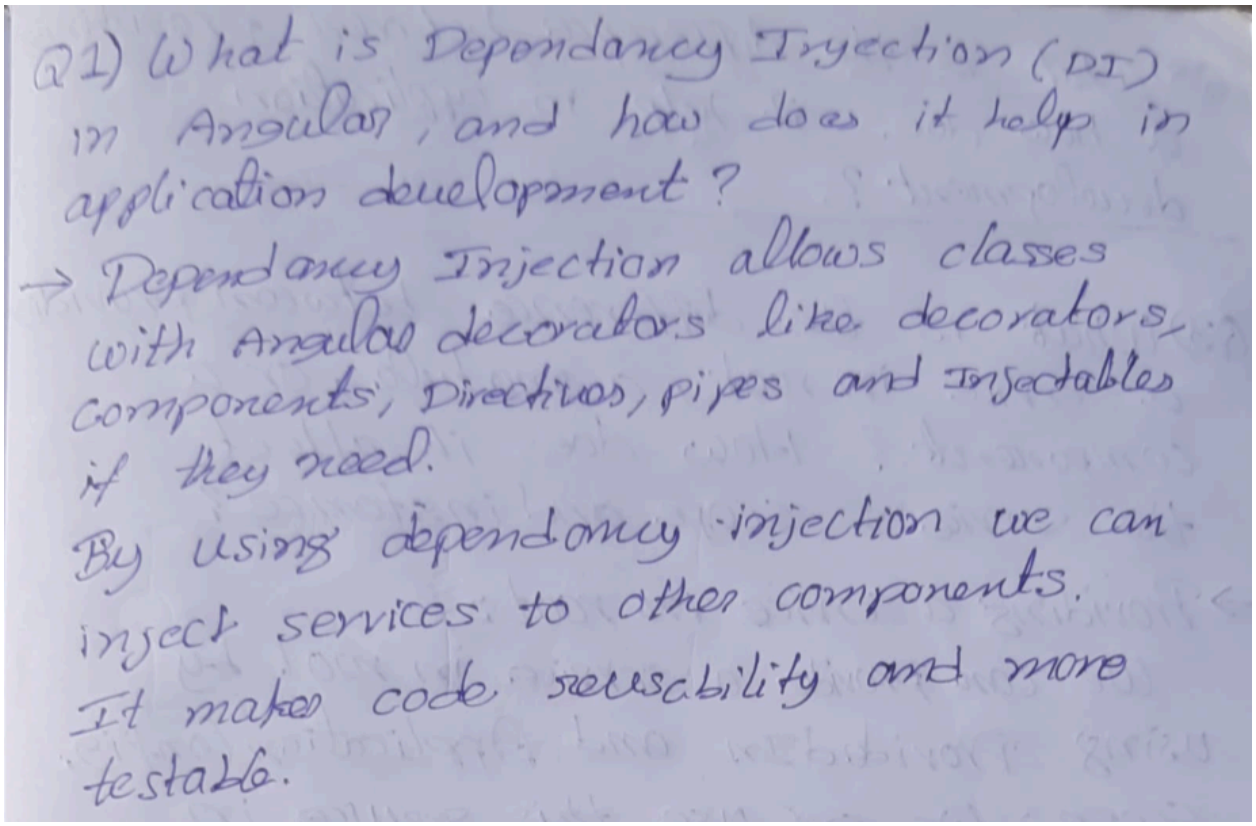
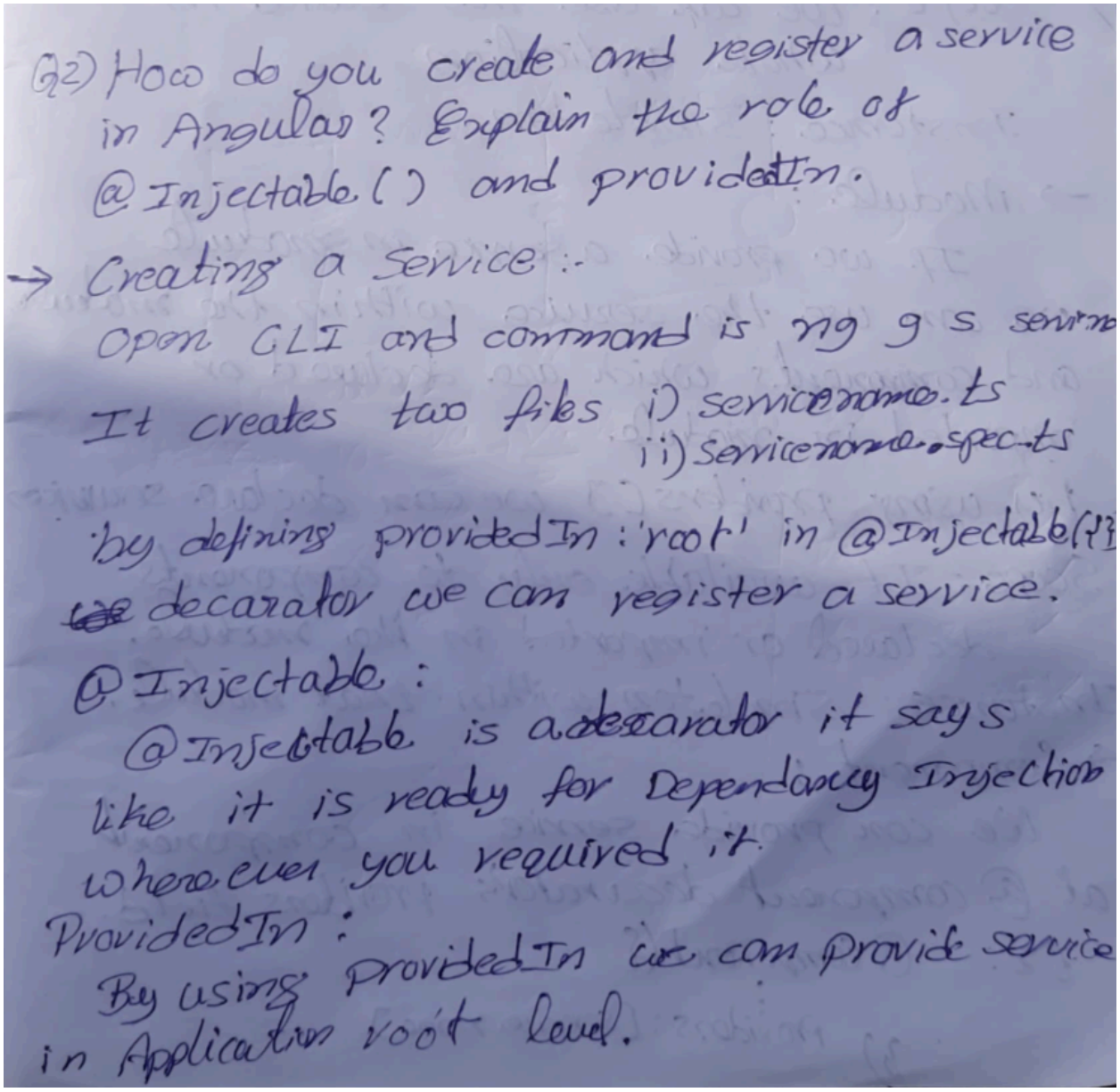
1) What is Dependency Injection (DI) in Angular, and how does it help in application development?



