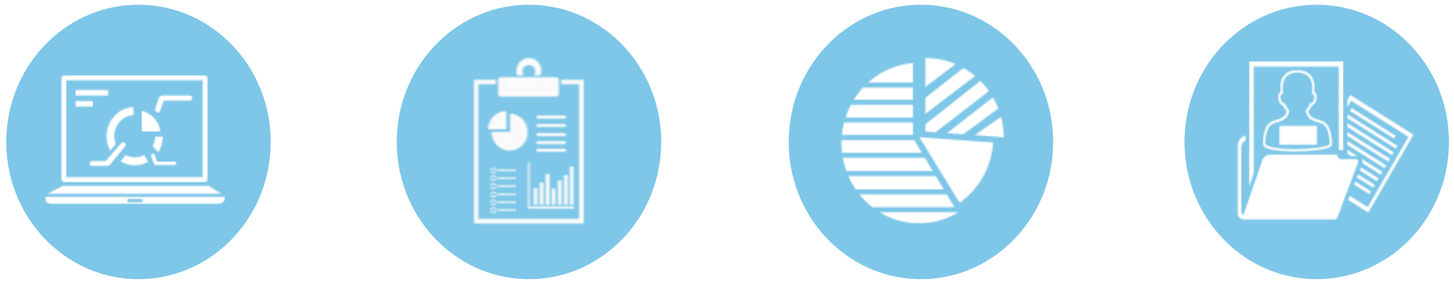
Advanced Java

Lesson 1—Introduction to Java EE



Learning Objectives

- Discuss Java editions
- Describe distributed applications
- Explain Client-Server Model and important terms related to it

Advanced Java

Topic 1—Java Editions

Java Editions

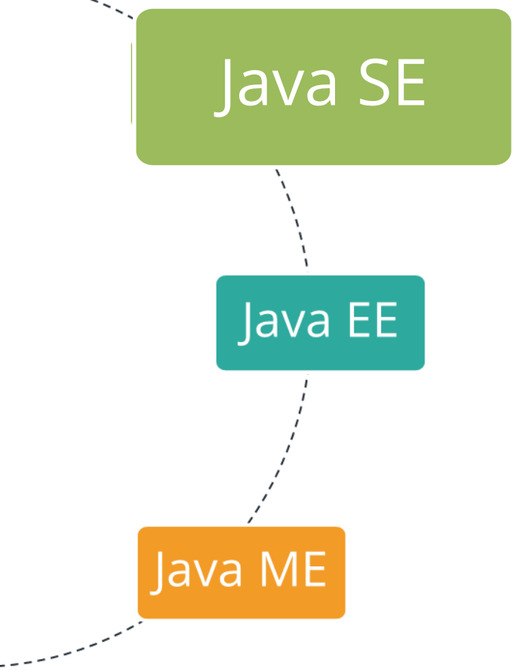
Java SE stands for Java's Standard Edition

Java EE stands for Enterprise Edition

Java ME stands for Micro Edition

Java Editions

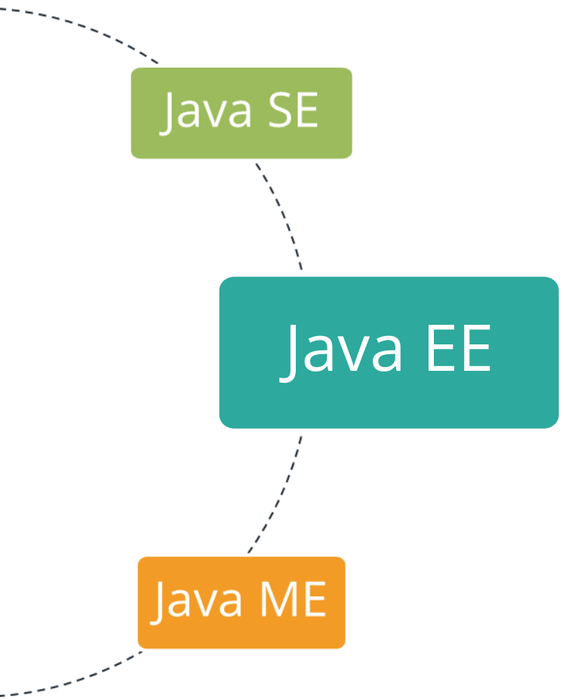
FEATURES OF JAVA SE (STANDARD EDITION)



- Java SE is the core Java programming platform. It contains all Java libraries.
- The latest version is **Java SE 8u131**
- It helps to develop and deploy Java applications on desktops, servers, and embedded environments. It is used to build standalone applications.

Java Editions

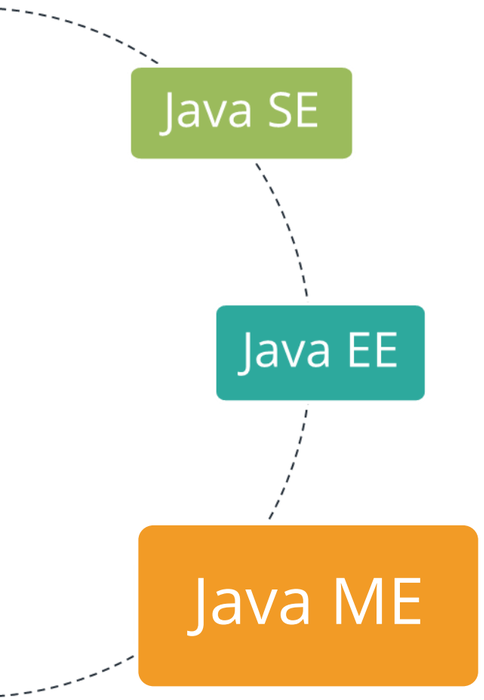
FEATURES OF JAVA EE (ENTERPRISE EDITION)



- Java EE platform is designed to help developers create large-scale, multi-tiered, scalable, reliable, and secure network applications.
- The latest version is **Java EE 7**
- It is portable, so it allows an application hosted at one common place to be accessed by users across the world over the Internet.
- It provides API (Application Programming Interface) to develop distributed applications that follows Client-Server Model.
- It facilitates the development of web applications.

