Java Enterprise Edition: J2EE

To develop Enterprise(Large scale business organization) level application:

--to develop a large scale Enterprise level application, in addition to the main business logic, we need some other services also , to make our business logic perfect.

```
public void transferAmount(int srcAccount, int destAccount, int amount){

//security
//transaction management
//logging


//check the validation of accounts
//check the amount inside source account
//deduct amount from srcAccount
//deposit amount to the desAccount

}
```

Sun microsystem : J2EE specification
Microsoft : .Net Specification

the implementaton of J2EE specification is : ApplicationServer s/w
the implementation of .Net specification is IIS server

Layared Architecture in Java Based Business application:-
===

1.maintaining the business data in secure and easily retrival manner.

--the logic that we write to implement this part of business application is known as Data Accees Logic.

2.processing the data according to the business rule .

--the logic that we write to implement this part of business application is known as Business/Service logic.

3.presenting the data to the user in user-understandable format.

--the logic that we write to implement this part of business application is known as Presentation logic.

--the above 3 logics is required for almost every business application.

```
class MyBusinessApplication{

//Data Access Logic  // fetch the account information from the DB

//Business Logic //calculating the Interest to the balance

//Presentation Logic // display the information to the client

}
```

Note:- we can write all the 3 logics to develop a business application in one program/class itself , if we do so, the following problem we will face:-

1.all the logics to develop the application will be mixed up with each other (no clear code seperation).

2.modification done in one logic may affect the other logic .

3. logics will depend upon each other so, parellal developement will not be possible.

4.testing each logic is also will become complex..

--to solve this problem, a java based business application ,we devide into 3 logical partition .

and each part we say as a layer:-

1.Presentation Layer(UI layer)

2.Business Logic Layer (Service layer)

3.Data Access Layer

--a bussiness application will be devided into the logical partition depending upon the role played by each part.

--logical partition of a business application is known as layer.

Presentation Layer :-

--it is set of java classes, which are responsible for generating user input screen and response page(output screen) is knwon as PL.

--this layer provides the intraction with the end-user.

Business Logic Layer/Service Layer:-
--

--programatical implementation of business rule of a business organization is nothing but business logic .
--a collection of java classes whose methods have business logic to process the data according to the business rule is known as SL/BLL.

Data Access Layer :-

--a set of java classes whose methods r exclusivly meant for performing CRUD operation with the DB server is known as DAL.

using JDBC and DAO pattern
using ORM approach (JPA with Hibernate)

****Note:- to communicate among these layers loose coupling should be promoted.**

WebServer: it is the small part or subset of Application Server.

diff bt WS and AS:
----

1. AS will have 2 container s/w i.e EJB container and Servlet container, where as Webserver will have only one container, i.e Servlet container.

2. We can connect with AS using any kind of protocol, but we can connect with the webserver with only http protocol.

Note: after invension of the Spring framework, to develop An enterprise application, we don't need the EJB container any more, because Spring IOC container will push or inject the required extra services(dependencies) to our main business logic, without any component contract.

--so using Spring f/w, we can manage our SL/ BLL using IOC container.
and we can develop our Enterprise application without the help of ApplicationSercer s/w also, i.e by the help of Webserver only

WebApplication:
==============

--the application that can be accessed over the web through the http protocol
by using web-browser s/w is known as Webapplication.

---webapp :-- server

--Webapplication resides/deployed on the Webserver/ApplicationServer.this type of application
executes on the webserver only.

--A Java based WA basically contains web-components or webresources like html file, image file, servlet program, jsp program.

--each webresource program is capable of generating one webpage.

webpage: that we can open through a web browser s/w.

the webpages classified into 2 categories:

1.static webpages : here content will be fixed for each user intraction, they are precreated (before client request) ex: home page of gmail.

--the webresources which can generate static webpages are called static webresource program ex: html file
image file, etc.

2.dynamic webpages: page contents will changed dynamically based on the input given by the user. ex: inbox of Gmail.

--they are not precreated before client request, they will be created based on the input value given by the client dynamically.

--the webresources which can generate dynamic webpages are called dynamic webresource program ex: Servlet program(it is a java class which implements Servlet interface), jsp file.

Webapplication = static(html file, image file, etc) + dynamic webresource program(Servlet, Jsp programs) + some configuration file

Servlet : it is a java class in which we write html code inside (it will generate dynamic html)

```
class MySrv implements Servlet{   //methods of this Servlet interface called by the
ServletContainer

//get the request from the browser/client
//connect with the DB
//int balance = get the balance from the DB

"<h1>Welcome to MyWebsite</h1>";
"<h1>Your Balance is : "+balance+"</h1>";
}
```

// from the server, Servlet is only responsible to get the request from the client and generate the response to the client

JSP: It is a html file only in which we can write java code.

MVC approach:

M : Java Bean classes : these classes contains the application related data, which will be processed inside the SL
and will be persisted inside the DAL.
V : JSP
C : Servlet

Note: A WA is a combination of static webresource + dynamic webresource program.

html file
js code (client side programming language, it will execute on the client machine)
css files
image files
mp3
mp4
servlet program
jsp files
configuration file
***.jar**

--then we need to arrange all the files in a standard folder structure given by sun-microsystem.and then compress that folder in a deployable unit.

---.war file (web-module) -----> deploy on the webserver/application server (Web-container)

Similarities/differences bt Webapplication and Webservice:
===

---WS is also a type of WA, that can generate dynamic result as raw data(json, xml, text) instead of generating a html response (webpage)

--WA generates the view(webpage) whereas WS generates raw data.

--this raw data generated by the WS can be reused inside any other WA or any other type of application developed in any technology.

```
Indigo -----> html   (Webapplication) : it is for enduser
          -----> raw data (Webservices)  :- it is for any other application
Yatra
 :----------Webapplication
```

--in traditional web app, all the activity will be taken care by the Servlet.
-- with the traditional web app, if multiple request comes to a servlet, it will be the burden to the servlet to maintain all the logics.

--to simplify the role of servlet, sun-micosystem has introduced another technology called JSP.

--initialy JSP invented to replace the Servlet, but very soon developers realised that instead of replacing the Servlet with JSP, JSP should complement the Servlet. and Sun-microsystem has invented a pattern called MVC patten to simplify the Java based Webapplication.

--In traditional mvc based webapplication developers need to write the Servlet class which will have all the flow control logic and configure that servlet inside our application.

--whereas in spring-mvc framework developers need not develop any Servlet class, instead Spring-mvc f/w has given a Servlet which will have all the flow control logic. (DispatcherServlet)

SpringBoot and Spring MVC:
========================

--in Spring-MVC, we get the help of DispatcherServlet class, with all the flow control logic, we just need to configure that DispatcherServlet inside our application. and then we need to deploy our application to the Webserver s/w by installing that server s/w.

but with the help of SpringBoot, even that part also we don't need to do that,

because SpringBoot follows convention over the configuration, In SpringBoot, that DS is already configured and also, we don't need to install any webserver explicitly, because SpringBoot will have integrated Tomcat server(Webserver) is there.

Spring f/w : Spring-core (IOC container)

Spring mvc:it helps to develop a webapplication by providing a Predefined Servlet called DispatcherServlet class

Spring Boot: it provide RAD(Rapid application development, by auto configuring the DS and providing interated Tomcat web server)

Client <--------- Http protocol -------------->Server(Web-Container (WA or WS))

https://www.gmail.com:--------- here we are generating a http request to the server

https://ipaddressofGmailserver:portnumberoftheServer/appname/index.html

--here /index.html is uri(uniform resource identifier) (home page of the gmail app)