Web Application	Desktop application
Highly scalable	Need installation on every machine to increase scalability
Internet is needed	Internet is not required
Runs on web server	It runs on users machine
No location barrier	Can use only at local machine
Web application is collection pages	
J2EE, springMVC, Spring Boot	J2SE -Standard Edition

==Steps for creating war file manually==

1. Download tomcat on your machine (version 10.x)
   https://tomcat.apache.org/download-10.cgi - from core---- download 64 bit windows zip file
2. Somewhere on your machine create folder by name mytomcat
3. Unzip the file into mytomcat folder
4. Goto environment variable ,Set JAVA_HOME ----- to folder where java is installed   -------
    C:\Program Files\Java\jdk1.8.0_161
   Or
   ==C:\Program Files\Java\jdk-17==
5. Set path to %JAVA_HOME%\bin
6. For linux
      ==Set TOMCAT_HOME=" /home/user1/Java -II\apache-tomcat-9.0.37-windows-x64\apache-tomcat-9.0.37==
7. For windows
   Go to environment variables and create a new variable by name TOMCAT_HOME and assign vale to E:\apache-tomcat-10.1.33-windows-x64\apache-tomcat-10.1.33
8. To create war file
   a. Create a folder by name myapp inside webapps folder (webapps is inside TOMCAT_HOME folder)
   b. Inside myapp folder
      i. Create a index.html file

         ```
         <html>

         <head>

         </head>

         <body>

         <h1>Hello world!</h1>

         </body>

         </html>
         ```

      ii. Create a file formdata.html

         ```
         <html>
         ```

```html
<head>
</head>
<body>
<h1>This is second file</h1>
<form>

  <label for="nm">Name: </label>
  <input type="text"  name="nm" id="nm">
  <label for="em">email: </label>
  <input type="text"  name="em" id="em">



<label for="addr">adress: </label>
  <input type="text"  name="addr" id="addr">


</form>
</body>
</html>
```

iii. create a folder WEB-INF inside myapp folder----- inside WEB-INF folder add following
  i.   Create a folder lib
  ii. Create a folder classes
  iii. Create file web.xml copy following contents into the file

```xml
<?xml version="1.0" encoding="UTF-8"?>
<web-app
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://java.sun.com/xml/ns/javaee"
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd"
id="WebApp_ID" version="3.0">
 <display-name>MyHelloProject</display-name>
 <welcome-file-list>
  <welcome-file>index.html</welcome-file>
 </welcome-file-list>


</web-app>
```

c. Open E:\apache-tomcat-10.1.33\apache-tomcat-10.1.33 \conf\tomcat-users.xml
Uncomment following contents into the file

<mark><user username="admin" password="" roles="manager-gui"/></mark>

<!-- <user username="robot" password="<must-be-changed>"
roles="manager-script"/>

-->

<!-- Open command prompt and change the directory to

E:\apache-tomcat-10.1.33-windows-x64\apache-tomcat-10.1.33\bin
E:\apache-tomcat-10.1.33-windows-x64\apache-tomcat-10.1.33\bin>startup.bat
In linux use startup.sh

d. Open browser and check whether home page of tomcat is visible
http://localhost:8080

   i. Type url
      http://localhost:8080/myapp/
      This will display index.html
      Because it is added in welcome files list in web.xml file.
   ii. Type url
      http://localhost:8080/myapp/formdata.html
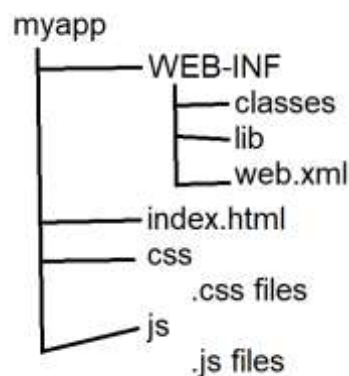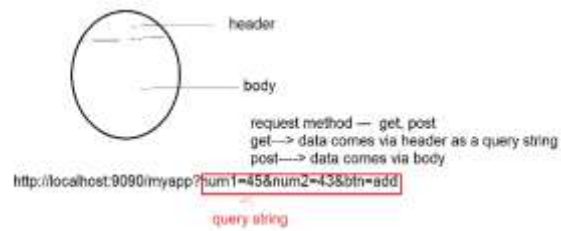   it will display output of formdata.html

e. You can click on manager app, enter username as admin, password is ""> then click
on myapp

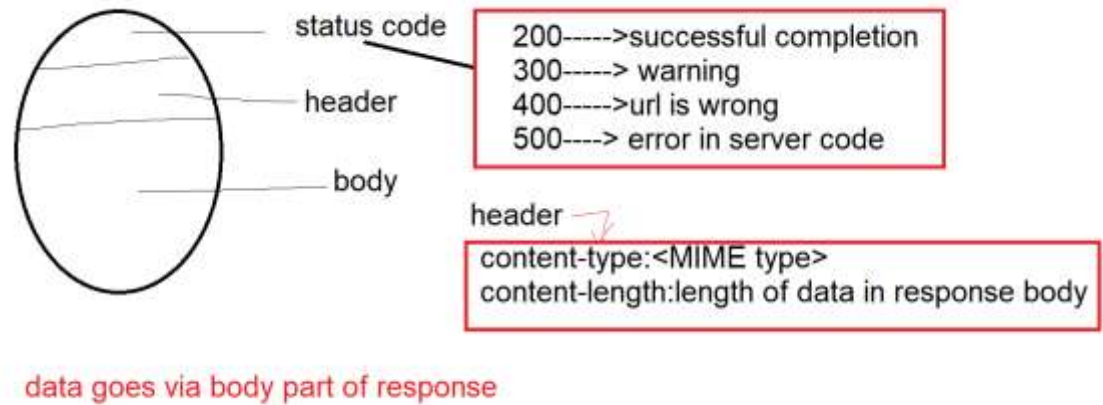<mark>Structure of war file</mark>

```
myapp
  └────── WEB-INF
  │           ├── classes
  │           ├── lib
  │           └── web.xml
  ├────── index.html
  ├────── css
  │            .css files
  └── js
        .js files
```

Request Object

header

body

request method — get, post
get —> data comes via header as a query string
post —-> data comes via body

http://localhost:9090/myapp?num1=45&num2=43&btn=add

query string

Response Object

status code

200----->successful completion
300-----> warning
400----->url is wrong
500----> error in server code

header

header

content-type:<MIME type>
content-length:length of data in response body

body

data goes via body part of response

CGI gateway(Common gateway Interface, In this to handle requests, multiple processing was used, hence server was able to handle less request

Multithreading
Every request will be handled by different thread, so it uses multithreading
Java uses multithreading