Java Editions

FEATURES OF JAVA ME (MICRO EDITION)



- Java ME (Java 2 Platform, Micro Edition) is a technology that allows programmers to develop programs for mobile application devices.
- The latest version is **Java ME 8.3**
- It consists of programming specifications and a special virtual machine, the K virtual machine, that allows a Java ME-encoded program to run on mobile devices.

Java EE vs. Java SE

The Java EE (Enterprise Edition) differs from Java SE (Java Standard Edition Platform) in terms of the libraries it provides to deploy:

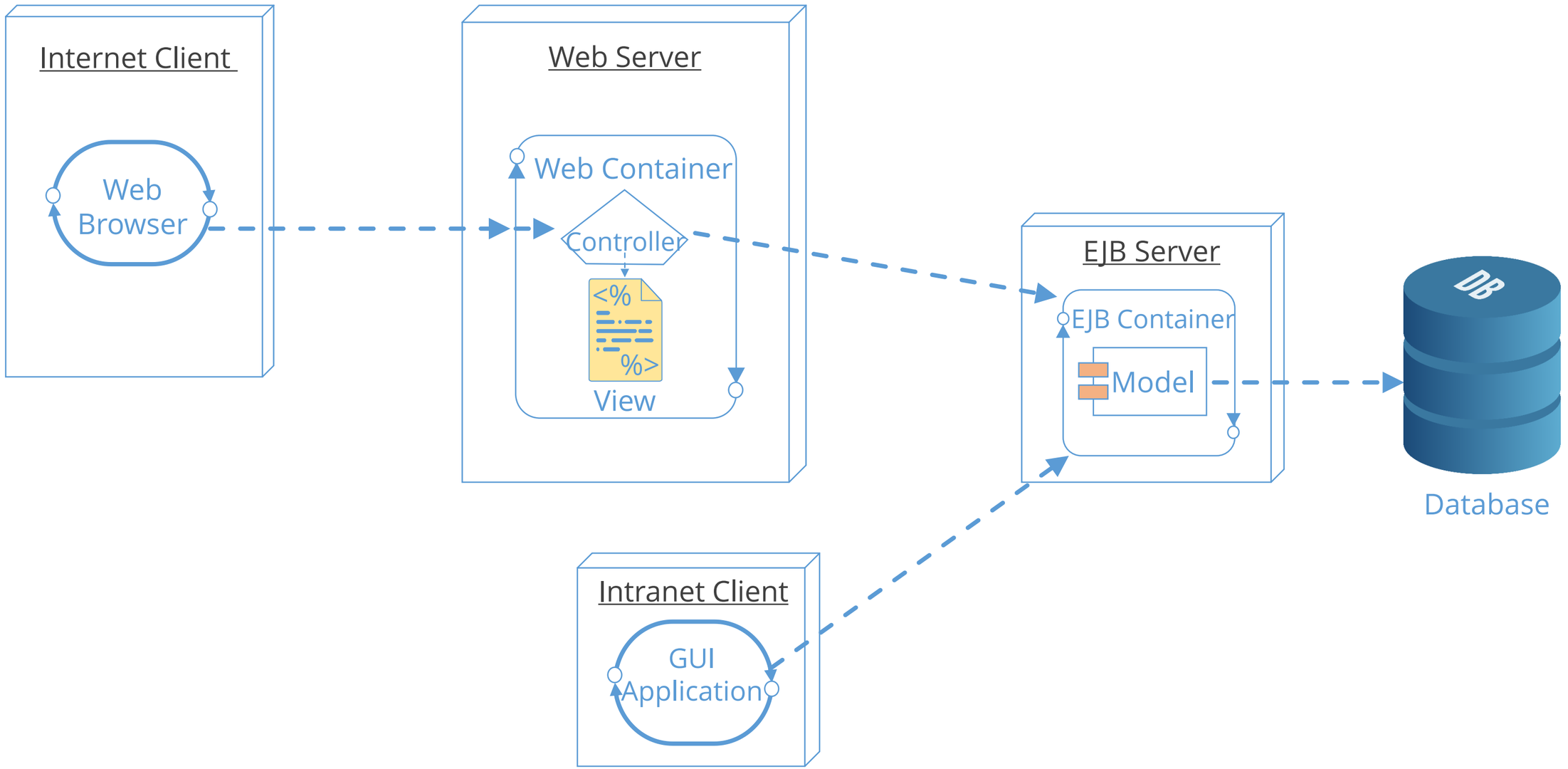
- fault-tolerant,
- distributed, and
- multi-tier Java software, based largely on modular components running on an application server.

Java EE Libraries and APIs

- Built on top of Java SE, Java EE provides additional libraries for
  - database access [JDBC (Java Database Connectivity), JPA (Java Persistence API)]
  - remote method invocation (RMI)
  - messaging [JMS (Java Message Service)]
  - web services
  - XML (eXtensible Markup Language) processing
- It also defines standard APIs for Enterprise JavaBeans, Servlets, Portlets, and Java Server Pages (JSP)

The installation guide containing the steps to download and configure Java JDK (standard edition) and Apache Tomcat (to access additional libraries) can be downloaded from your LMS.

