LAYERED ARCHITECT.

Q Explain and illustrate the concept of layered architecture for s/w design. Also, detail the benefits.

A. When we are developing software, our project will contain many classes. These classes will look after elements of our project such as the GUI (how the data or application is presented), the logic of the application (any calculations, data manipulation, data handling etc), and also communication between remote sub systems (data bases, servers etc).

- Rather than having all these classes lumped into one location, we can reduce the complexity by seperating the sub-systems into layers, hence reducing coupling.

In our case study - Nicoles Petrol App - we used a three tiered architecture seperating classes into the view layer, domain / logic layer and services layer.

- View layer: main activity, Fillup activity, change Vehicle Act., settings activity
- Logic layer: Petrol Tracker (singleton), Fillup, vehicle, settings
- Service layer: Geo location, local storage, Network Connections

Each layer does not know about all or any of the other layers, only the layer in which it has a controller link to. For example

The view layer can only reference the logic layer as there is a singleton controller waiting for any events caused by the view layer.

The logic layer does not know of the view layer. This reduces coupling when coding and helps to reduce the amounts of changes to classes if a class in the view layer needs changing.

Also, as the logic layer is independent of the view layer it can be applied to any interface with different view layers.