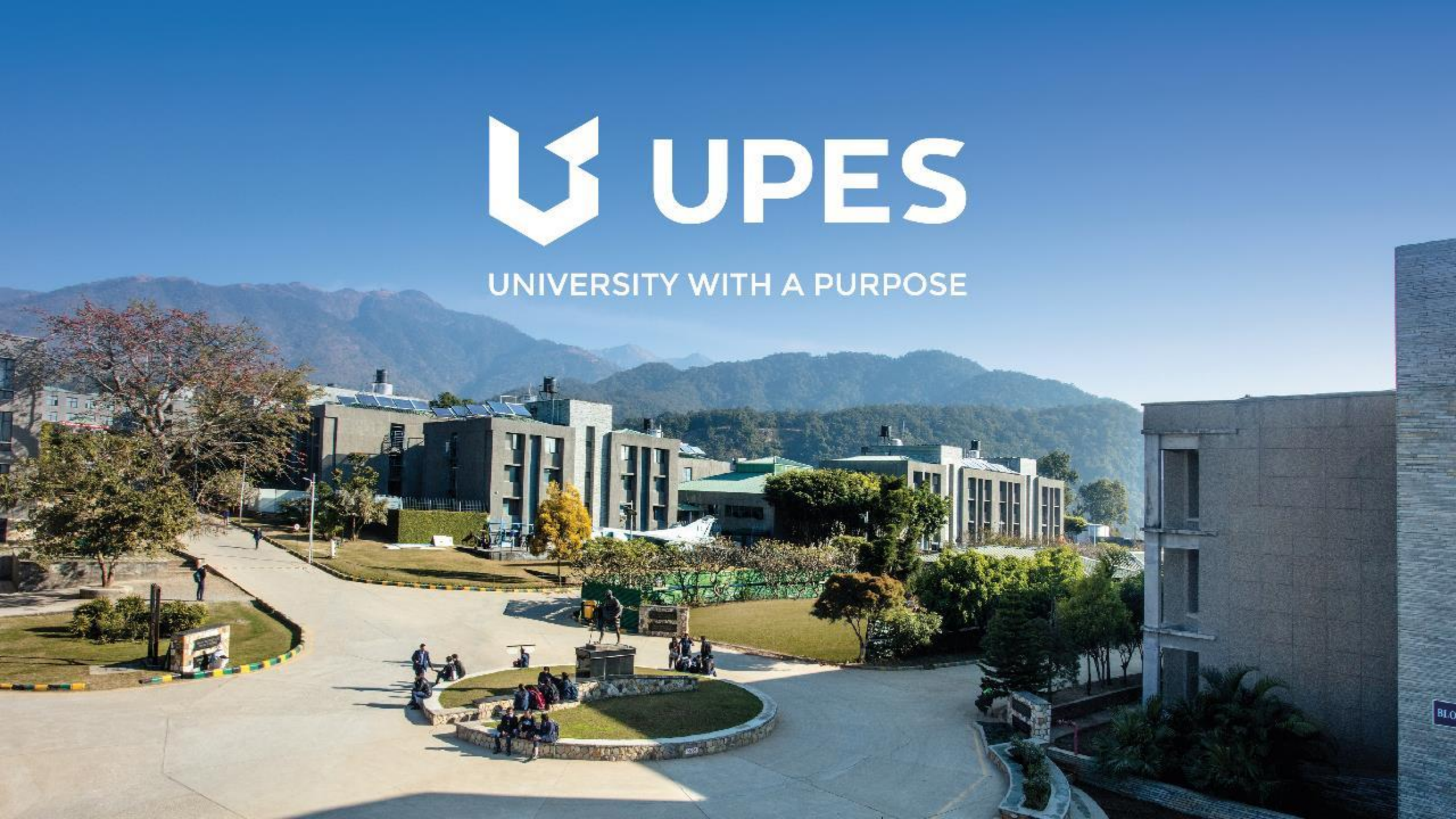
