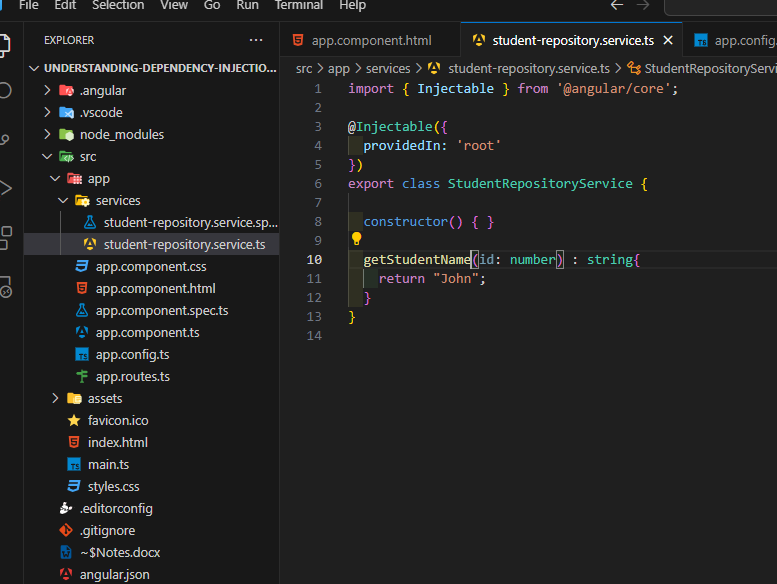
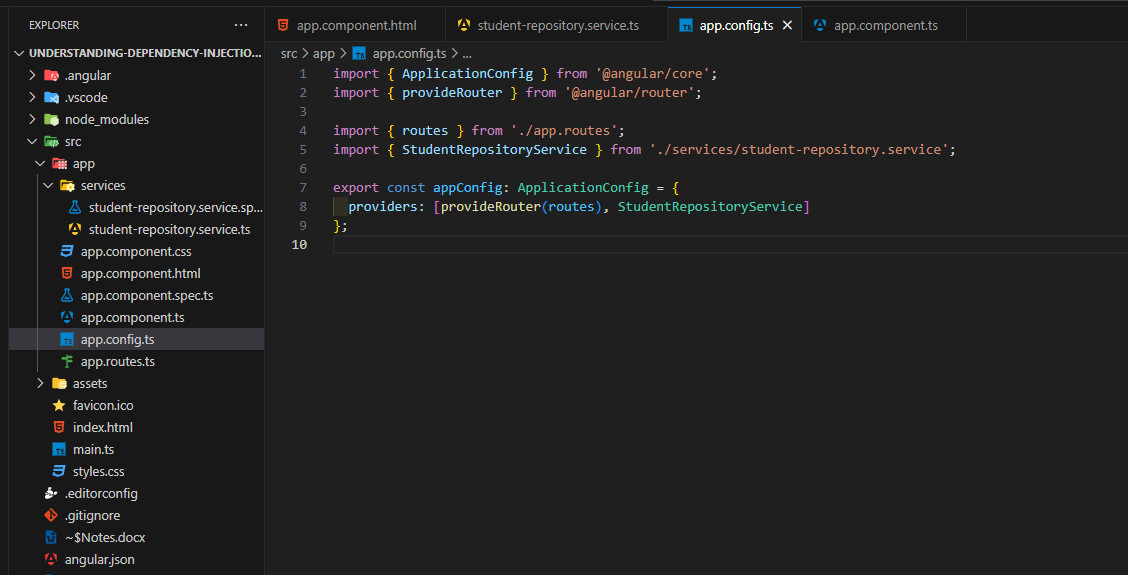
PLAESE SEE COMMITS.

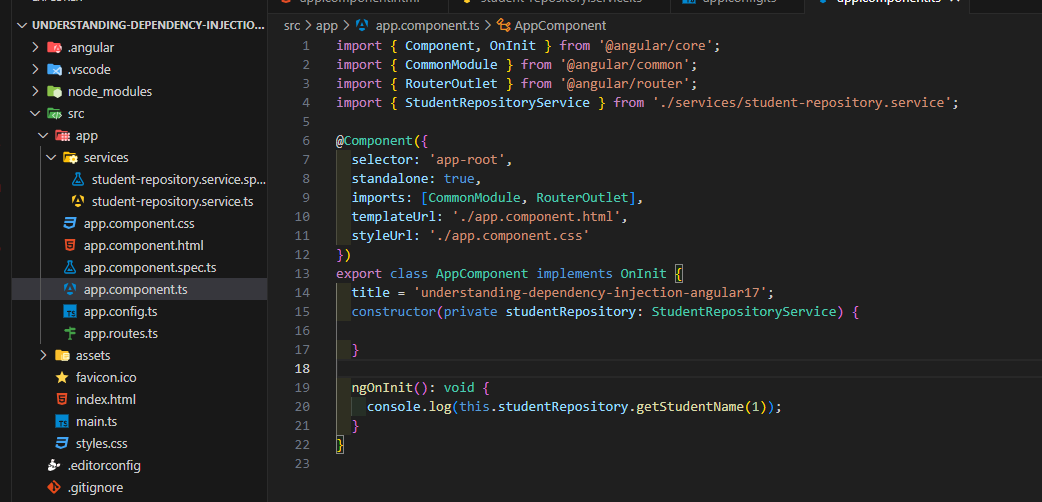
First create a service 🡺 ng generate service services/StudentRepository



Now import “StudentRepositoryService” in app.config.ts file in order to use in every component.