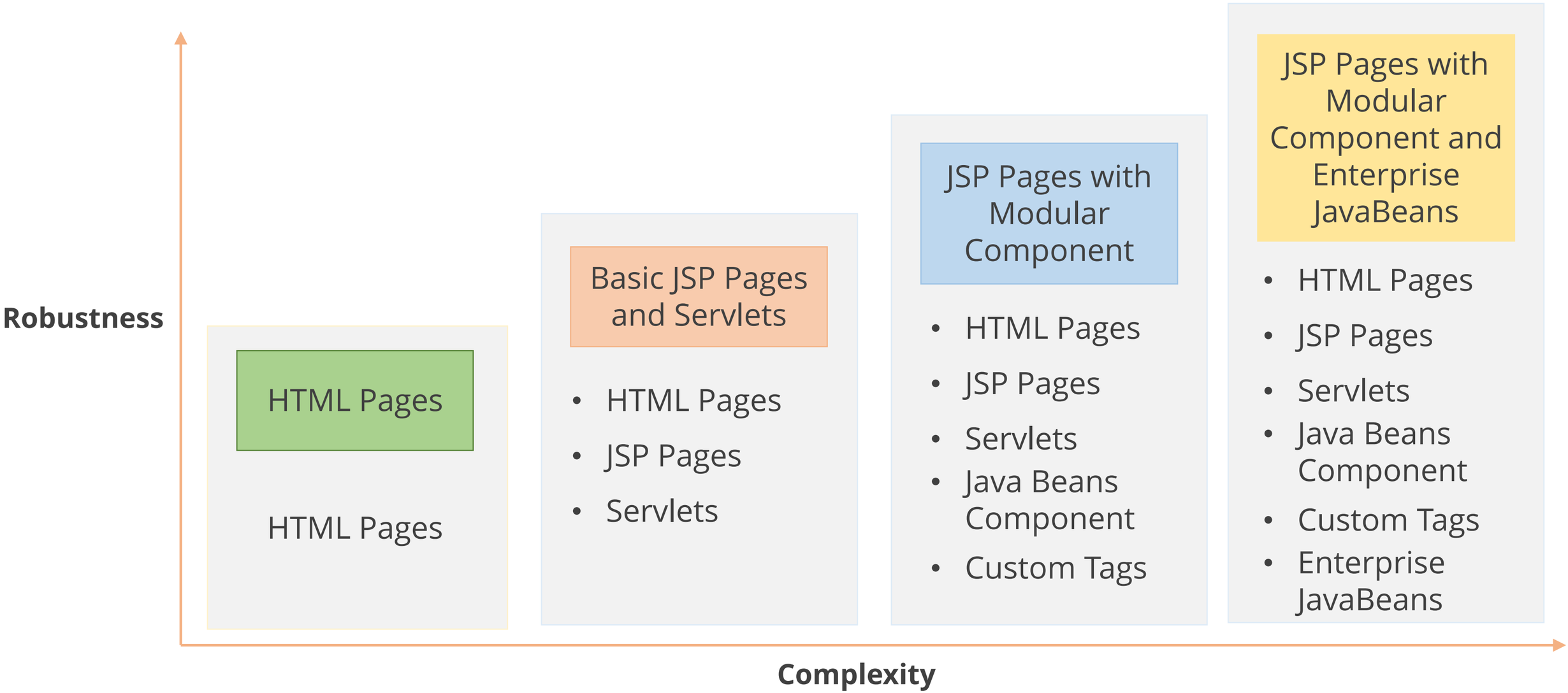
Java EE Architecture



Java EE Architecture: Features

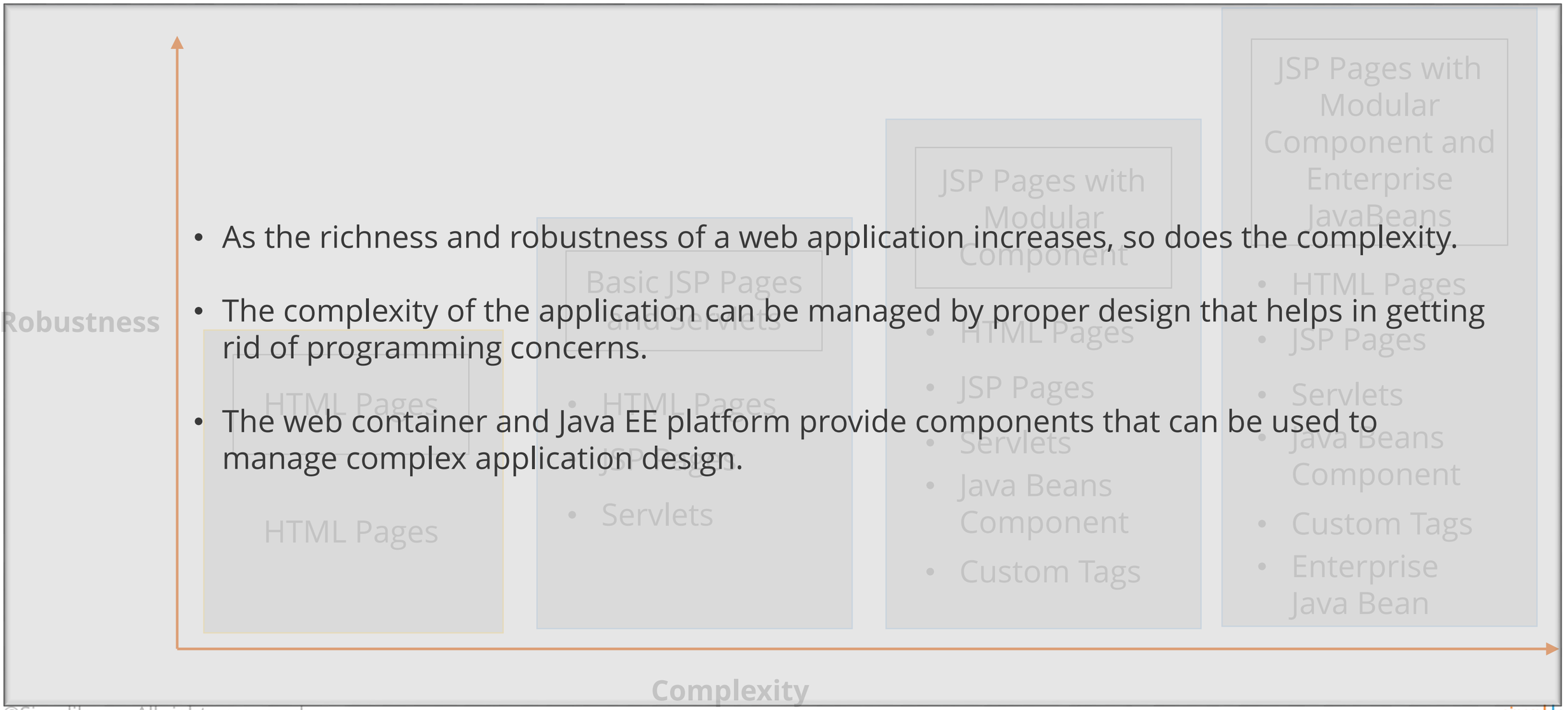
- Java EE platform facilitates an architecture in which the business components are placed in a separate tier.
- The Java EE architecture enhances features such as scalability, extensibility, and maintainability.
- Its modular design allows easy modification of the business logic. It provides more security because it partitions business services from the web tier. It also permits clean job separation of job roles.
- The enterprise components can leverage their EJB container for service, such as component and resource management, security, persistence, and transactions.

