**Web ?**

1. Techgatha => 3-tier architecture
2. Use HTTP protocol to communicate over the web
3. Browsers understand HTML/ CSS/ Js only => static websites [ banners on the streets ]
4. Javascript has limitations as it cannot communicate with the database directly **if** used for client side scripting.
5. To create dynamic websites [ showing dynamic data based on users request or who is using that website ]
6. Dynamic data => always involves database based on who is logged in / based on what user is searching
7. Browser[ Client] sends request to the server and if the request goes to a java application then need a java compliant server
8. There are few terminologies with respect to HTTP
   1. HTTP status codes : 1xx, 2xx, 3xx, 4xx, 5xx
   2. HTTP methods : GET, POST, PUT , DELETE , PATCH, HEAD, TRACE
   3. HTTP headers: Accept/ Content-Type/ Cookies

**Servlet**

1. Is a technology to communicate over HTTP protocol and process the request and send appropriate response
2. Uses java as a programming language
3. To create a servlet, the class should extend HttpServlet class
4. Servlets provide methods for respective HTTP methods doGet(), doPost()
5. doGet(), doPost() methods take 2 parameters:
   1. HttpServletRequest => responsible to process the request
   2. HttpServletResponse -> responsible to send content as a response that can be HTML / TEXT/ PDF/ JSON/ XML/ Images / Video/ Audio…
6. PrintWriter => to send HTML / TEXT content
7. GET request can be made via URL refresh, anchor links, form submissions or javascript AJAX
8. POST request can be made via form submissions or javascript AJAX
9. GET => used for fetching data. Less secured and has limitation on the amount of data that can be appended as part of the url
10. POST => used for form submission. Preferred as it secures data by not exposing over the URL. No limitations on the amount of data that can be sent
11. To process the HTTP request and delegate to respective Servlet, a server is needed
12. Tomcat / JBOSS/ Glassfish/ WebLogic etc server to process the HTTP request
13. Servlet is used to embed HTML in java code
14. To access the request parameters : request.getParameter(“<parameter-name>”)  
    parameter-name => the key passed either in the query string ex: search as bleow  
    <http://localhost:8080/ServletDemo/products?search=Laptop>  
    OR can be the form parameters with every form element having the **name** attribute
15. Database integration
    1. Add jar file under webapp/WEB-INF/lib folder
    2. Integrate database code as normal java jdbc code
16. sendRedirect :
    1. It is used to redirect the response to other resource inside or outside the server
    2. It makes the client[browser] create a new request to get to the resource and it is not execute on the server side
    3. It involves client to make the request to the redirected url
    4. URL changes to the redirected url
17. RequestDispatcher [RD ]:
    1. Inter-servlet communication
    2. This request is processed on the server side.
    3. The client is unaware about the various calls happening on the server side
    4. The end result accumulated or not from all the servlet/jsp calls is sent back to the browser
    5. URL does not changes
    6. Request can either be forwarded or included using RD  
       rd.forward(req, resp);  
       rd.include(req, resp);
    7. The request-response objects are the same instances across the request cycle
18. Session Management: Since HTTP is a stateless protocol, if the application needs to maintain the state of who is accessing the application and generate respective responses, need to implement session management techniques or making it stateful as follows:
    1. URL Rewriting
    2. Cookies
    3. Hidden form fields
    4. HttpSession
19. HttpSession => Provides with methods to set, retrieve, update or delete the attributes from the session
    1. To get the session object reference: HttpSession session = request.getSession(). Server provided with the references as and when needed and this is called as Dependency Injection [DI] and the dependencies are injected by the server. Hence server controls when to create what and this is called as Inversion Of Control
    2. Add data in the session : session.setAttribute()
    3. Remove data : session.removeAttribute()
    4. Invalidate the session : session.invalidate()
    5. Server adds a JSESSIONID as cookie on the client machine to be able to process a request-response cycle
20. Filters:
    1. Extend the class HttpFilter and implement Filter interface
    2. Filters are used to write common functionalities that needs to be processed for every request or response
    3. doFilter method: It is the location the BL resides whether to call the next filter or servlet in the chain.
    4. If chain.doFilter() is not invoked the request will not go further and it will just return the response back to the client
21. Init() methods in filter or servlets are used to initialize the servlets or filters with any specific initialization code. Ex: Preferences even before the request of the client is processed.