Now inject service in app component using constructor injection.

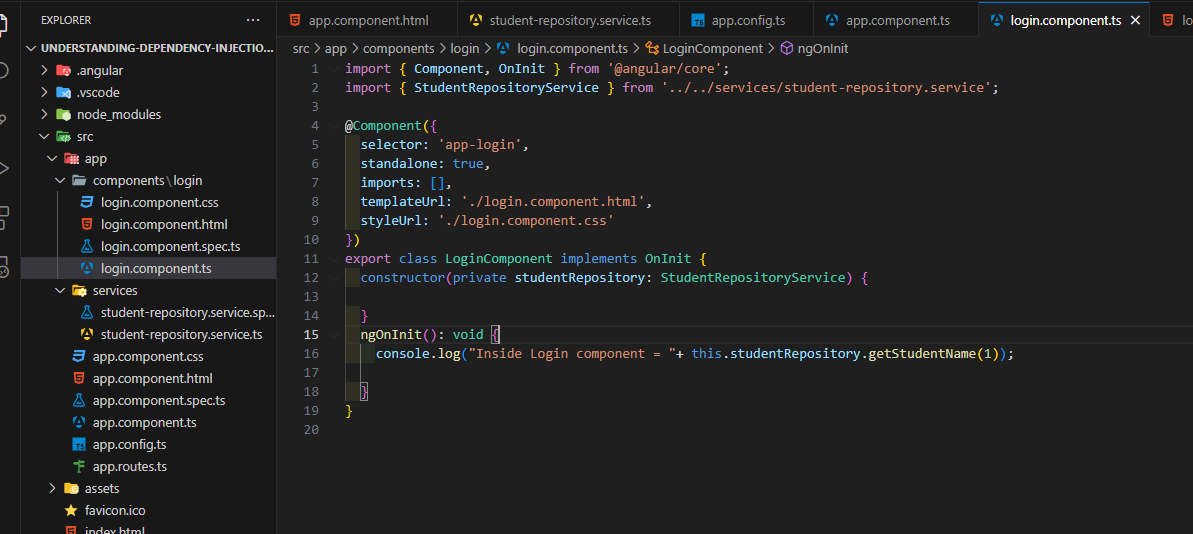


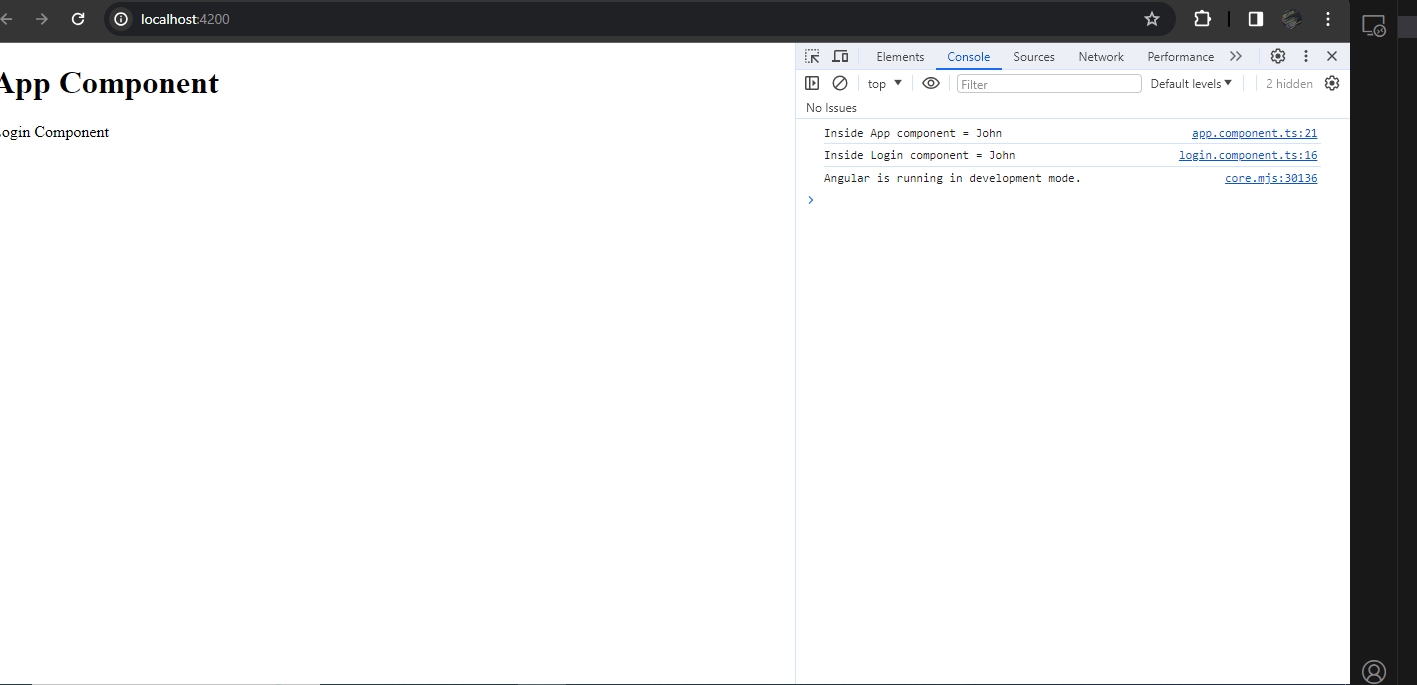
And here you will see the result on chrome console ☺

