**Clean Architecture**

Before we start talking about clean architecture, I think It is fitting to talk about the core principle the architecture depends on, SOLID.

The SOLID principle is a set of five principles that are used in software development to create a maintainable,extensible and easy to understand softwares.

* Single responsibility Principle : Every class, method or block of code should have only a single reason to be changed. In other words It states that the above constructs should have only one objective when they are designed.
* The Open Closed principle : A software should be open to extension but closed for modification. By using ideas like inheritance and abstraction we should allow classes and methods to be easily extended but still make them closed for modification.
* Liskov’s substitution principle : An object that inherits from a base class should be substitutable for the base class.
* Interface segregation : Large and General Interfaces shouldn’t be used in development. Instead these softwares should be broken down into smaller pieces.
* Dependency inversion. Higher level modules shouldn’t depend on lower level implementations. It states that the two layers should communicate through interfaces so that even if our implementation changes we don’t have to change our core business logic.

**What is clean architecture?**

For simple applications it is completely fine to write all the code in a single file. The negative consequence of doing so is only felt as the application becomes bigger and bigger. When this happens it starts to be difficult to include additional features to test and debug the code thoroughly. This is where clean architecture comes into play. By consistently following the SOLID principles it provides organizational schemes and a set of policies that are useful to build a clean, extensible, testable and maintainable software.

**Components of clean architecture**

**Core :** In the clean architecture the core is the central project. The core should never depend on any of the projects in the software.It should be created as its own separate project and it houses the domain entities, the business logic and the contracts of the application

**Infrastructure**: This should also be its own separate project. It is the part of the software where access to database,external apis and system infrastructure are implemented.

**API** : This is the part of the project where we implement our endpoints. In addition it is also the part of the software where presentation layer dots are placed.

**Test** : If we test our software It too should be placed in its own separate project.