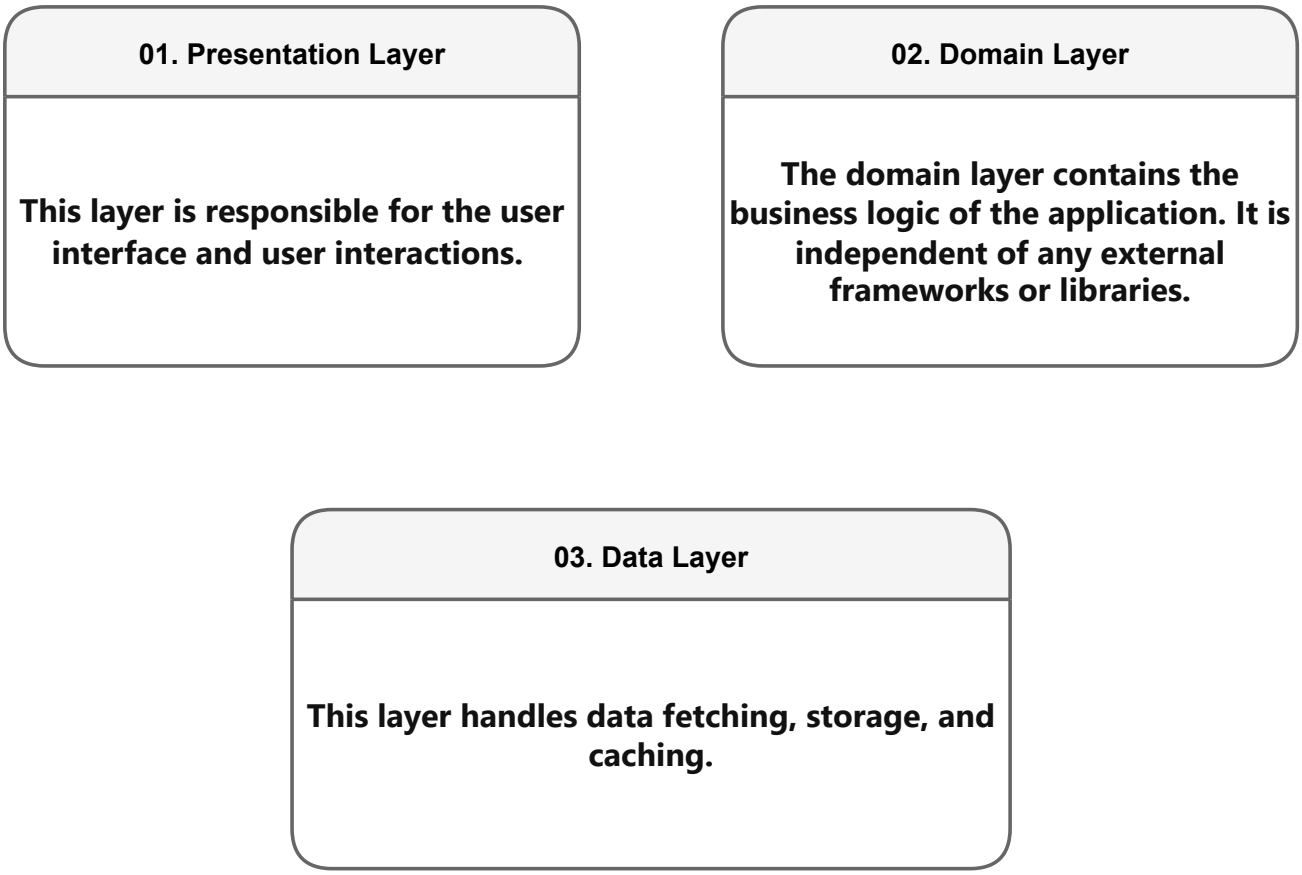
To build your application in Clean Architecuter you must divide your app in to a number of feature, each of them contains 3 layers



Domain Layer

The domain layer contains the business logic of the application. It is independent of any external frameworks or libraries. This layer includes:

01. Entities

Core objects of the application, representing business models.

What is the shape of the data needed by the app to work properly.

02. Use Cases

Specific actions or operations that can be performed, encapsulating the business rules.

03. Repositories (Interfaces)

Abstract definitions of data operations, which are implemented in the data layer

Data Layer

This layer handles data fetching, storage, and caching. It includes:

01. Data Sources

Responsible for accessing data from various sources like APIs, databases, or local storage.

02. Models

Data transfer objects (DTOs) that map to the data structures used by the data sources.

03. Repositories (Implementations)

Concrete implementations of the repository interfaces defined in the domain layer.

Presentation Layer

This layer is responsible for the user interface and user interactions. It includes:

01. Widgets

The UI components that users interact with.

02. State Management

Manages the state of the UI, often using tools like Provider, Bloc, or GetX.

03. Input Validation

Ensures that user inputs are valid before passing them to the domain layer.

Example Flow

1. **User Action**: A user interacts with the UI (e.g., clicks a button).
2. **State Management**: The state management solution (e.g., Bloc) handles the event and calls a use case.
3. **Use Case**: The use case performs the business logic and interacts with the repository.
4. **Repository**: The repository fetches data from a data source (e.g., a remote API).
5. **Data Source**: The data source retrieves the data and returns it to the repository.
6. **Repository**: The repository returns the data to the use case.
7. **Use Case**: The use case processes the data and returns the result to the state management solution.
8. **State Management**: The state management solution updates the UI with the new state.