Relating Java EE Architecture Complexity and Robustness



Relating Java EE Architecture Complexity and Robustness

KEY POINTS



- As the richness and robustness of a web application increases, so does the complexity.
- The complexity of the application can be managed by proper design that helps in getting rid of programming concerns.
- The web container and Java EE platform provide components that can be used to manage complex application design.

Advanced Java

Topic 2—Distributed Applications and Client-Server Model

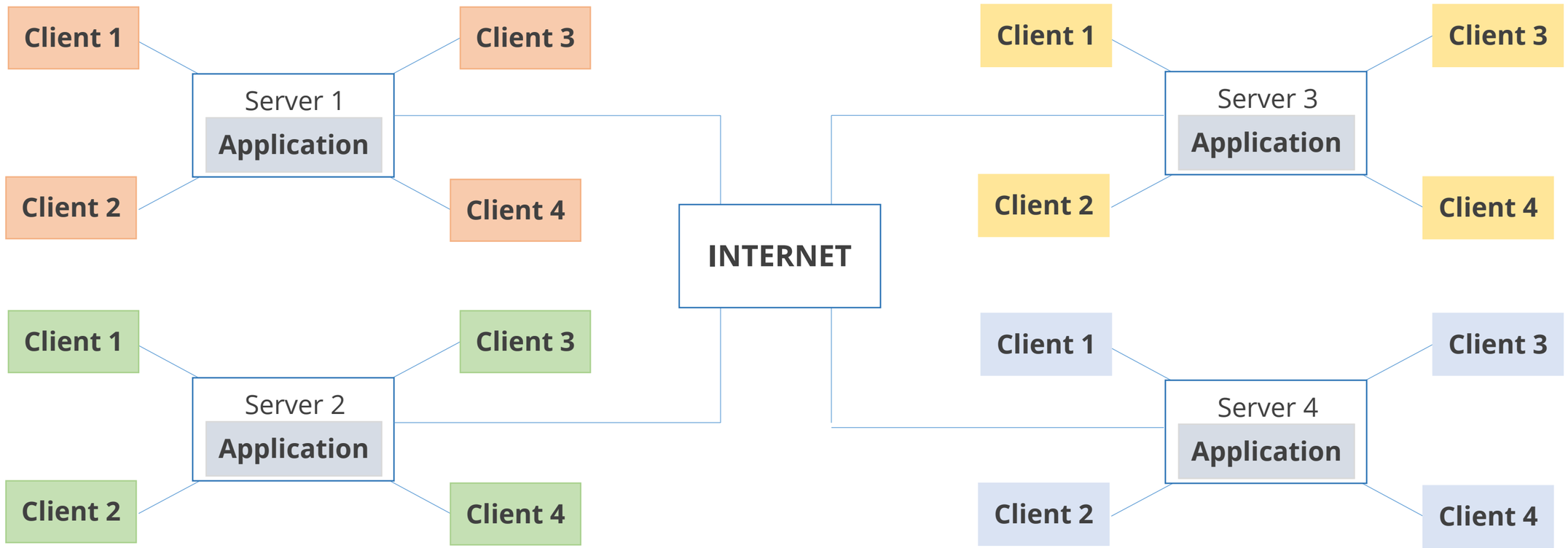
- Distributed Applications
- Client Server Model
- HTTP Protocol
- Types of Server Response
- Client Request Types
- HTML
- Running HTML code with Apache

Recall!

Java EE provides API (Application Programming Interface) to develop distributed applications that follow Client-Server Model.

Distributed Applications

- In distributed applications, same application can run across different machines at the same time using internet.
- One machine acts as Server where application is deployed and the other machines act as Client.