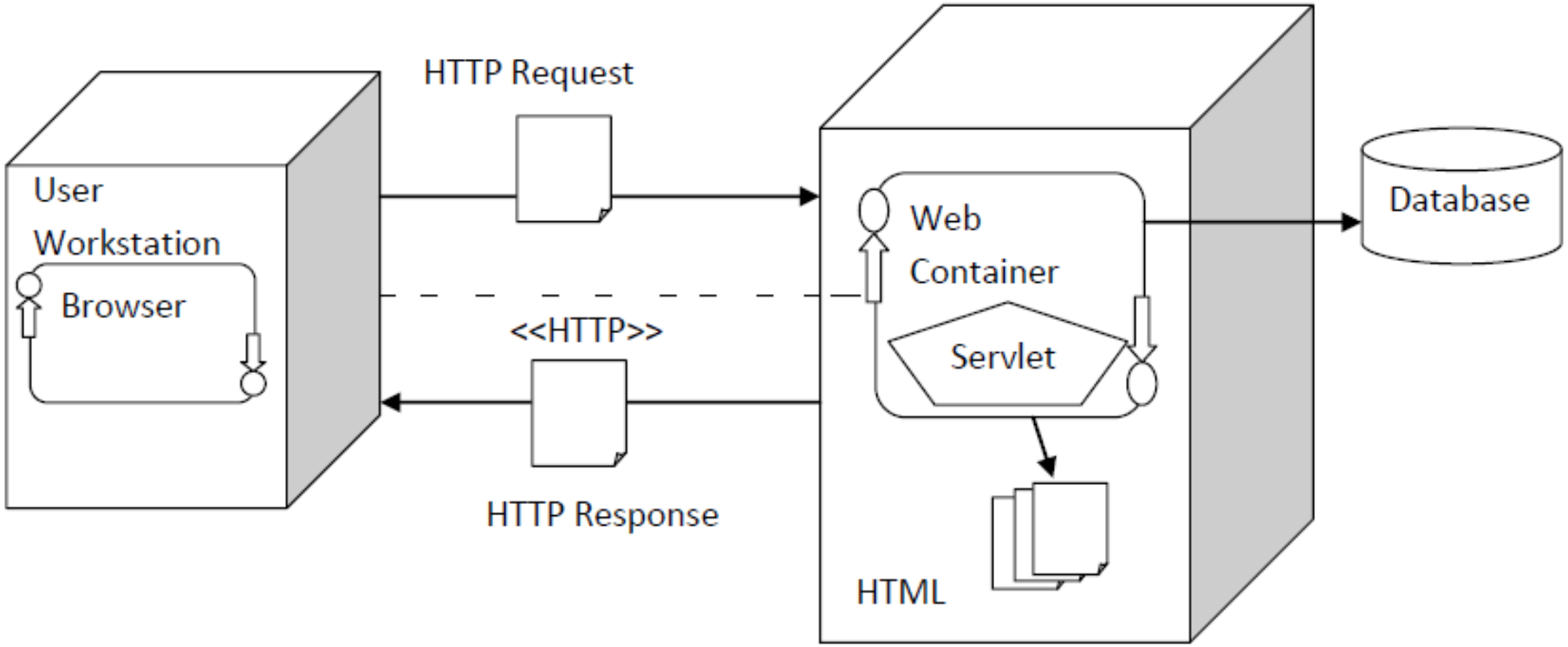

