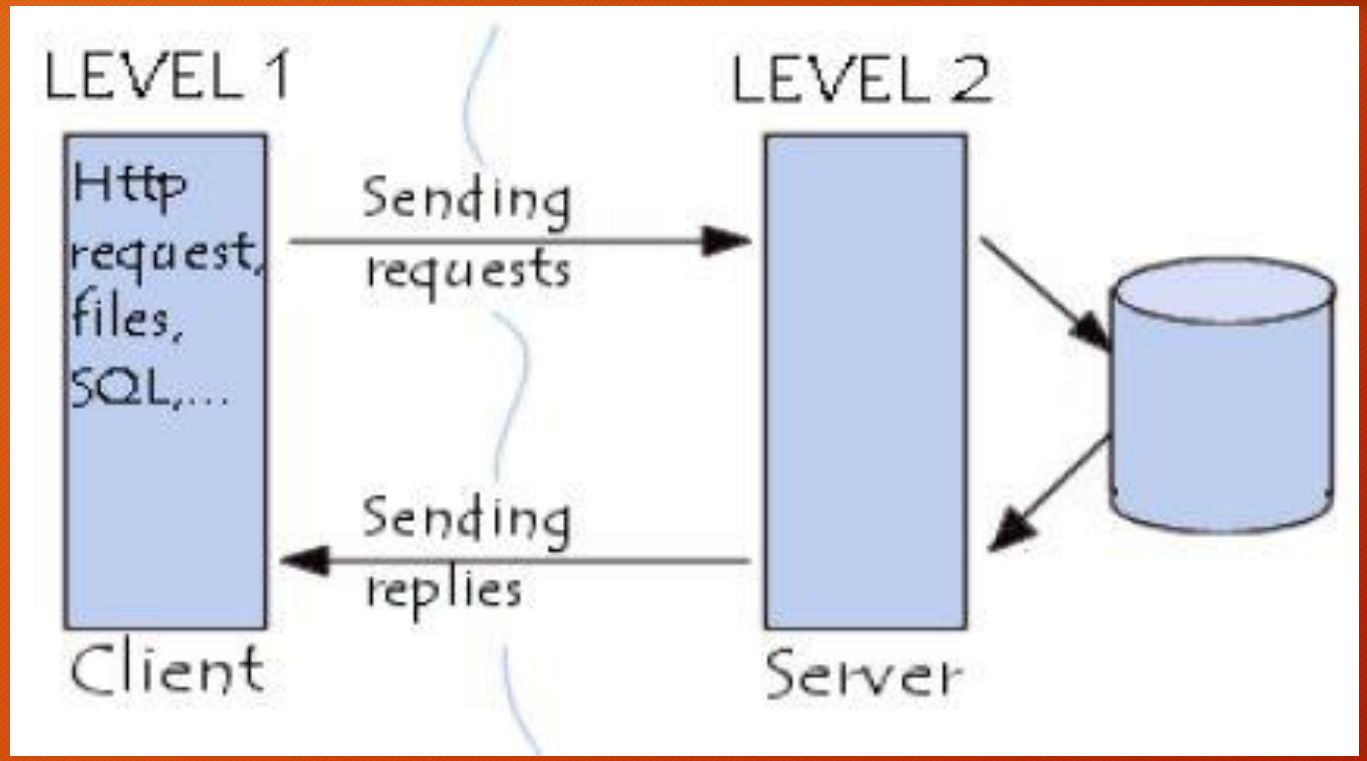
2-Tier, 3-Tier and n-Tier Architecture

Presented by : Er. Sudan Prajapati

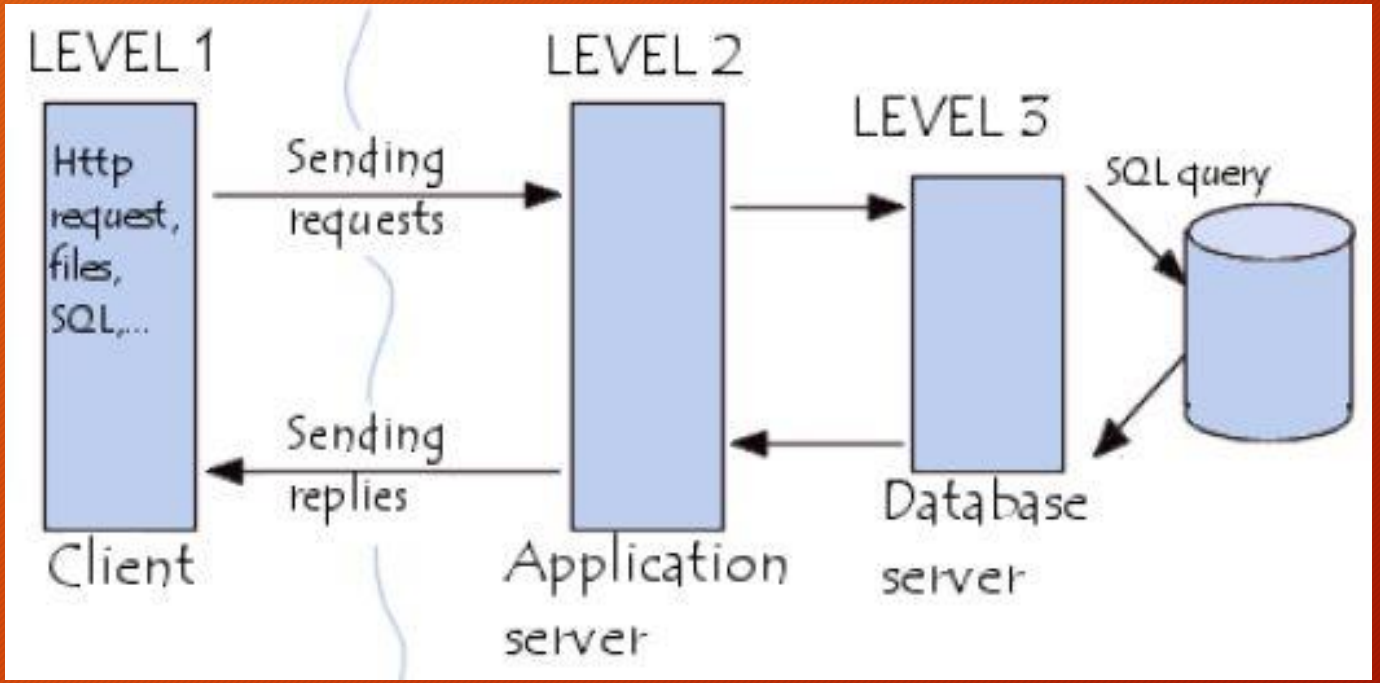
2-Tier Architecture

- 2-tier architecture is defined as client/server systems where the client requests resources and the server provide services to the client as requested, using its own resources.
- This means that the server does not rely on another application in order to provide part of the service.
- It runs the client processes separately from the server processes, usually on a different computer
- A two-tier architecture is a software architecture in which a presentation layer or interface runs on a client, and a data layer or data structure gets stored on a server.
- Separating these two components into different locations represents a two-tier architecture.