Client-Server Model

DEFINITION

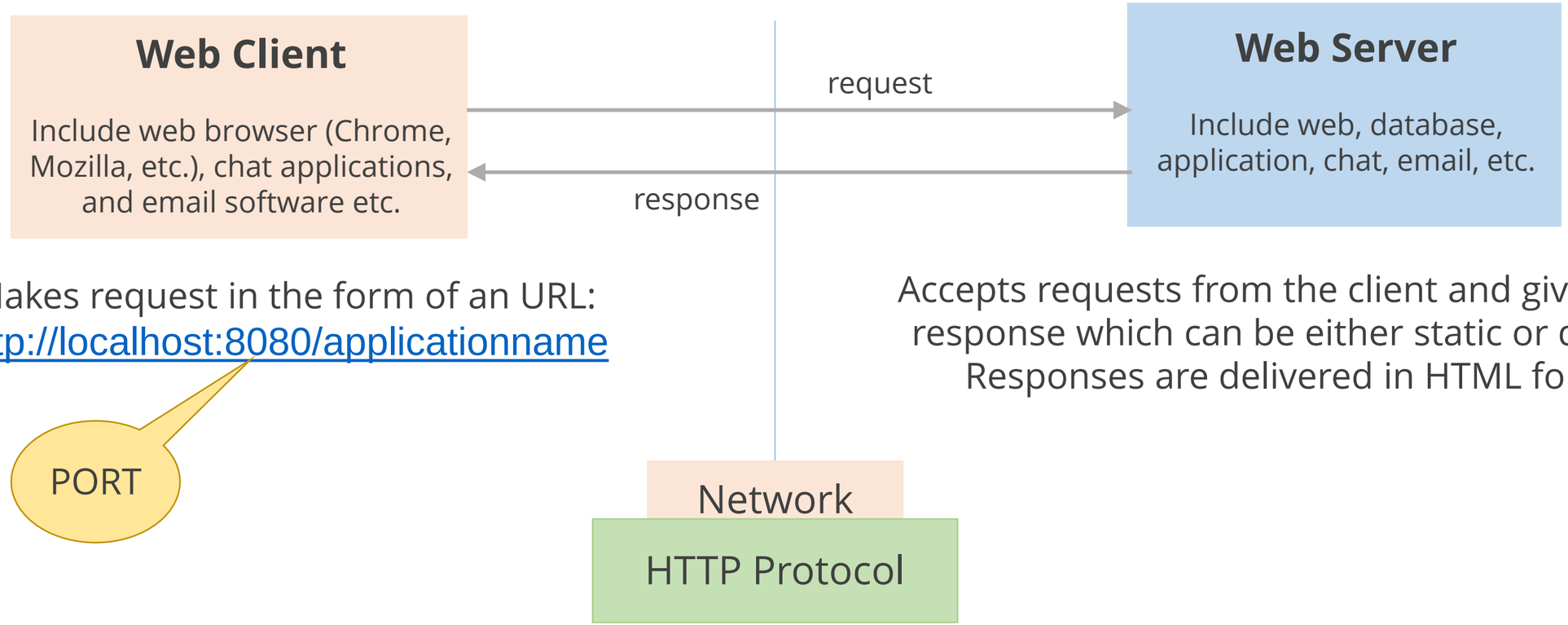
The client–server model is a distributed application structure that partitions tasks or workloads between the providers of a resource or service (web servers) and service requesters (web clients).

Both web client and web server are programs.

A web client is not a user, but a program. Someone who uses this program to send request to web server is known as “end user.”

Client-Server Model

ARCHITECTURE



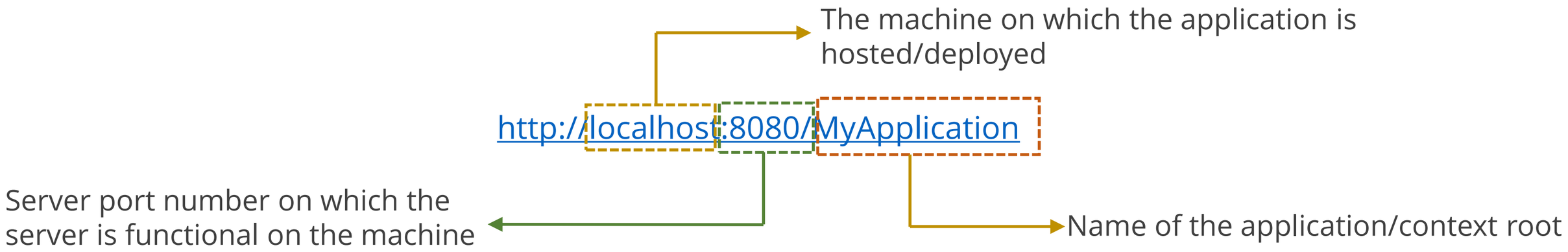
HTTP (Hypertext Transfer Protocol) is a TCP/IP based communication protocol, which is used to deliver data over the web.

It provides a standardized way for the client to communicate with the server.

Client-Server Model

SIGNIFICANCE OF URL AND PORT

Uniform Resource Locator (URL) uniquely identifies a resource on the internet and follows HTTP protocol.



Tomcat server runs on port 8080. Remember to check during installation.

Client-Server Model

FEATURES OF HTTP PROTOCOL

- HTTP (Hypertext Transfer Protocol) is a TCP/IP based communication protocol, which is used to deliver data over the web.
- This protocol is used to change or transfer the hypertext.
- Hypertext is a structure text that uses logical links between containing text.
- It is a stateless protocol—it does not require the HTTP server to retain information about each user.

Client-Server Model

SERVER RESPONSE

Server generates response for Client request. It sends three things:

1. Status which tells whether client's response has successfully reached the server or not. Example: 404 error for page not found.