2) How do you create and register a service in Angular? Explain the role of @Injectable() and providedIn.



3. What is the difference between providing a service in root, a module, or a component? How does it affect the service's scope and instance?

Q3) What is the difference between providing a service in root, a module, or a component? How does it affect the service's scope and instance?

→ Providing a service in root:

We can provide a service in root by using ProvidedIn and ApplicationConfig.

Scope: We can use this seeurce in whole application

Instance: Singleton.

→ Module:

If we provide a servce in module we can use the seruice within the module and components which are declared or imported in module.

by using providers[] we can declare seruices.

Scope: It available only to components declared or imported in the module.

Instance: Singleton within that module.

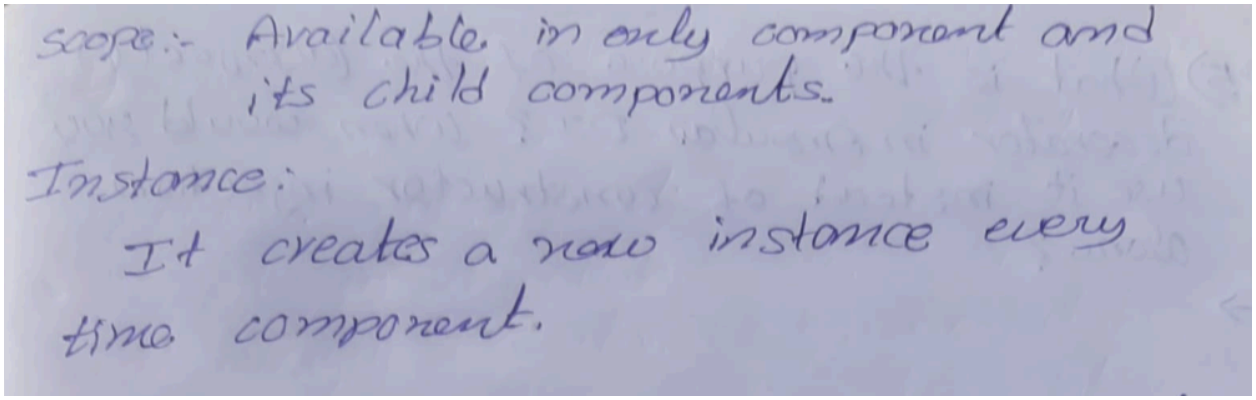
→ Component:

We can provide servrice in componeut at @component decaratar's providers field.

Ex:- @Component({

providons: [service name]

3)



4. Explain the concept of hierarchical injectors in Angular with an example.

Q4) Explain the concept of hierachical injectors in Angular with an example.

→ Angular creates tree like structure injectors hierarchical way.

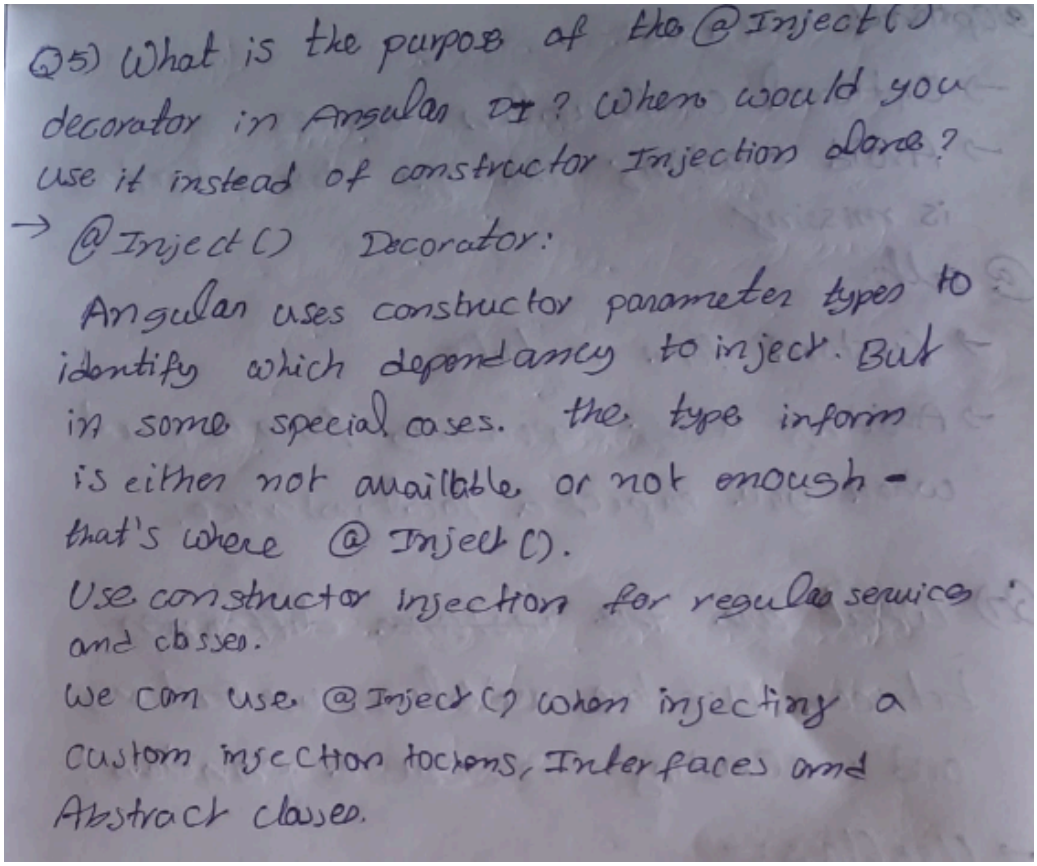
→ Angular starts looking for a dependancy from the component' injector and moved up the tree like child → parent → root

