

Fig: Movie app clean architecture.

**Folder Structure:**

* /presentation:

1. /bloc: using bloc for flutter state managament.
2. /journey : Screen files for homepage screen, favorite screen, about screen, feedback screen, search screen, movie details screen, login screen.
3. /themes: Themes used throughout application across different screens
4. /widgets: reusable widgets

* /data:

1. /core: Implements module for fetching and parsing remote data
2. /data\_source: Implements app’s use cases.
3. /repositories: Responsible for making data available to the domain layer.
4. /models: To map received response(json) into handalable output. Model classes are used by /data\_source to give app oriented return values.
5. /tables: Contains database mapping classes (not implemented)

* /doamain:

1. /entities: Entity classes are used to present data to ui. They are extended by /models classes.
2. /repositories: Contains abstract classes which only tell what data has to be fetched and carries parameters from /usecases classes.
3. /usecases: These classes implements core functionality of app like GetTrending GetPopular GetComingSoon SearchMovie etc. They simple carries info from UI to /data and returns /entities type details to blocs. They are used in blocs only.

* /commons:

1. /constants: Constant values used throughout app.
2. /extensions: Built-in class extensions for ondemand handling of built-in types.
3. /screen\_utils: Classes in this folder is responsible for providing dynamic viewport values so that app widgets adapts to different screen sizes.

* /di:

1. get\_it.dart : init() function that registers all usable componenets in app in GetIt class. (supplied by get\_it package). This way a central hub that can inject ondemand dependency of instances which will be signleton.