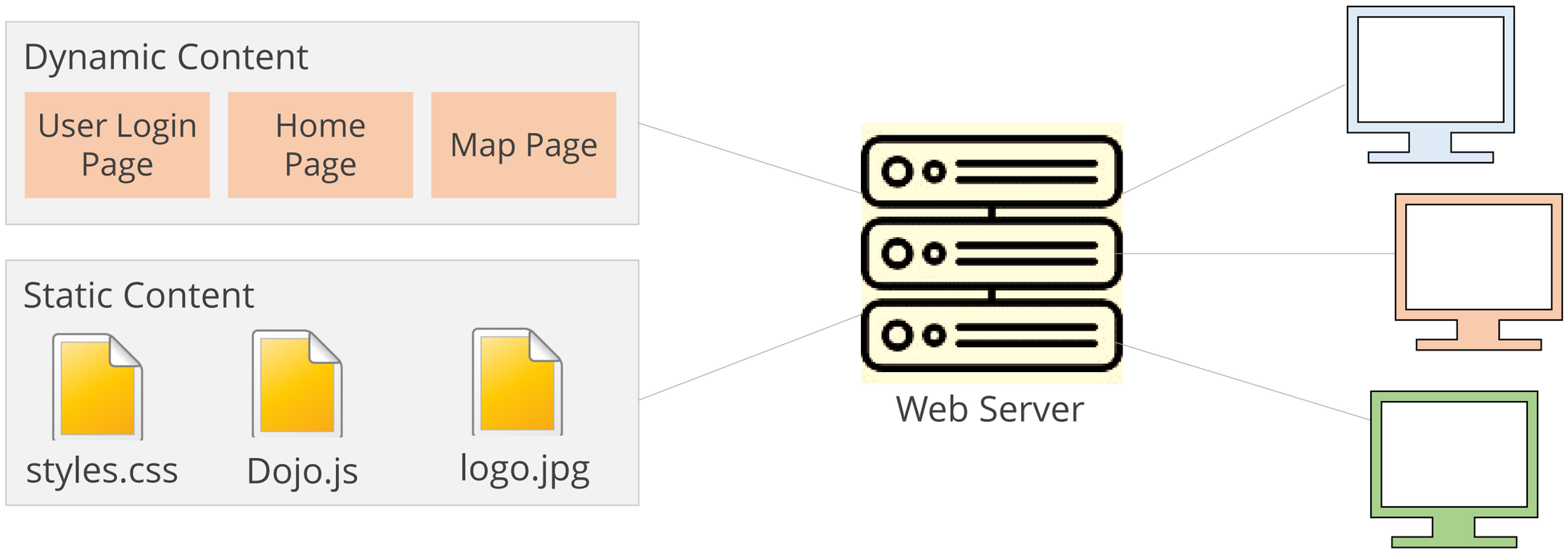
2. Information about the type of content that the server is going to send. The server sends HTML pages as response.

3. Actual content, example: when client types **http://www.google.com** in any browser, the server navigates the client to the Google home page.

Client-Server Model

SERVER RESPONSE TYPES

- Static Page: There is no change in the content. Client requests a web page and server gets the page back as response.
- Dynamic Page: The page content changes frequently. Server may not be able to process the request.



Client-Server Model

TYPES OF SERVERS

Commonly used server types are as follows:

- Web Server
- Application Server
- Proxy Server
- Virtual ServerBlade Server
- File Server
- Policy Server

We will focus on web and application server in this course.

Web Server Vs. Application Server

Web Server	Application Server
Web Server program runs on the server.	It is a component based product.
It looks for incoming requests and services those requests. Once the web server receives a request, depending on the type of request, it might look for a web page, or execute a program on the server.	It provides middleware services for security and state maintenance, along with data access and persistence.
Web servers have plugins to support scripting languages like Perl, PHP, ASP, JSP etc. through which they can generate dynamic HTTP content.	Application Server can do whatever Web Server is capable of doing as it has Web Server as an integral part. Additionally, it has components and features to support Application level services such as Connection Pooling, Object Pooling, Transaction Support, Messaging services, etc.
It acts as a container which serves static content.	It acts as container upon which business logic is built.
Web Server only supports Servlets and JSP.	Application Server supports distributed transaction and EJB.

Client-Server Model

CLIENT REQUEST TYPES

- GET
- POST
- PUT
- DELETE
- TRACE
- OPTIONS
- HEAD

Client-Server Model

CLIENT REQUEST TYPES: GET

GET
POST
PUT
DELETE
TRACE
OPTION
HEAD

- Used to GET the resource
- It can send limited data
- It is not secure, can be bookmarked and, the data is visible in URL
- It is the default method

Client-Server Model

CLIENT REQUEST TYPES: POST

- GET
- POST
- PUT
- DELETE
- TRACE
- OPTION
- HEAD

- It is used to process data, save data, change data, etc.
- It is secure and cannot be bookmarked
- No limit on data; sent as “payload” and not appended in URL

Client-Server Model

CLIENT REQUEST TYPES: PUT

GET
POST
PUT
DELETE
TRACE
OPTION
HEAD

- If the Uniform Resource Identifier (URI) refers to an already existing resource, it is modified. If the URI does not point to an existing resource, then the server can create the resource with that URI using PUT.

- It replaces all current representation of the target resource with the uploaded content.

Client-Server Model

CLIENT REQUEST TYPES: DELETE

GET
POST
PUT
DELETE
TRACE
OPTION
HEAD

- DELETE request is used to delete a document of the target resource given by URL.
- To allow eventual consistency while replication, deleted documents remain in the database forever.

Client-Server Model

CLIENT REQUEST TYPES: TRACE

GET
POST
PUT
DELETE
TRACE
OPTION
HEAD

- TRACE request is used when the client wants to see if any change has been done by intermediate server.
- This request works for targeted resources.

Client-Server Model

CLIENT REQUEST TYPES: OPTION

GET
POST
PUT
DELETE
TRACE
OPTION
HEAD

- OPTION request returns the HTTP method which is supported by the server.
- It allows the client to determine the option requirements associated with a resource.

Client-Server Model

CLIENT REQUEST TYPES: HEAD

GET
POST
PUT
DELETE
TRACE
OPTION
HEAD

- HEAD request is used for testing hypertext links for accessibility, validity, and recent modification.
- This is similar to GET method, but it asks for response without response body.

Client-Server Model

SERVER RESPONSE: HTML

- HTML stands for Hypertext Mark-up Language.
- It refers to the way the web pages (HTML documents) are linked together. The link available on a web page is called Hypertext.
- HTML contains tags and elements which are used to format web pages.

