Explain JSP with anatomy of a JSP Page and it's elements.

JavaServer Pages (JSP) is a technology used for developing dynamic web pages in Java. JSP allows developers to embed Java code directly into HTML pages, making it easier to create dynamic content.

```
<%@ page language="java" contentType="text/html; charset=UTF-8" %>  <%--JSP Directive--%>
<!DOCTYPE html>                                                     <%--HTML Structure--%>
<html>
<head>
    <title>Example JSP Page</title>
</head>
<body>

    <h1>Hello, <%= request.getParameter("name") %></h1>

    <%-- JSP Declaration --%>
    <%!
        int count = 0;
        public String getMessage() {
            return "This is a JSP message.";
        }
    %>

    <%-- JSP Scriptlet --%>
    <%
        for (int i = 0; i < 3; i++) {
    %>
        <p>Loop iteration <%= i + 1 %></p>
    <% } %>

    <%-- JSP Expression: --%>
    <p>The count is <%= count %></p>

    <%-- JSP Action  --%>
    <jsp:include page="included.jsp" />

</body>
</html>
```

Anatomy of the JSP Page:

1. **JSP Directive:** Sets global properties for the JSP page, such as the language used and content type.
2. **HTML Structure:** Defines the appearance and structure of the page
3. **JSP Expression:** Outputs dynamic content based on a Java expression
4. **JSP Declaration:** Allows the declaration of variables and methods in Java
5. **JSP Scriptlet:** Embeds Java code directly into the HTML to perform dynamic actions
6. **JSP Action (Include):** Dynamically includes the content of another JSP page
7. **Comments:** Used for comments within JSP code

Describe a general JDBC Architecture and name the common JDBC API components?

1. **Two-tier model:** The java application communicates directly with the data source. When a user sends a query to the data source, the results are sent back to the user.
2. **Three-tier model:** In this, the user's queries are sent to middle-tier services, from which the commands are again sent to the data source. The results are sent back to the middle tier, and from there to the user.

Common JDBC API Components:

- **DriverManager:** Manages JDBC drivers.
- **Driver:** Implements JDBC API for a specific database.
- **Connection:** Represents a connection to the database.
- **Statement:** Executes SQL statements.
- **PreparedStatement:** Precompiled SQL statement for better performance.
- **ResultSet:** Holds results of an SQL query.

Explain the steps how to create a simple MVC application using Servlet and JSP.

1. Define the Model: Create a Java class representing your application's data model. This class encapsulates data and includes logic for manipulation.

2. Implement the Controller Servlet: Create a Servlet class to handle user requests and interact with the model and view. The servlet should:

- Read request parameters and identify the desired action.
- Retrieve data from the model based on the action.
- Generate appropriate data for the view.
- Forward the request to the appropriate JSP page with the data.

3. Develop the View JSP Pages: Create JSP pages for different views of your application. These pages display data received from the controller.

4. Configure Deployment Descriptors: Create a web.xml file to configure application deployment on the web server. This file defines servlets, JSP pages, and URL-to-servlet mappings.

Describe the constrained properties and Introspection of Java Bean?

Constrained properties are a special type of bound property in JavaBeans. They provide a way to allow interested parties (other beans or components) to influence changes to the property value before it is actually changed. This helps in maintaining data consistency and enforcing business rules.

Benefits of using constrained properties:

- Data consistency
- Data integrity
- Enforced business rules
- Modular design

Introspection is the ability of a JavaBean to provide information about its properties, methods, and events to other components. This information is used by tools and frameworks to interact with the bean and understand its capabilities.

Benefits of introspection:

- Automatic tool interaction
- Dynamic binding
- Simplified development

Write a XML Program to display the information about the Voter in the Voter Id Card. Describe your own tags and write appropriate DTD for it.

XML Program

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE VoterIDCard SYSTEM "VoterIDCard.dtd">

<VoterIDCard>
    <PersonalInformation>
        <Name>John Doe</Name>
        <Gender>Male</Gender>
        <Nationality>US Citizen</Nationality>
    </PersonalInformation>

    <Address>
        <ResidentialAddress>
            <Street>123 Main St</Street>
            <City>Anytown</City>
        </ResidentialAddress>
    </Address>

    <IDCardDetails>
        <CardNumber>ABC123XYZ456</CardNumber>
```

```
    </IDCardDetails>
</VoterIDCard>
```

VoterIDCard.dtd

```
<!ELEMENT VoterIDCard (PersonalInformation, Address, IDCardDetails)>

<!ELEMENT PersonalInformation (Name, Gender, Nationality)>
    <!ELEMENT Name (#PCDATA)>
    <!ELEMENT Gender (#PCDATA)>
    <!ELEMENT Nationality (#PCDATA)>

<!ELEMENT Address (ResidentialAddress)>
    <!ELEMENT ResidentialAddress (Street, City)>
        <!ELEMENT Street (#PCDATA)>
        <!ELEMENT City (#PCDATA)>

<!ELEMENT IDCardDetails (CardNumber)>
    <!ELEMENT CardNumber (#PCDATA)>
```

Write complete HTML Program to design a 'Online Shopping' Web site like Flipkart or Snapdeal etc.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Online Shopping</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            margin: 0;
            padding: 0;
            background-color: #f4f4f4;
        }

        header {
            background-color: #333;
            color: #fff;
            padding: 1em;
            text-align: center;
        }

        section {
            margin: 20px;
            padding: 20px;
            background-color: #fff;

        }
    </style>
</head>
```

```
<body>
    <header>
        <h1>Online Shopping</h1>
    </header>

    <section>
        <h2>Featured Products</h2>
        <div class="product">
            <img src="product1.jpg" alt="Product 1">
            <h3>Product 1</h3>
            <p>Description of Product 1.</p>
            <p>$19.99</p>
            <button>Add to Cart</button>
        </div>
        <div class="product">
            <img src="product2.jpg" alt="Product 2">
            <h3>Product 2</h3>
            <p>Description of Product 2.</p>
            <p>$29.99</p>
            <button>Add to Cart</button>
        </div>

    </section>

    <footer>
        <p>&copy; Made by Yash Aryan</p>
    </footer>
</body>
</html>
```

JSDK

The Java Servlet Developers Kit (JSDK) is an add-on to the standard Java Development Kit (JDK). It provides additional libraries and tools specifically for developing web applications using Java Servlets and JSPs.

Key Components of JSDK:

- **Servlet Runner:** This is a tool that allows developers to test servlets without needing a web server.
- **Servlet Templates:** These are pre-built servlet templates for tasks like session management and request forwarding.
- **Tag Libraries:** These libraries provide custom JSP tags for specific tasks like database access and form handling.

Deployment Descriptor File

In a java web application a file named web.xml is known as deployment descriptor. It is a xml file and <web-app> is the root element for it. When a request comes web server uses web.xml file to map the URL of the request to the specific code that handle the request. It resides in the webapp under the WEB-INF directory.

Sample web.xml code

```
<web-app>

	<servlet>
		<servlet-name>abc</servlet-name>
		<servlet-class>com.yash.aryan</servlet-class>
	</servlet>

	<servlet-mapping>
		<servlet-name>abc</servlet-name>
		<url-pattern>/add</url-pattern>
	</servlet-mapping>

</web-app>
```