Now to check whether the service is available in all components, we have to create a new component.

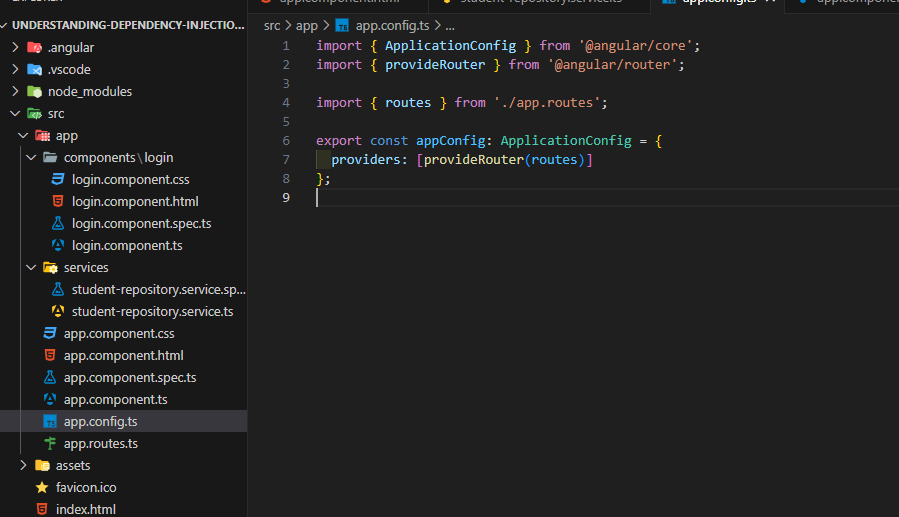
Create a new component 🡺 ng generate component components/Login

Do same things for that component





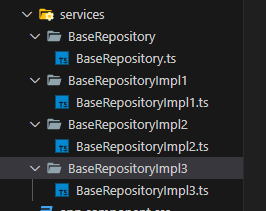
Now we are going to inject dependency in a specific component and not globally.

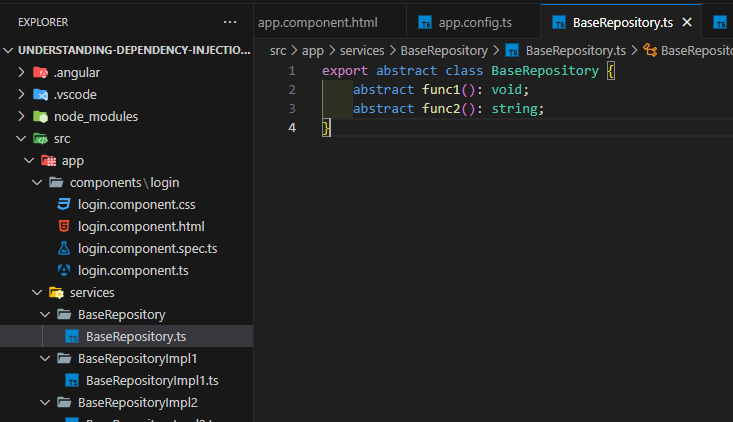


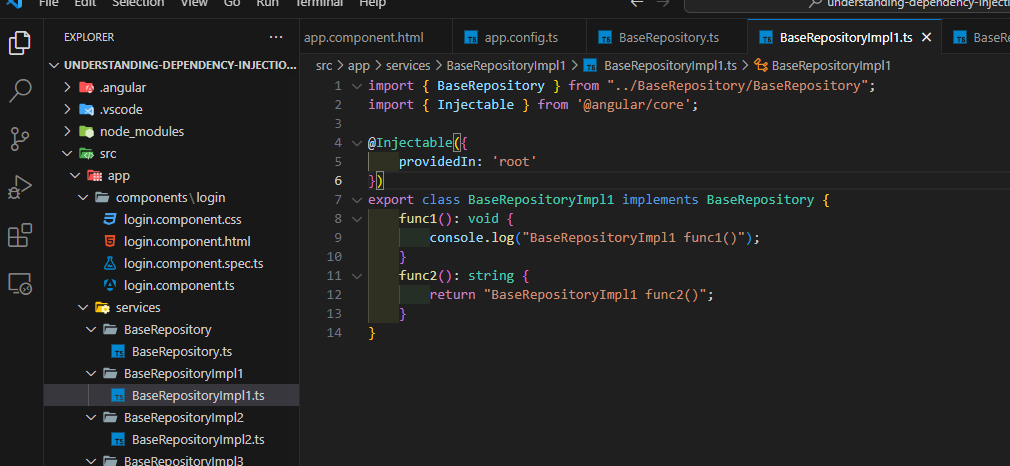
As you see, I removed StudentRepositoryService from providers array And still everything is working fine.