Client-Server Model

SERVER RESPONSE: HTML DOCUMENT STRUCTURE

A typical HTML document has the following structure:

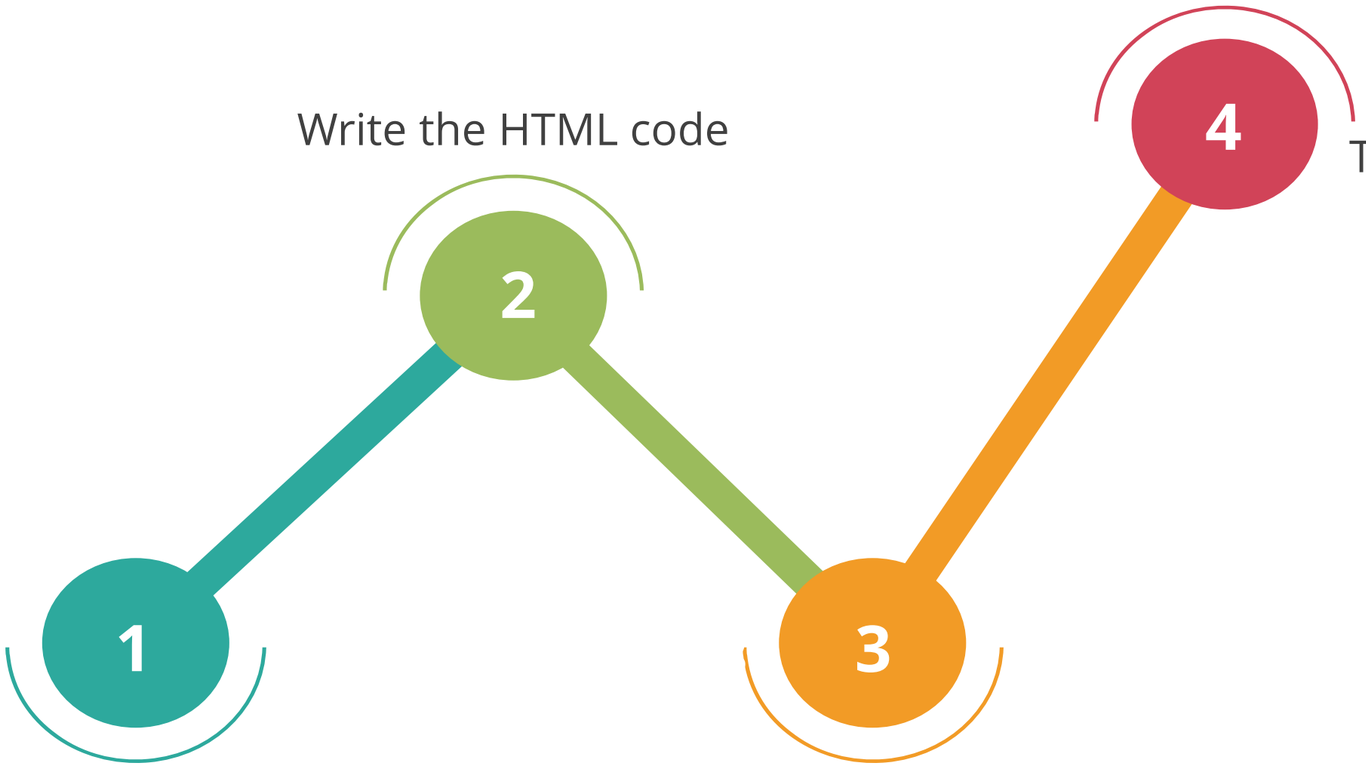
```
<!DOCTYPE html>
<html>
<head>
<title>Html Program </title>
</head>
<body>

<h1>This is a Heading</h1>
<p>welcome To HTML Program.</p>

</body>
</html>
```