UPES
UNIVERSITY WITH A PURPOSE

UPES
UNIVERSITY WITH A PURPOSE

Servlet-II

Servlet


HTTP Request
User
Workstation
Browser
<<HTTP>>
HTTP Response
Web
Container
Servlet
HTML
Database

Servlet

- Java servlets are server-side programs (running inside a web server) that handle clients' requests and return a customized or dynamic response for each request.
- The dynamic response could be based on user's input (e.g., search, online shopping, online transaction) with data retrieved from databases or other applications, or time-sensitive data (such as news and stock prices).
- Java servlets typically run on the HTTP protocol.
- HTTP is a request-response protocol. The client sends a request message to the server. The server, in turn, returns a response message.
- The messages consists of two parts: header (information about the message) and body (contents). Header provides information about the messages.

Servlet Life Cycle

- When user request for dynamic contents from web server, even though web server has access to database it cannot produce web pages.
- Here Servlets comes into middle and generate dynamic web pages upon user requests.
- It’s not necessary that application from client side should written in java to get Servlets response.
- Hence application written in any programming language can consumes Servlets outputs.
- Servlet container popularly known as Servlet Engine which provide execution environment for Servlets and manage Servlets lifecycle from creation to destroy phase.

Static vs Dynamic website

Static Website	Dynamic Website
Prebuilt content is same every time the page is loaded.	Content is generated quickly and changes regularly.
It uses the **HTML** code for developing a website.	It uses the server side languages such as **PHP,SERVLET, JSP, and ASP.NET** etc. for developing a website.
It sends exactly the same response for every request.	It may generate different HTML for each of the request.
The content is only changed when someone publishes and updates the file (sends it to the web server).	The page contains "server-side" code which allows the server to generate the unique content when the page is loaded.
Flexibility is the main advantage of static website.	Content Management System (CMS) is the main advantage of dynamic website.

HTTP Requests

- The request sent by the computer to a web server, contains all sorts of potentially interesting information; it is known as HTTP requests.
- The HTTP request method indicates the method to be performed on the resource identified by the Requested URI (Uniform Resource Identifier). This method is case-sensitive and should be used in uppercase.

HTTP Request	Description
GET	Asks to get the resource at the requested URL.
POST	Asks the server to accept the body info attached. It is like GET request with extra info sent with the request.
HEAD	Asks for only the header part of whatever a GET would return. Just like GET but with no body.
TRACE	Asks for the loopback of the request message, for testing or troubleshooting.
PUT	Says to put the enclosed info (the body) at the requested URL.
DELETE	Says to delete the resource at the requested URL.
OPTIONS	Asks for a list of the HTTP methods to which the thing at the request URL can respond

Get vs. Post

GET	POST
1) In case of Get request, only **limited amount of data** can be sent because data is sent in header.	In case of post request, **large amount of data** can be sent because data is sent in body.
2) Get request is **not secured** because data is exposed in URL bar.	Post request is **secured** because data is not exposed in URL bar.
3) Get request **can be bookmarked.**	Post request **cannot be bookmarked.**
4) Get request is **idempotent** . It means second request will be ignored until response of first request is delivered	Post request is **non-idempotent.**
5) Get request is **more efficient** and used more than Post.	Post request is **less efficient** and used less than

Servlet Sample Program

To begin, create a file named **HelloServlet.java** that contains the following program:

```
import java.io.*;
import javax.Servlet.*;
public class HelloServlet extends GenericServlet
{
        public void service(ServletRequest request,
        ServletResponse response)
        throws ServletException, IOException
        {

            response.setContentType("text/html");
            PrintWriter pw = response.getWriter();
            pw.println("Hello!");
            pw.close();
        }
}
```

- **javax.servlet** package contains the classes and interfaces required to build servlets.
- **HelloServlet** as a subclass of **GenericServlet**.
- The **GenericServlet** class provides functionality that simplifies the creation of a servlet. For example, it provides versions of **init( )** and **destroy( )**, which may be used as is. You need supply only the **service( )** method.
- Inside **HelloServlet**, the **service( )** method (which is inherited from **GenericServlet**) is overridden. This method handles requests from a client. Notice that the first argument is a **ServletRequest** object. This enables the servlet to read data that is provided via the client request. The second argument is a **ServletResponse** object. This enables the servlet to formulate a response for the client.
- The call to **setContentType( )** establishes the MIME type of the HTTP response. In this program, the MIME type is text/html. This indicates that the browser should interpret the content as HTML source code.
- Next, the **getWriter( )** method obtains a **PrintWriter**. Anything written to this stream is sent to the client as part of the HTTP response. Then **println( )** is used to write some simple HTML source code as the HTTP response.

Setting up Eclipse for Servlet Programs:

Help->New Software-> Choose your version -> Web, XML, JavaEE

Window-> Perspective -> Other Perspective -> JavaEE

To Add Apache TomcatServer in Eclipse -> Download and unzip the server from https://tomcat.apache.org/download-90.cgi -> In Eclipse click on “Create a new server ” -> Click on Apache -> Select version of Tomcat and paste the path to the unzipped downloaded folder.

For creating a dynamic web project **click on File Menu -> New -> Project..-> dynamic web project -> write your project name e.g. first -> Finish**.

For creating a servlet, **explore the project by clicking the + icon -> explore the Java Resources -> right click on src -> New -> servlet -> write your servlet name e.g. Hello -> uncheck all the checkboxes except doGet() -> next -> Finish**

For adding a jar file, **right click on your project -> Build Path -> Configure Build Path -> click on Libraries tab in Java Build Path -> click on Add External JARs button -> select the servlet-api.jar file under tomcat/lib -> ok.**

THANK YOU