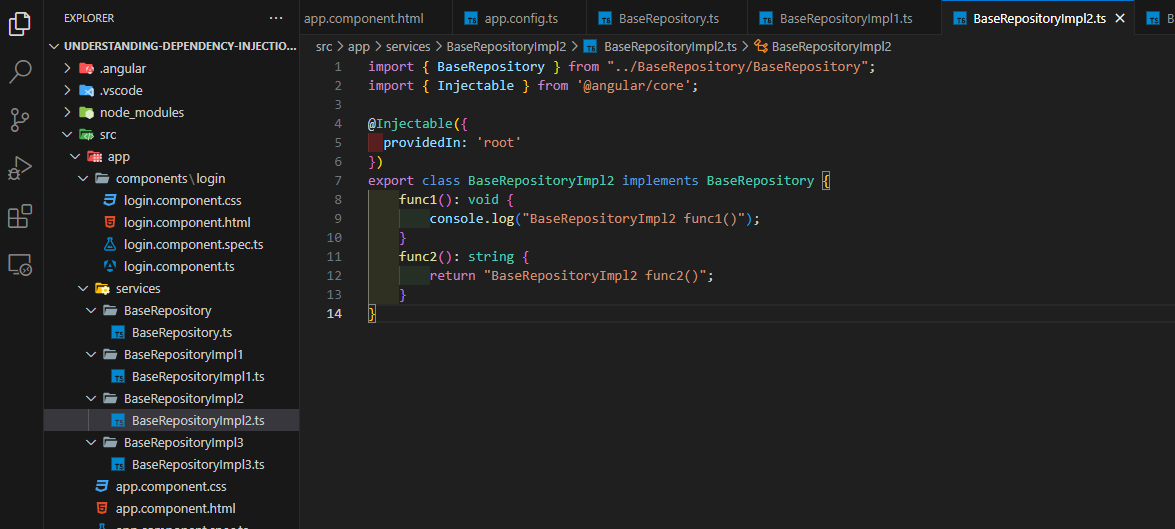
=====================================REPOSITORY PATTERN IN ANGULAR 17==============

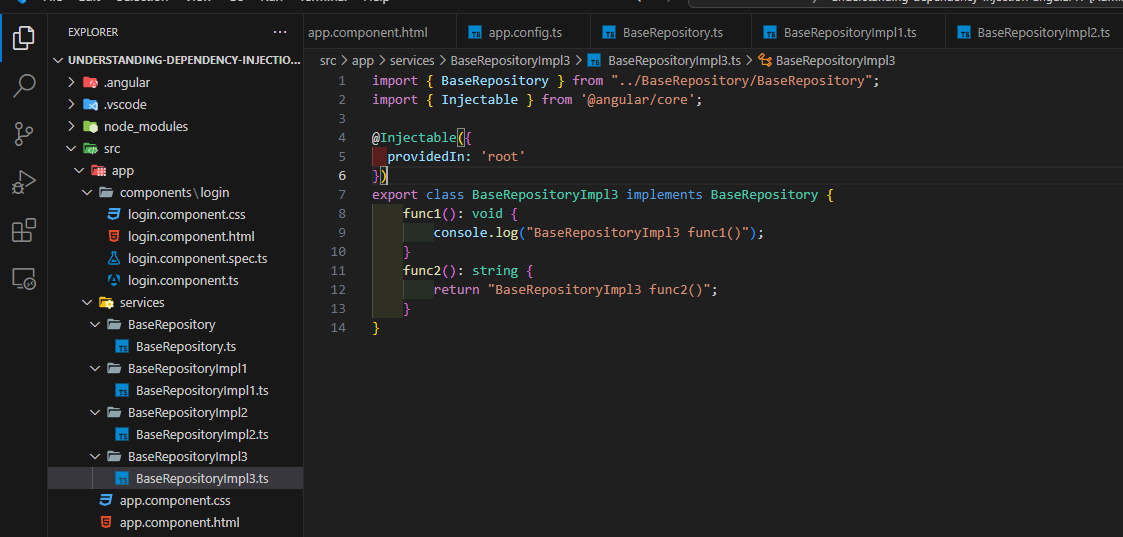
Create following folder hierarchy.