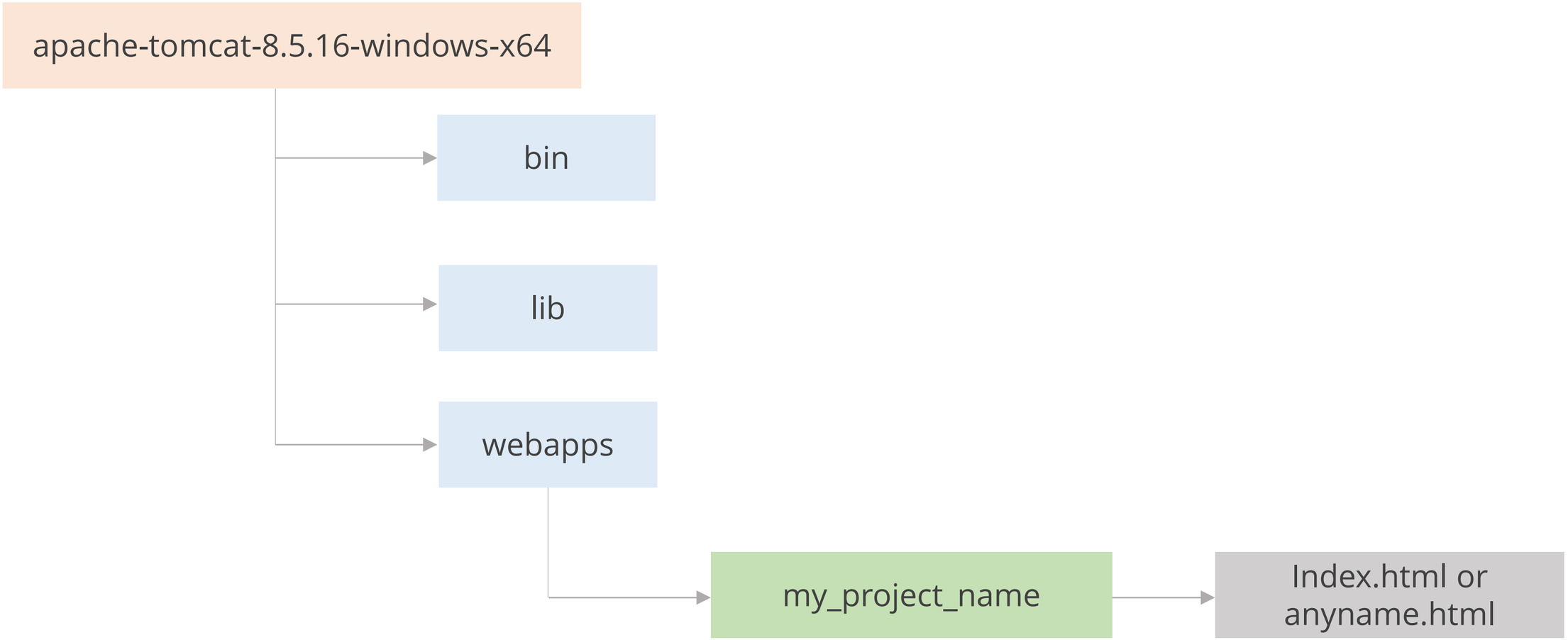
Client-Server Model

SERVER RESPONSE: RUNNING AN HTML CODE



Running an HTML Code with Apache

TOMCAT APACHE DIRECTORY HIERARCHY



Running an HTML Code with Apache

1. Create a sub-folder in webapps folder

2. Add a name to the project

3. Place your HTML page inside project folder with index.html name or any_other_name.html

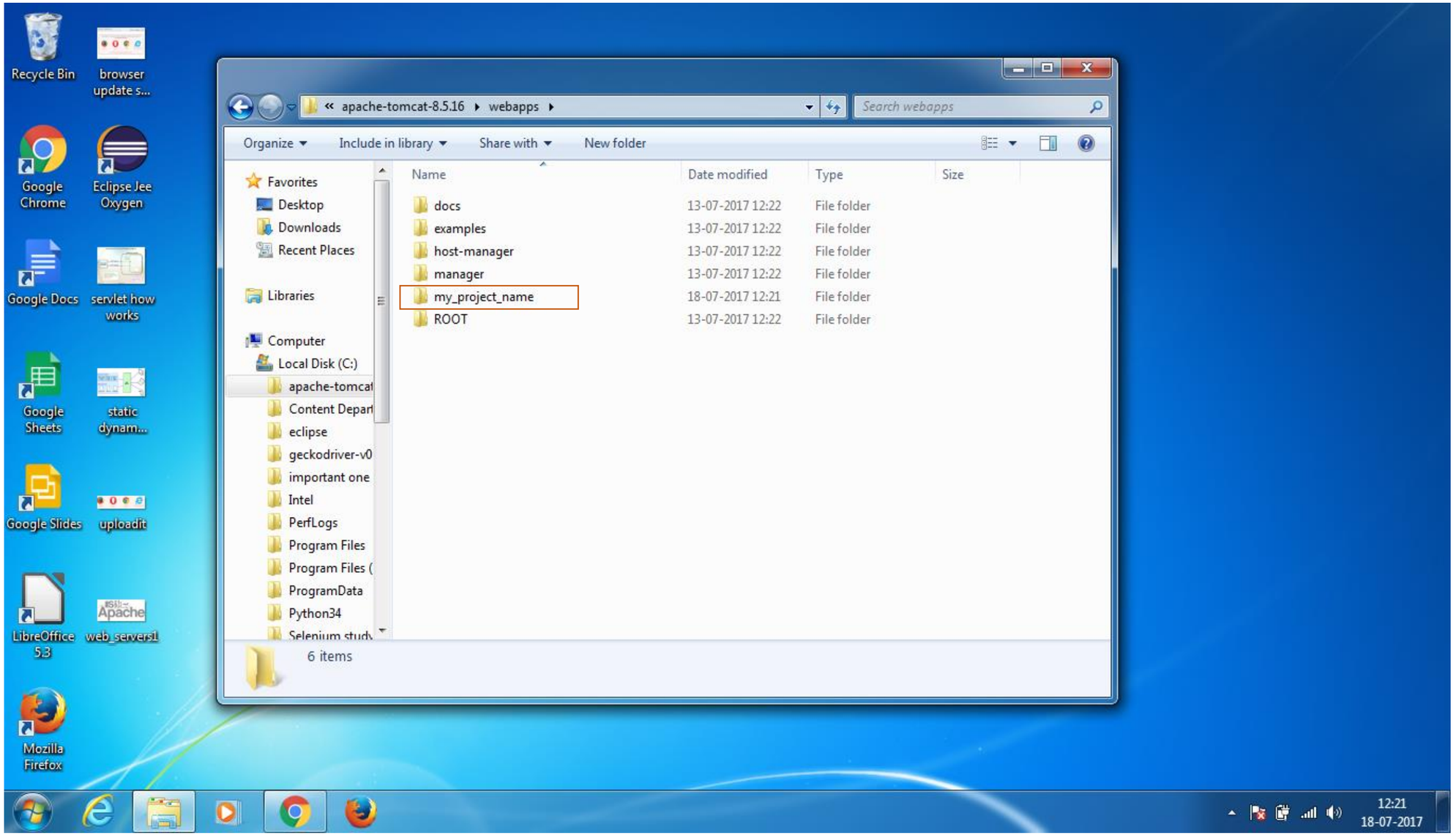
4. Open browser to send request through URL

5. Enter this for the URL: http://localhost:8080//my_project_name

6. If html filename is index.html, the above link will open an HTML page. If not, use the following:

http://localhost:8080//my_project_name/any_other_name.html

Screenshot of Project in Apache Webapp



Key Takeaways

- Java has three editions: J2SE (standard), J2EE (Enterprise), and J2ME (Micro)
- The J2EE (Enterprise Edition) differs from J2SE (Java Standard Edition Platform) in terms of the libraries it provides to deploy fault-tolerant, distributed and, multi-tier Java software
- In distributed applications, same application can run across different machines at the same time using internet. One is client and the other is Server.
- The client–server model is a distributed application structure that partitions tasks or workloads between the providers of a resource or service (web servers) and service requesters (web clients).

Quiz

Thank You