Module 4

Web Technologies in Java

- **HTML**

Tags: Anchor, Form, Table, Image, List Tags, Paragraph, Break, Label

QUES: Introduction to HTML and its structure.

HTML stands for Hyper Text Markup language, which is widely used for creating a web pages and web application.
It is a markup language not a programming language.
HTML is all about organizing and displaying information on a webpage.
The basic structure of an HTML page is shown below. It contain the essential building block elements(i.e. doctype declaration, HTML, Head , title and body elements) upon which all web pages are created.

```
<!DOCTYPE html>              → tells version of HTML
<html>                       → HTML root Element
<head>                       → used to contain page HTML metadata
   <title> page title </title>    →Title of HTML page
</head>
<body>                       →holds content of HTML
    <h2> Heading Content </h2>   → HTML heading tag
    <p> Paragraph Content </p>   → HTML Paragraph tag
</body>
</html>
```

QUES: Explanation of key tags

- <a> Anchor tag for hyperlinks.
- <form> form tag for user input.
- <table> tag for data representation.
- <img> image tag for embedding images
- , <ol>, and <li> for list tag.
- <p> paragraph tag
-
 line break
- <label> label for form input.

- **CSS (Inline CSS, Internal CSS, External CSS)**

QUES: Overview of CSS and its importance in web design.
A Cascading Style Sheet commonly known as CSS is a layer of design over HTML elements. It is a general idea that HTML is like a skeleton and CSS is the skin over it. To get a general idea, HTML is responsible for the basic structure of web pages, the contents and their sequence is determined by the HTML document but CSS styles the HTML document and makes it look presentable by adding font, size and color to it. It saves a lot of work and can control the layout of multiple web pages in one go. JavaScript is used to make websites interactive and dynamic.

It is a highly effective tool HTML that provides easy control over layout and presentation of website pages by separating content from design. It improves website presentation, makes updates easier and smoother and helps web pages load faster.

QUES: Types of CSS:

1. Inline CSS: Directly in HTML elements.
2. Internal CSS: Inside a <style> tag in the head section
3. External CSS: Linked to an external file.

CSS: Margin and Padding

QUES: Definition and difference between margin and padding.
Margin: Margin is the space outside the border of an element. It creates space between the element and its neighboring elements. We can create margin for all four sides (top, bottom, and right, left).
Padding: Padding is the space inside the border of an element, between the border and the content. It creates space between the content of the element and its border. We can set padding for all four sides like (top, bottom and right, left).

QUES: How margins create space outside the element and padding creates space inside.
Margin adds space outside an element, creating gaps between elements and padding adds space inside an element, between its content and border.
For example: In margin 20px; which adds 20 pixels of space outside the element, separating it from other element.
In padding 20px; which adds 20 pixels of space inside the element, between its content and its border.

CSS: Pseudo-Class

QUES: Use of pseudo-classes to style elements based on their state.
A pseudo-class is a keyword added to a CSS selector, prefixed by colon (:), to define a specific state or condition of an element.

:hover → This applies when the user hover over an element and this will change background color of the button.
:focus → This is applies when an element receives focus like text input clicked. This will change the border of the input field.
:active → applies when an element is being clicked, this will change the background color of the button when it is clicked.
:visited → applies to links visitor has visited.this will change the color of the visited links to purple.
:link → applies to links that the user has not visited yet. This will remain unvisited links.

CSS: ID and Class Selectors

Ques: Difference between id and class in CSS.

ID Attribute	Class Attribute
Identifies only one element.	Can be applied to many element
Primarily used for styling.	Class also use for styling
Only one element have one id.	Multiple element can share the same class.

Introduction to Client-Server Architecture

QUES: Overview of client-server architecture.
Client server architecture is a fundamental of modern system design. Where network include multiple client and central server. Here, clients are devices or programs that makes request for services or resources, while server is powerful machine or software that fulfills these request. Communication between clients and the servers follows a request- response protocol such as HTTP/HTTPS for web services or SQL for database queries.

QUES: Difference between client-side and server-side processing.
Client-side scripting refers to scripts that are executed on the user's browser rather than on the web server. These scripts are typically embedded within HTML and are used to enhance the user experience by making web pages more interactive and responsive.
Server-side scripting involves scripts that are executed on the web server. These scripts generate dynamic content and interact with databases to provide customized responses to user requests.