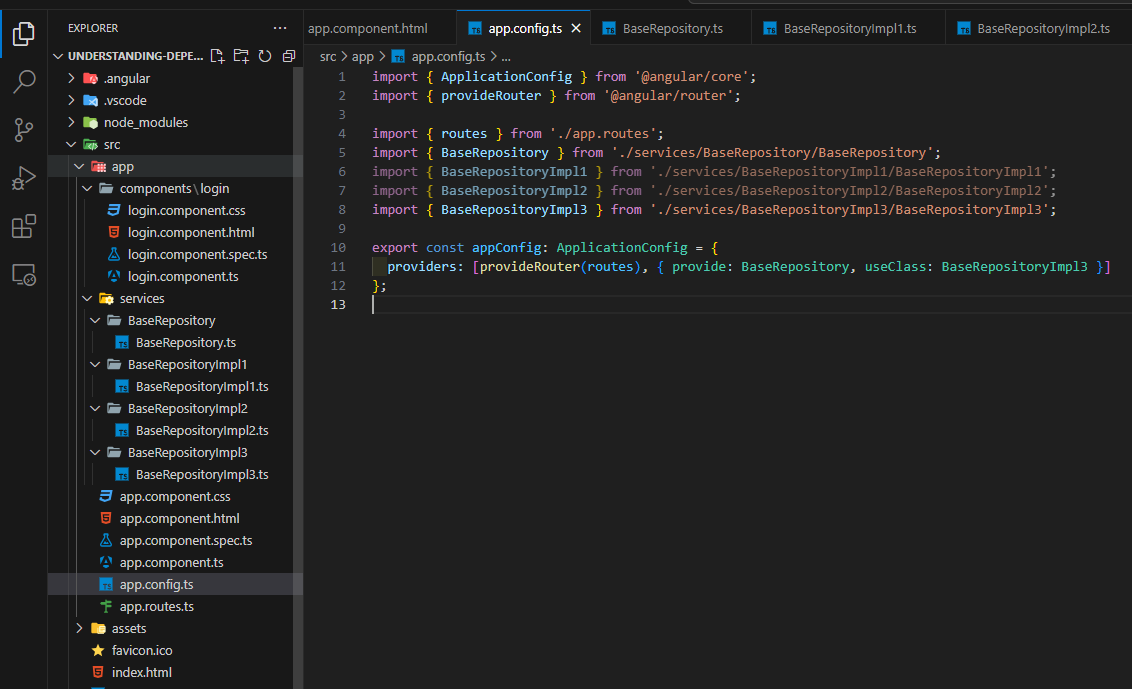


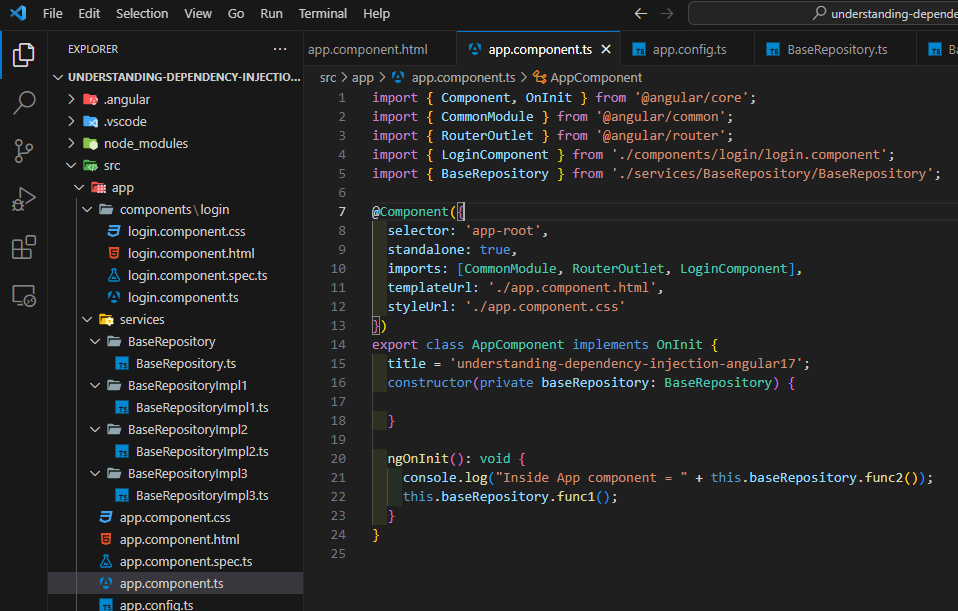


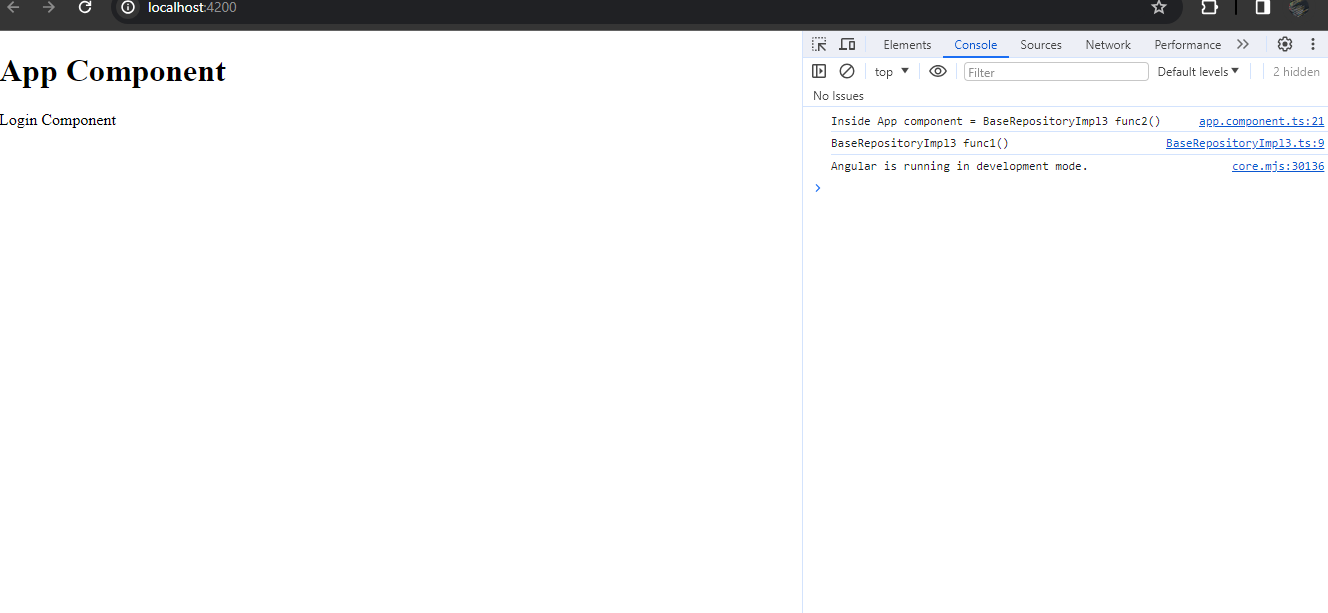




Suppose we want to use BaseRepositoryImpl3 implementation of BaseRepository. Also remove “student-repository-service” as