Movie App Architecture & Modules

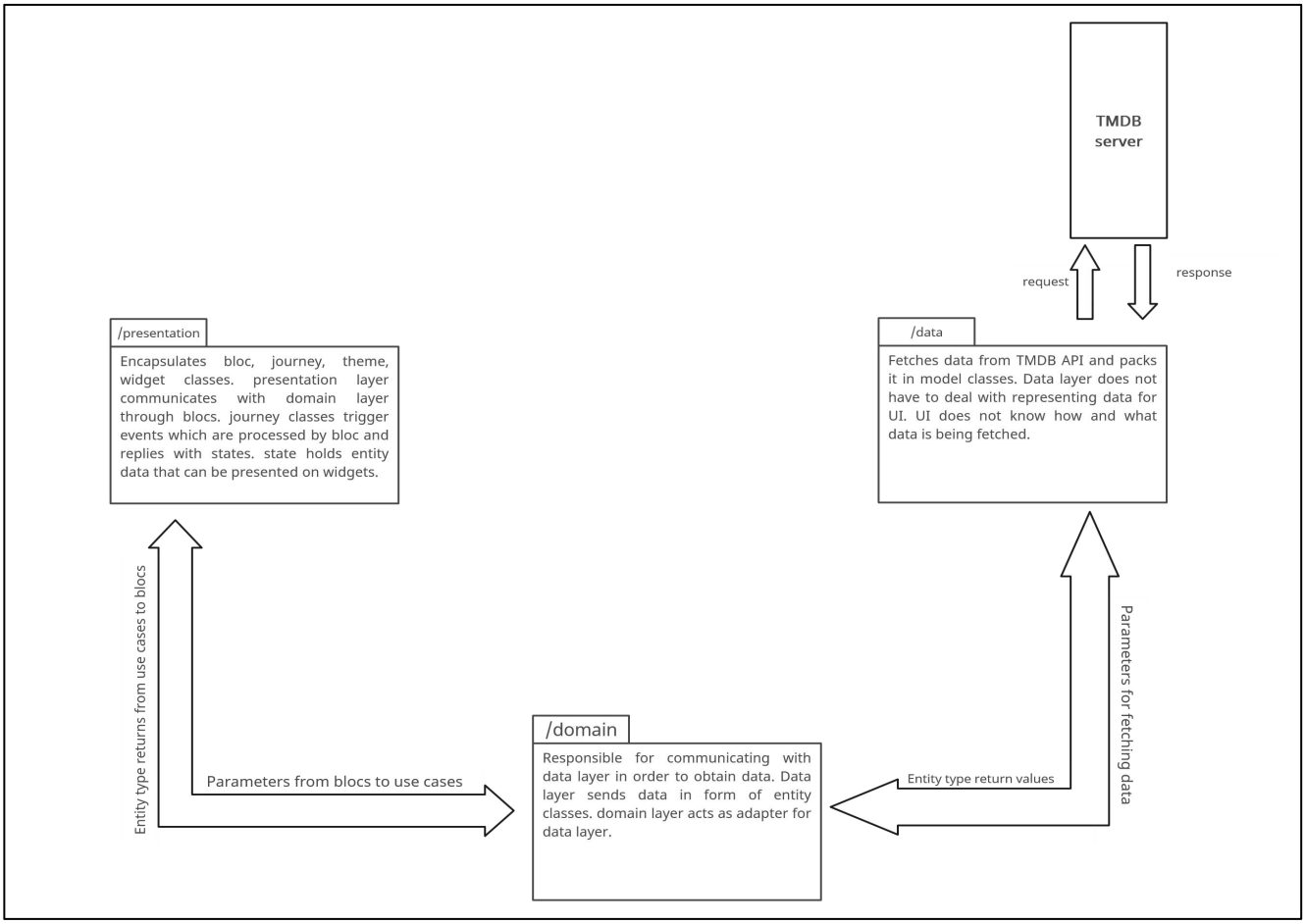

Fig: Movie app clean architecture.

Folder Structure:

- /presentation:
  - I. /bloc: using bloc for flutter state managament.
  - II. /journey : Screen files for homepage screen, favorite screen, about screen, feedback screen, search screen, movie details screen, login screen.
  - III. /themes: Themes used throughout application across different screens
  - IV. /widgets: reusable widgets

Movie App Architecture & Modules

- /data:
  I. /core: Implements module for fetching and parsing remote data
  II. /data_source: Implements app's use cases.
  III. /repositories: Responsible for making data available to the domain layer.
  IV. /models: To map received response(json) into handalable output. Model classes are used by /data_source to give app oriented return values.
  V. /tables: Contains database mapping classes (not implemented)

- /doamain:
  I. /entities: Entity classes are used to present data to ui. They are extended by /models classes.
  II. /repositories: Contains abstract classes which only tell what data has to be fetched and carries parameters from /usecases classes.
  III. /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:
  I. /constants: Constant values used throughout app.
  II. /extensions: Built-in class extensions for ondemand handling of built-in types.
  III. /screen_utils: Classes in this folder is responsible for providing dynamic viewport values so that app widgets adapts to different screen sizes.

- /di:
  I. 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.