- The business logic that validates data, monitors security and permissions, and performs other business rules can be housed on either the client or the server, or split between the two.


LEVEL 1
LEVEL 2
Http request, files, SQL,...
Sending requests
Sending replies
Client
Server

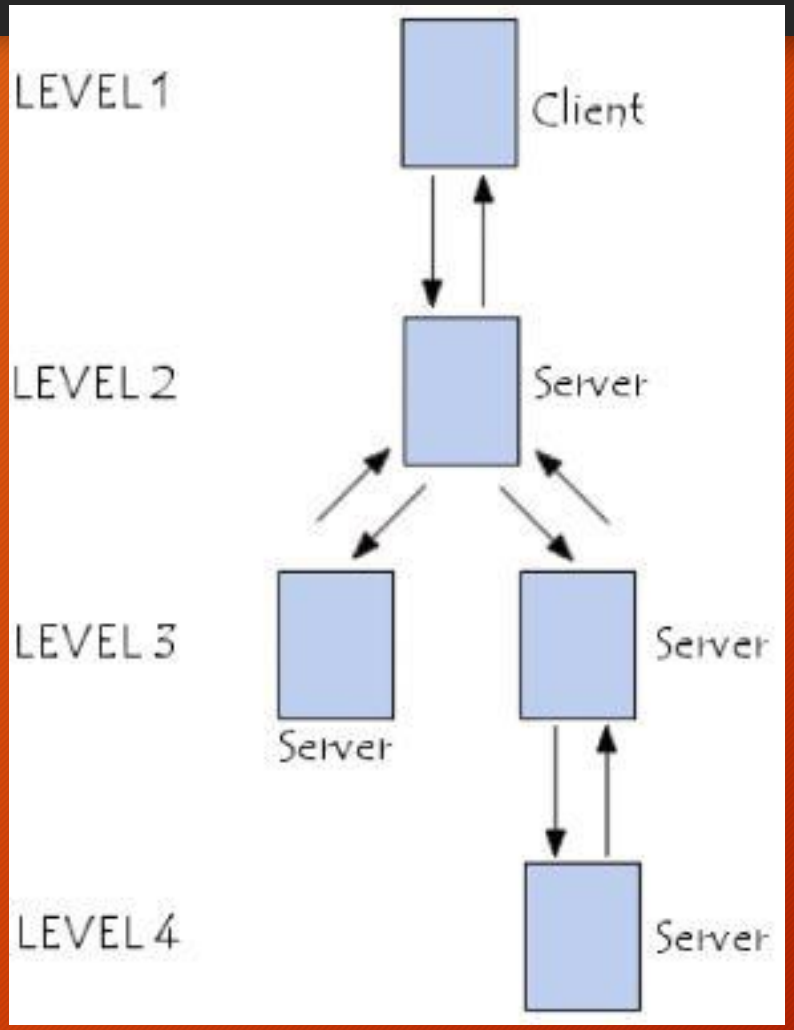
3-Tier Architecture

- In 3-tier architecture, there is an intermediary level, meaning the architecture is generally split up between:
  - A client, i.e. the computer, which requests the resources, equipped with a user interface (usually a web browser) for presentation purposes.
  - The application server (also called middleware), whose task is to provide the requested resources, but by calling on another server.
  - The data server, which provides the application server with the data it requires.