we don’t need them anymore.





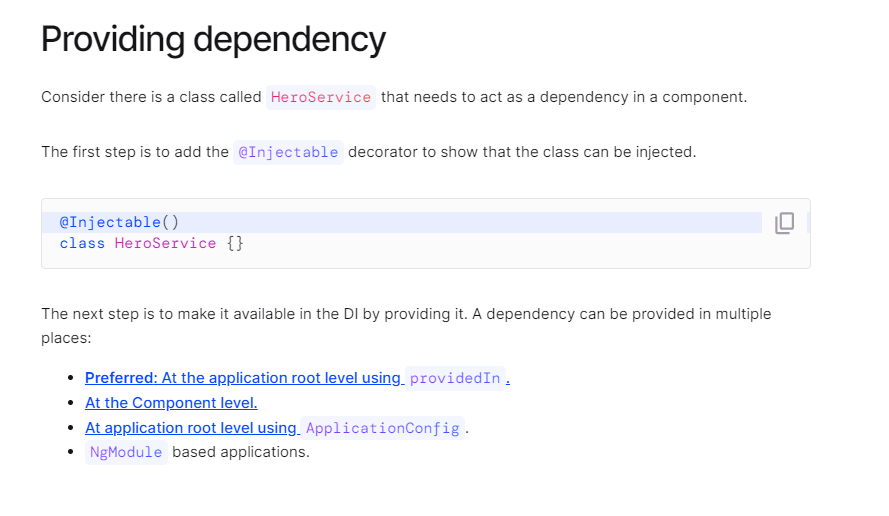


NOW REMOVE EVERYTHING THAT I CREATED AND GO BACK TO THE DEFAULT FOLDER STRUCTURE OF Angular 17 PROJECT.

NOW START LEARNING DOCUMENTATION OF ANGULAR 17.