QUES: Roles of a client, server, and communication protocols.

- Client: The client is a device or application that requests services or resources from the server.
- Server: The server is a powerful machine or application that provides services or resources to clients.

- Protocols: Protocols are standardized rules that govern data transmission and communication between clients and servers.

HTTP Protocol Overview with Request and Response Headers

QUES: Introduction to the HTTP protocol and its role in web communication.
HTTP is Hypertext Transfer Protocol is a fundamental protocol of internet, which enables the transfer of data between a client and a server.
It is a foundation of data communication for the World Wide Web.
Http provides connection between the web browser and web server.
It transfers the data like text, images and other multimedia files from one computer to another.

QUES: Explanation of HTTP request and response headers.
HTTP headers are key-value pairs sent in HTTP requests and responses, providing essential information about the communication between the client and server. They include details such as content type, encoding, cache control, authentication, and more, helping manage the behavior of HTTP transactions.

Request Header: This type of header contains information about the fetched request by the client.

Response Header: This type of header contains the location of the source that has been requested by the client.

J2EE Architecture Overview

QUES: Introduction to J2EE and its multi-tier architecture.

J2EE (Java 2 Platform, Enterprise Edition), now known as Jakarta EE, is a platform for building enterprise-level applications.
It provides a multi-tier architecture that divides the application into different layers, each responsible for specific functionalities.
This architecture enhances scalability, maintainability, and flexibility.

FOUR TIER ARCHITECHTURE

1. Client Tier/Presentation Tier
   Client tier consist of program that interact with user. These component prompt the user for input, process the request and return the result.
2. Web Tier
   The Web Tier provides internet functionality to the J2EE application. It uses HTTP to receive requests and send responses to the client.
   3.Enterprise JavaBeans Tier

This tier contains the business logic for J2EE applications. It includes EJBs that manage transactions, security, multi-threading, and persistence.

4.Enterprise Information System Tier

This Tier links with corporate network.It interfaces with technologies like DBMS and mainframes using CORBA or Java Connectors.

QUES: Role of web containers, application servers, and database servers.

A web container, also known as a servlet container, is a component of a web server that interacts with Java Servlets. It is responsible for managing the lifecycle of servlets, mapping URLs to specific servlets, and ensuring that the URL requester has the correct access rights. The web container handles requests to servlets, Jakarta Server Pages (JSP) files, and other types of files that include server-side code.

An application server is a server program in a computer in a distributed network that provides the business logic for an application program.

A database server is a computer system that provides other computers with services related to accessing and retrieving data from a database.

Web Component Development in Java (CGI Programming)

QUES: Introduction to CGI (Common Gateway Interface).

CGI is a standard protocol that allows web servers to interact with external programs (usually scripts or applications) to generate dynamic content in response to user requests.

CGI provides a way for web servers to run external programs, which can be written in various programming languages like Python, PHP, or even C. When a user submits a request (e.g., by filling out a form or clicking a link), the web server runs a CGI script or program, processes the data, and returns a response, typically in HTML format, back to the user's browser.

QUES: Process, advantages, and disadvantages of CGI programming.

Process of CGI Programming

1. **Client Request**: A user sends a request to the web server, usually through HTML page.
2. **Server Execution**: The web server receives the request and identifies the corresponding CGI script or program to handle it. This script is executed by the server.
3. **Environment Variables**: The web server passes various environment variables to the CGI script, such as information about the HTTP request, client information, and form data.

4. **Data Processing**: The CGI script processes the input data. It can interact with databases, perform calculations, or process forms.
5. **Output Generation**: The CGI script generates an output, often in HTML format, which is sent back to the web server.
6. **Response to Client**: The web server sends the generated HTML or other content as the response to the client's browser, which is then rendered for the user.

Advantage : Language Flexibility, Platform independent, secure etc.

Disadvantage: complexity, security risk, performance overhead, limited scalability etc.

Servlet Programming: Introduction, Advantages, and Disadvantages

QUES: Introduction to servlets and how they work. Advantages and disadvantages compared to other web technologies.