Example:-

Root
↓
Weather service A
↓
Weather service B

It starts checking if the service is availabe in Weather service B or not. If not it will go for its parent service If not present it will search in Root level.

5. What is the purpose of the @Inject() decorator in Angular DI? When would you use it instead of constructor injection alone?

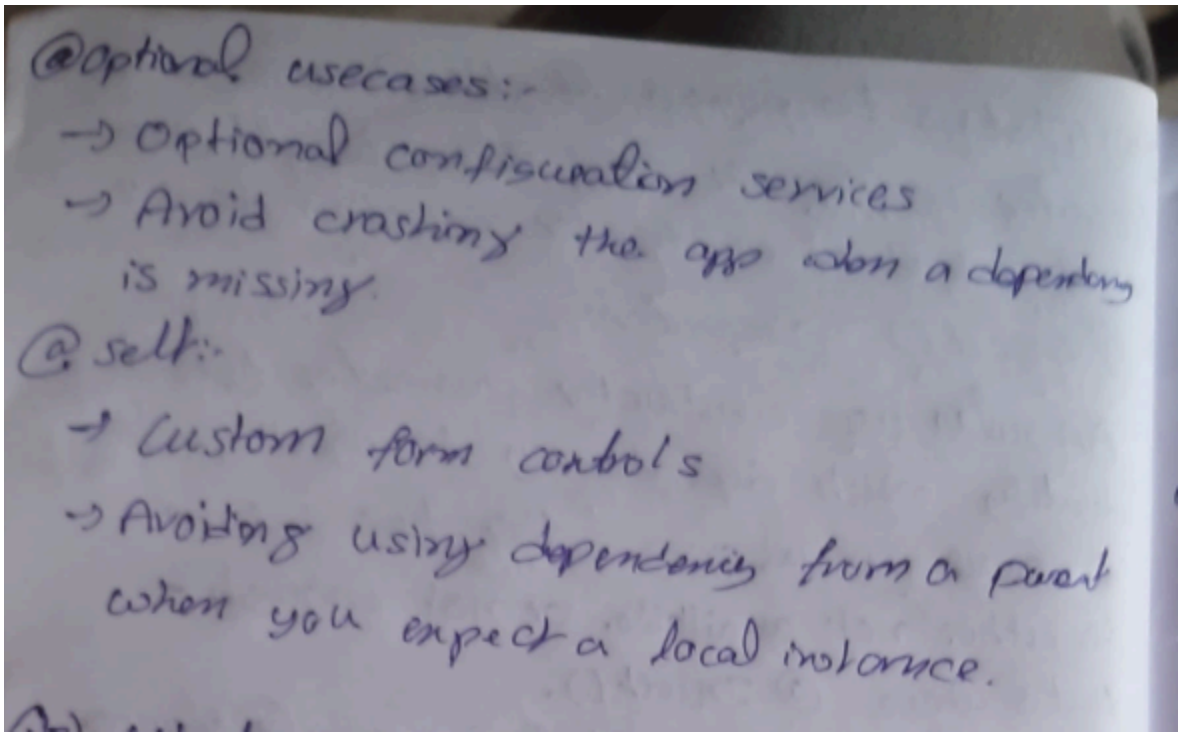


6. What is the difference between @Optional() and @Self() in Angular dependency injection? Provide use cases.

Q6) What is the difference between @Optional and @Self() in Angular dependency Injection?