LEVEL 1
LEVEL 2
LEVEL 3
Http request, files, SQL,...
Sending requests
SQL query
Sending replies
Client
Application server
Database server

N-Tier Architecture (multi-tier)

- N-tier architecture is 3 tier architectures in which the middle tier is split up into new tiers.
- The application tier is broken down into separate parts.
- These parts differ from system to system.
- The primary advantage of N-tier architectures is that they make load balancing possible.
- Since the application logic is distributed between several servers, processing can then be more evenly distributed among those servers.
- N-tiered architectures are also more easily scalable, since only servers experiencing high demand, such as the application server, need be upgraded.
- The primary disadvantage of N-tier architectures is that it is also more difficult to program and test an N-tier architecture due to its increased complexity.


LEVEL 1
Client
LEVEL 2
Server
LEVEL 3
Server
Server
LEVEL 4
Server

Presentation tier

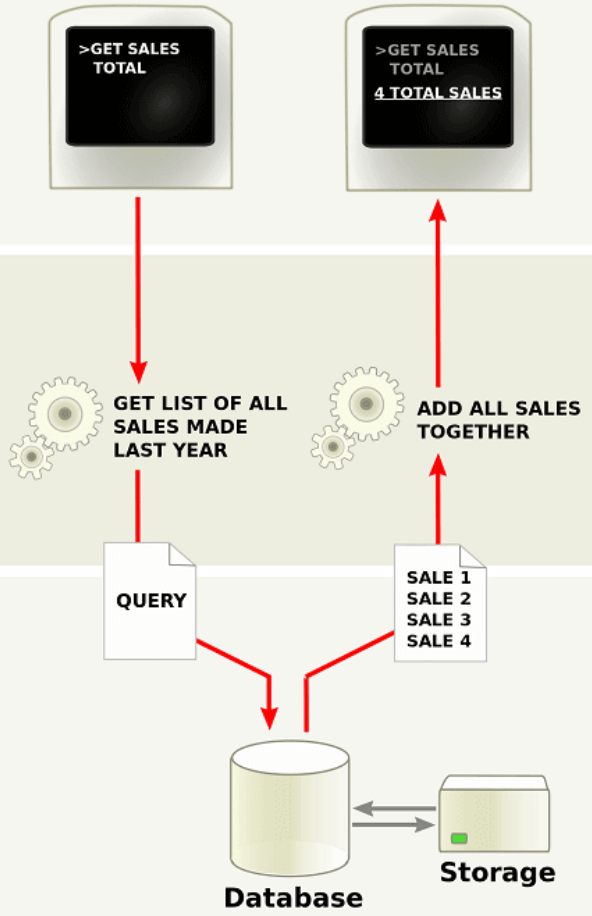
The top-most level of the application is the user interface. The main function of the interface is to translate tasks and results to something the user can understand.

Logic tier

This layer coordinates the application, processes commands, makes logical decisions and evaluations, and performs calculations. It also moves and processes data between the two surrounding layers.

Data tier

Here information is stored and retrieved from a database or file system. The information is then passed back to the logic tier for processing, and then eventually back to the user.



What are the Benefits of N-Tier Architecture

- Secure: You can secure each of the three tiers separately using different methods.
- Easy to manage: You can manage each tier separately, adding or modifying each tier without affecting the other tiers.
- Scalable: If you need to add more resources, you can do it per tier, without affecting the other tiers.
- Flexible: Apart from isolated scalability, you can also expand each tier in any manner that your requirements dictate.
- More efficient development. N-tier architecture is very friendly for development, as different teams may work on each tier. This way, you can be sure the design and presentation professionals work on the presentation tier and the database experts work on the data tier.
- Easy to add new features. If you want to introduce a new feature, you can add it to the appropriate tier without affecting the other tiers.
- Easy to reuse. Because the application is divided into independent tiers, you can easily reuse each tier for other software projects. For instance, if you want to use the same program, but for a different data set, you can just replicate the logic and presentation tiers and then create a new data tier.

How It Works

- When it comes to n-tier architecture, a three-tier architecture is fairly common. In this setup, you have the presentation or GUI tier, the data layer, and the application logic tier.
- **The application logic tier.**
  - The application logic tier is where all the “thinking” happens, and it knows what is allowed by your application and what is possible, and it makes other decisions.
  - This logic tier is also the one that writes and reads data into the data tier.
- **The data tier.**
  - The data tier is where all the data used in your application are stored.
  - Store data on this tier, do transaction, and even search through volumes and volumes of data in a matter of seconds.

- **The presentation tier.**
  - The presentation tier is the user interface.
  - This is what the software user sees and interacts with.
  - This is where they enter the needed information. This tier also acts as a go-between for the data tier and the user, passing on the user's different actions to the logic tier.