- Servlets is a technology which is used to create a web application.
- Servlet is an API that provides many interfaces and classes including documentation.
- Servlet is an interface that must be implemented for creating any Servlet.
- Servlet is a class that extends the capabilities of the servers and responds to the incoming requests. It can respond to any requests.
- Servlet is a web component that is deployed on the server to create a dynamic web page.

How Servlets work

It is important to learn how servlet works for understanding the servlet well. Here, we are going to get the internal detail about the first servlet program.

The server checks if the servlet is requested **for the first time**.

If yes, web container does the following tasks:

- loads the servlet class.
- instantiates the servlet class.
- calls the init method passing the ServletConfig object

else

- calls the service method passing request and response objects

There are many advantages of Servlet over CGI.

1. **Better performance:** because it creates a thread for each request, not process.
2. **Portability:** because it uses Java language.
3. **Robust:** JVM manages Servlets, so we don't need to worry about the memory leak, garbage collection, etc.
4. **Secure:** because it uses java language.

There are many disadvantages of Servlet.

1. Requires advanced Java knowledge.
2. Implicit objects are not efficiently utilized.
3. Exception handling needs to be specifically coded.
4. Challenging to write HTML code in Servlet programming.
5. Complexity and overhead.
6. Challenging to scale.
7. Time-consuming.

Servlet Versions, Types of Servlets

QUES: History of servlet versions.

Java Servlet is a technology that allows programmers to create dynamic websites using Java programming language. A servlet is a server-side component that can receive requests from clients, interact with other business components, and send responses back to the clients.

Since its first announcement in 1996, Java Servlet has become the foundational technology of Java web application development stack. The following table provides an overview of the history of Java Servlet in terms of Java Servlet API specification versions and the corresponding Java EE platform.

Java Servlet is part of Java EE. And since Oracle donated the Java EE platform to the Eclipse Foundation at the end of 2017, Java Servlet has been renamed ***Jakarta Servlet***.

Java Servlet Version	Release Time	Java EE Platform
Java Servlet 1.0	December 1996	
Java Servlet 2.0	December 1997	
Java Servlet 2.1	November 1998	
Java Servlet 2.2	August 1999	J2EE 1.2
Java Servlet 2.3	August 2001	J2EE 1.3
Java Servlet 2.4	November 2003	J2EE 1.4
Java Servlet 2.5	September 2005	Java EE 5
Java Servlet 3.0	December 2009	Java EE 6
Java Servlet 3.1	May 2013	Java EE 7
Java Servlet 4.0	September 2017	Java EE 8

QUES: Types of servlets: Generic and HTTP servlets.

Generic Servlets: These are those servlets that provide functionality for implementing a servlet. It is a generic class from which all the customizable servlets are derived. It is protocol-independent and provides support for HTTP, FTP, and SMTP protocols. The class used is '**javax.servlet.Servlet'** and it only has 2 methods – init() to initialize & allocate memory to the servlet and destroy() to deallocate the servlet.

HTTP Servlets: These are protocol dependent servlets, that provides support for HTTP request and response. It is typically used to create web apps And has two of the most used methods – doGET() and doPOST() each serving their own purpose.

QUES: Detailed comparison between HttpServlet and GenericServlet.●

1. **Protocol Dependency**: GenericServlet: Protocol-independent and can be used with any protocol. HttpServlet: Protocol-dependent and specifically designed for HTTP.
2. **Service Method**: GenericServlet: The service() method is abstract. HttpServlet: The service() method is non-abstract and is overridden to handle HTTP requests.
3. **Usage**: GenericServlet: Not commonly used in modern web applications. HttpServlet: Commonly used for handling HTTP requests in web applications.
4. **Package Definition**: GenericServlet: Defined in the javax.servlet package. HttpServlet: Defined in the javax.servlet.http package.
5. **Inheritance**: GenericServlet: Extends Object class and implements Servlet, ServletConfig, and Serializable interfaces. HttpServlet: Extends GenericServlet and implements Serializable interface^1^.

QUES: Explanation of the servlet life cycle: init(), service(), and destroy() methods.