→ @Optional is allows injection of dependency if available. If it is not available it returns null, instead of throwing an error.

@self is restricts angular to look only in the current injector. It will throw an error if not found at the current level.



7. What are the differences between using useClass, useValue, useFactory, and useExisting in service providers? Provide examples.

Q7) What are the ~~instance~~ differences between Useclass, useValue, useFactory and useExisting in service providers?

→ UseClass :-

Default class based provider

Providers a class by new keyword.

Provides a service by creating a new instance of a class.

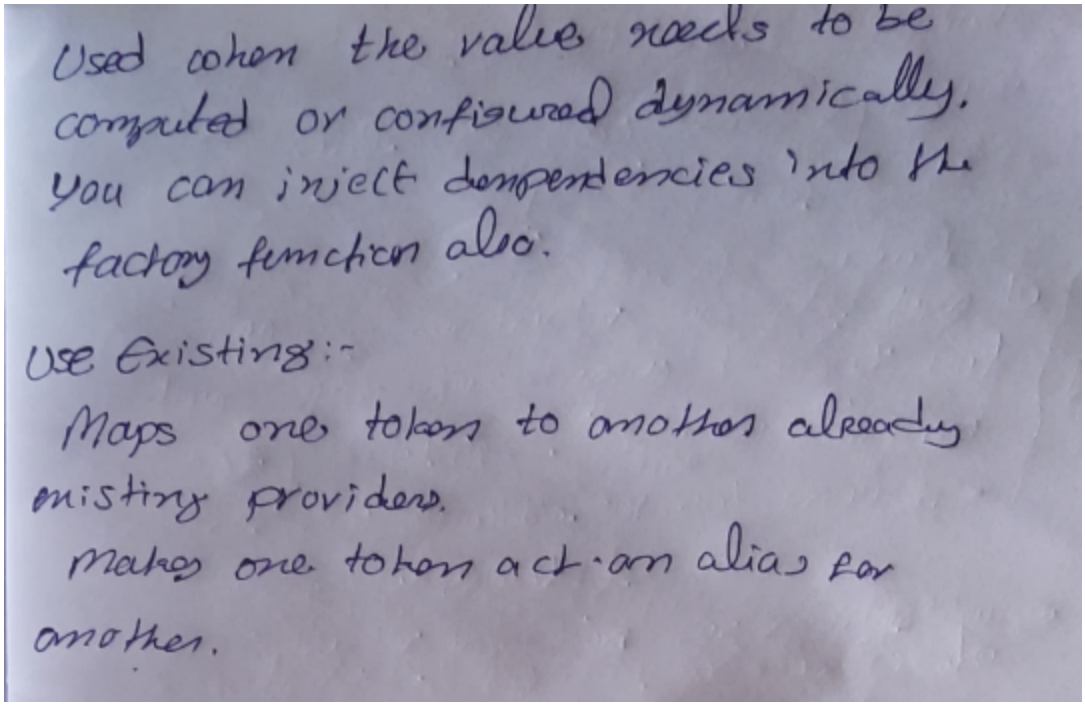
UseValue:

Provides a fixed value (object, string, number)

Use for configurations constants or static data.

UseFactory :-

Uses a factory function to returns the value or object



8. How can you override a service provided in the root injector at a component level? What are the implications?

Q8) How can you overiride a service provided in the root injector at a component level? What are the implication.

→ Some times we want a component (or sub) use a different instance or custom implementation of a service separate from global one.

When you want a separate service instance per component.

For testing, mocking, or shadowing the service.

Scope is only the component and its children use the overridden instance.