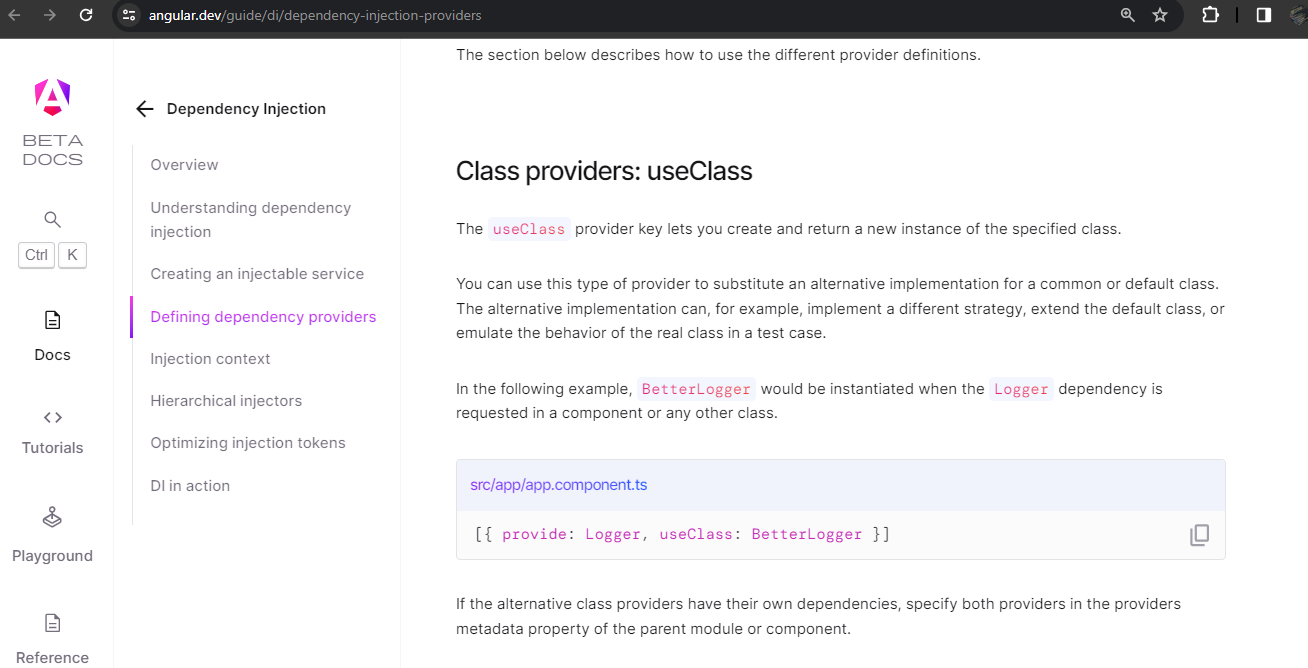
<https://angular.dev/guide/di/dependency-injection>

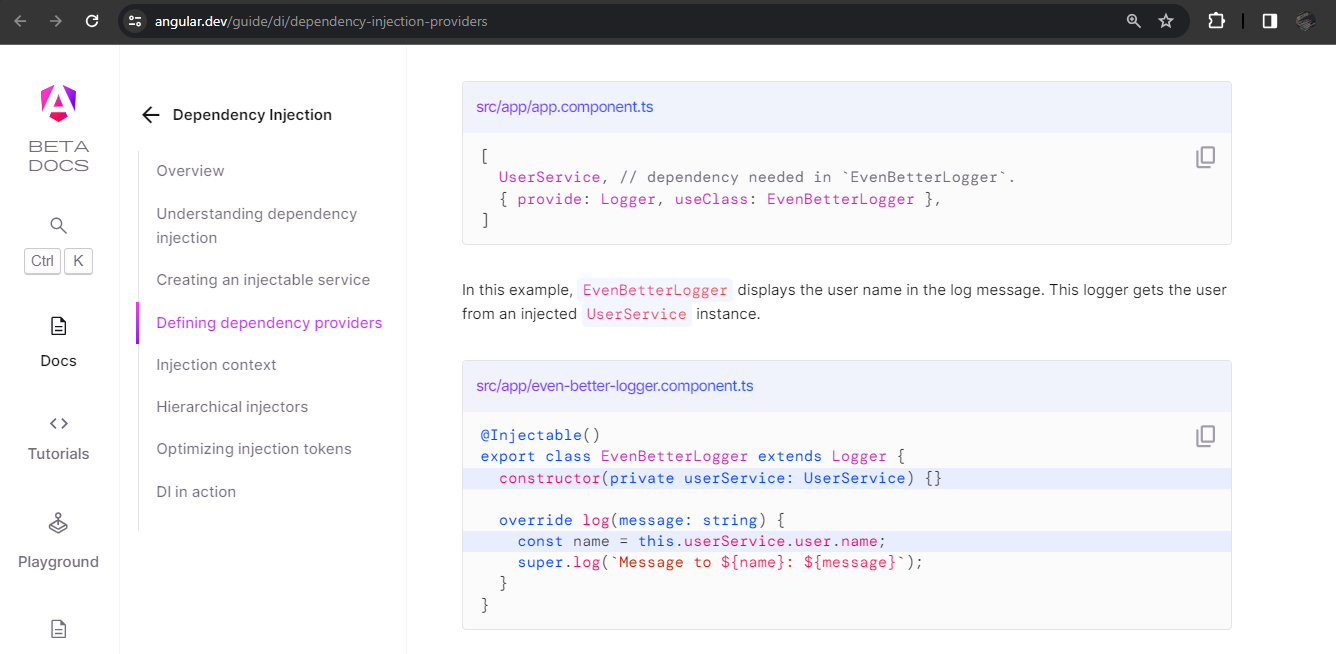
There are 4 ways of providing dependency.



In commit “0bc0679c7bb0375e5703f6de4e58b854de92439c” , I still using services globally and that because of “provieIn” parameter of “Injectable” decorator.

Finally I found something from angular 17 deocumentation that can help me in developing repository pattern.