The life cycle of a servlet is managed by the servlet container and involves several stages from creation to destruction. Understanding these stages is crucial for developing efficient and effective servlets.

Stages of the Servlet Life Cycle

1. **Loading a Servlet**: The servlet container loads the servlet class when the first request for the servlet is received or when the server starts, depending on the configuration
2. **Initializing the Servlet**: After the servlet instance is created, the servlet container initializes it by calling the `init(ServletConfig config)` method. This method is called only once and is used to perform any one-time setup tasks, such as establishing database connections[2].
3. **Handling Requests**: Once initialized, the servlet is ready to handle client requests. The servlet container creates `ServletRequest` and `ServletResponse` objects and passes them to the `service(ServletRequest req, ServletResponse res)` method. This method determines the type of request (GET, POST, etc.) and calls the appropriate method (e.g., `doGet`, `doPost`)[3].
4. **Destroying the Servlet**: When the servlet is no longer needed, the servlet container calls the `destroy()` method. This method is called only once and allows the servlet to release any resources it holds, such as closing database connections. After this method is called, the servlet instance is eligible for garbage collection[1].

Life Cycle Methods

1. **init() Method**

The `init()` method is called by the servlet container to indicate that the servlet is being placed into or one-time initialization tasks.

2. **service() Method**

   The `service()` method is the main method for handling client requests. It is called each time a request is made to the servlet.

3. **destroy() Method**

The `destroy()` method is called once at the end of the servlet's life cycle. It is used to perform cleanup tasks.

Creating Servlets and Servlet Entry in web.xml

QUES: How to create servlets and configure them using web.xml.

Steps to create a Servlet.

In this we will create a basic servlet .

Step 1: Create a Dynamic web project (in eclipse go to file →New→Dynamic Web Project and click on it.

Step 2: fill required details.

This web module will have our all the HTML and JSP pages.

We can also generate a Deployment Descriptor – **web.xml** file to define the mappings between URL paths and the servlets to handle the requests with those paths. This can also be done using @WebServlet() annotation in the Servlet class.

here we are working with the Servlets , we need to have the servlet api.jar file in our project. This jar is a library that contain all the interfaces and classess of the Servlets API , also we can use its functionality to develop our web application

step 3: Create Servlet Class
To create a Servlet, go to folder src -> New -> Servlet.
If the Servlet option is not there, go to Other and search for Servlet.

Step 4: Implement the logic
In the doGet() method, implement the logic to display the welcome message to the user.

Step 5: Run the Project
Right-click on the HelloServlet.java class, Run As -> Run on Server.

Logical URL and ServletConfig Interface

QUES: Explanation of logical URLs and their use in servlets.
A logical URL provides an identifier for a resource. It does not use implementation references, nor does it indicate a content (MIME) type,
Example: http://www.airlineInc.com/7489

QUES: Overview of ServletConfig and its methods.
Servlet container uses the object of ServletConfig interface to provide the information to servlet during initialization. ServletConfig object fetch this information from web.xml file. So if

we have to provide some dynamic information to servlet then we can pass it from web.xml file without making any change in our servlet class.

Methods of ServletConfig

String getInitParameter(String name)	This method returns the value of specific parameter name.
Enumeration getInitParameterNames()	This method provides the enumeration of names.
ServletContext getServletContext()	This method returns the object of ServletContext.
String getServletName()	This method returns the name of servlet instance.

RequestDispatcher Interface: Forward and Include Methods

QUES: Explanation of RequestDispatcher and the forward() and include() methods.

The RequestDispatcher interface provides the facility of dispatching the request to another resource it may be html, servlet or jsp. This interface can also be used to include the content of another resource also. It is one of the way of servlet collaboration.

Methods of RequestDispatcher interface

1. **public void forward(ServletRequest request,ServletResponse response)throws ServletException,java.io.IOException**

   (Forwards a request from a servlet to another resource (servlet, JSP file, or HTML file) on the server.)

2. **public void include(ServletRequest request,ServletResponse response)throws ServletException,java.io.IOException**

   (includes the content of a resource (servlet, JSP page, or HTML file) in the response.)

ServletContext Interface and Web Application Listener

Introduction to ServletContext and its scope.

An object of ServletContext is created by the web container at time of deploying the project. This object can be used to get configuration information from web.xml file. There is only one ServletContext object per web application.

The object of ServletContext provides an interface between the container and servlet.

The ServletContext object can be used to get configuration information from the web.xml file.
The ServletContext object can be used to set, get or remove attribute from the web.xml file.
The ServletContext object can be used to provide inter-application communication.

Java Filters: Introduction and Filter Life Cycle

Ques: What are filters in Java and when are they needed?

In Java, filters are typically used in the context of **servlets** and web applications, especially within Java EE (Jakarta EE) environments. They are components that allow you to process requests and responses in a flexible, reusable way. Filters can be used to modify request and response objects before they reach the servlet or after the servlet has processed them.

Filters are useful when you want to apply the same logic to multiple servlets or pages without duplicating code.

Ques: Filter lifecycle and how to configure them in web.xml.

In a Java web application, the filter lifecycle is crucial for controlling and manipulating the request and response objects before they reach a servlet or after the servlet processes them. Filters are commonly used for tasks such as authentication, logging, data compression, and input sanitization.

Steps to Configure a Filter:

1. Define the filter:
    - Use `<filter>` to define the filter class (`<filter-class>`).
2. Map the filter to specific URL patterns:
    - Use `<filter-mapping>` to specify the URLs where the filter should be applied using `<url-pattern>`.

JSP Basics: JSTL, Custom Tags, Scriplets, and Implicit Objects

Ques: Introduction to JSP and its key components: JSTL, custom tags, scriplets, and implicit objects.

JavaServer Pages (JSP) is a technology used to develop dynamic web applications. It is part of the Java EE (Enterprise Edition) and allows developers to embed Java code in HTML pages, making it easier to create interactive and data-driven websites. JSP works by transforming the JSP page into a Java servlet on the server side, which is then executed to generate dynamic content.JSP allows for separation of concerns between the user interface (HTML, CSS) and business logic (Java code), which is crucial for maintainability and scalability of web applications.

Key Components of JSP

1. JSTL (JavaServer Pages Standard Tag Library):

- JSTL is a set of standard tags that help in simplifying JSP code. Instead of writing Java code for common tasks (like loops, conditionals, or formatting data), you can use JSTL tags, making the JSP page cleaner and easier to maintain.
- Key JSTL tags include:
  - Core Tags: for conditional operations, looping, and variable management.
  - Formatting Tags: for formatting dates, numbers, etc.
  - SQL Tags: for accessing and manipulating databases.
  - XML Tags: for working with XML documents.

2. Custom Tags:

- Custom tags allow developers to create their own tags in JSP, enabling more complex or reusable components than the built-in tags.

3. Scriplets:

- Scriplets are blocks of Java code embedded directly within a JSP page. They are enclosed in `<% %>` tags. While scriplets are a part of the original JSP specification, their use is discouraged in favor of more modern approaches like JSTL and EL (Expression Language), as scriplets can lead to cluttered code and tightly coupled logic.

Session Management and Cookies

QUES: Overview of session management techniques: cookies, hidden form fields, URL rewriting, and sessions.

Session management is crucial for maintaining state across multiple requests in a stateless protocol like HTTP. Different techniques are used to track and manage user sessions. Here's an overview of four common session management techniques**:** cookies**,** hidden form fields**,** URL rewriting**,** and sessions**.**

Cookies

How it works: Cookies are small pieces of data stored on the client's browser. When a user visits a website, the server sends a cookie to the browser, which then stores it locally. On subsequent requests, the browser sends the cookie back to the server.

Hidden Form Fields

How it works: Hidden form fields are used to store data that is passed between different pages on the web. These fields are embedded in HTML forms, but they are not visible to the user. When a user submits a form, the data in these fields is sent back to the server.

URL Rewriting

How it works: URL rewriting involves appending session information to the URL (usually as a query string or path parameter). The session identifier is passed between pages via URLs, and the server can use this to associate the request with a specific user session.

Sessions

How it works: A session is typically stored on the server side and contains information about the user's interactions. A unique session ID is generated and assigned to the user, often stored in a cookie or passed through URL rewriting. The session ID is sent with each request, and the server retrieves the associated session data.