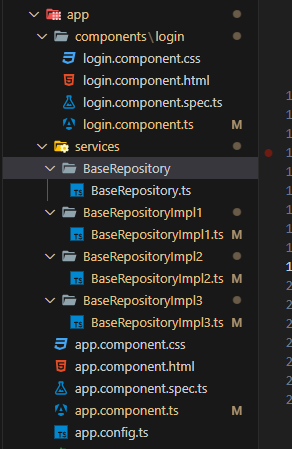


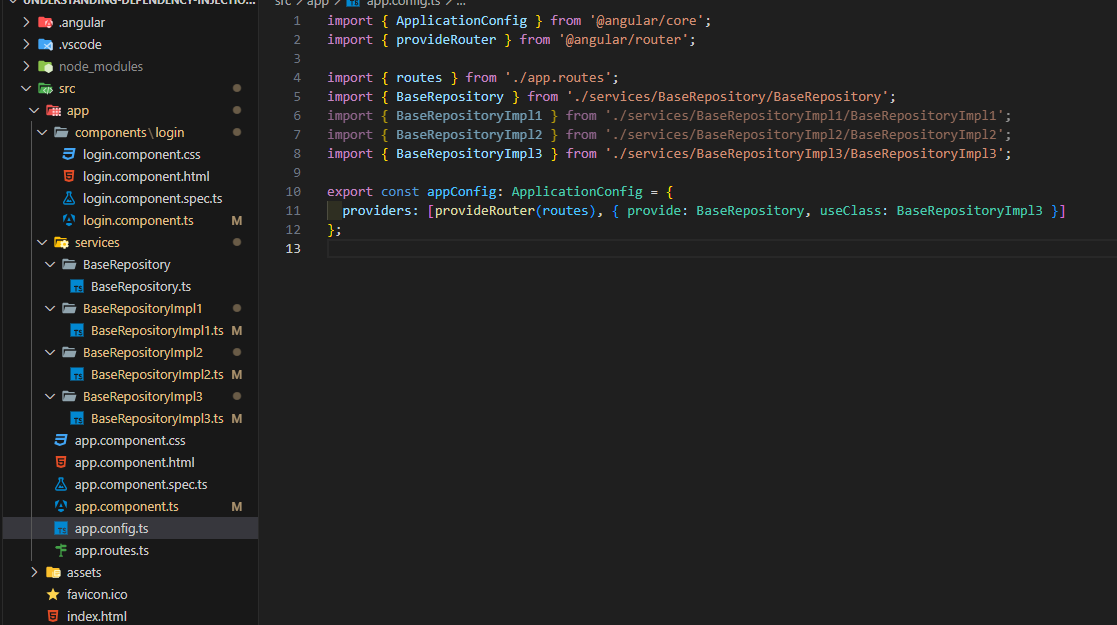


Now create BaseRepository and its 3 impleemntations as I did earlier.

Now remove following line from Injectable decorator of all 3 implementations.



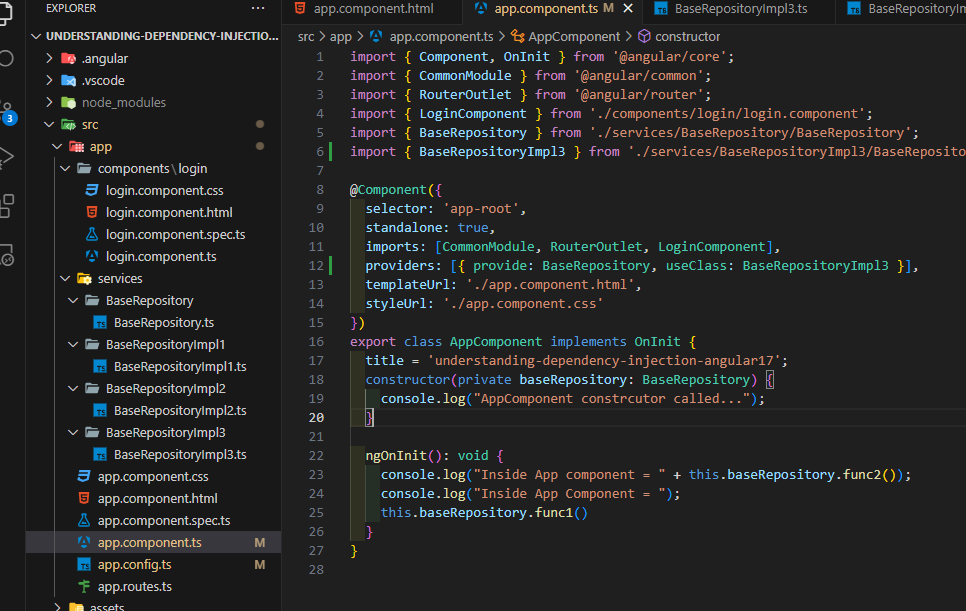




Remember in that scenario, only one object of BasicRepositoryImpl3 was created and use throught the application. To identity this, create a constructor in each implementation of BasicRepository. This is an example of Singleton service.

Now my task is to do inverse of singleton pattern.

I want to provide dependency in only app component.



Provide the service in providers array of Component decorator.

When I did this, LoginComponent that I am using in app component’s template will also got service and only one object of service is created in that case(one call to service constructor ).