Assignment 7

Q1.What is the use of JDBC in java?

ans)JDBC (Java Database Connectivity) is a Java API that allows Java programs to interact with relational databases. It provides a set of classes and methods to establish a connection with a database, execute SQL queries, and perform database operations such as inserting, updating, and deleting data.

The main uses of JDBC are:

- Database connectivity: JDBC enables Java applications to connect to various databases, regardless of the underlying database technology.
- SQL query execution: JDBC allows executing SQL statements and retrieving the result sets from the database.
- Database operations: JDBC provides methods to perform common database operations such as inserting, updating, and deleting records.
- Transaction management: JDBC supports transaction management, allowing developers to control the atomicity and consistency of database operations.

Q2.What are the steps involved in JDBC?

ans)The steps involved in using JDBC (Java Database Connectivity) are as follows:

1. Import JDBC packages: Import the necessary JDBC packages into your Java program.
2. Load and register the JDBC driver: Load the JDBC driver class specific to your database vendor using

the **Class.forName()** method. This step is required to make the driver available for establishing a database connection.

3. Establish a database connection: Use the **DriverManager.getConnection()** method to establish a connection to the database. Provide the database URL, username, and password as parameters to this method.

4. Create a statement: Create a “Statement” or “PreparedStatement” object from the `Connection` to execute SQL queries or commands. Use the **createStatement()** or **prepareStatement()** methods to create these objects.

5. Execute SQL queries or commands: Use the “executeQuery()” method of the “Statement” or “PreparedStatement” object to execute SELECT queries and retrieve the result set.

6. Process the result set: If executing a SELECT query, process the returned result set using methods such as **next()**, **getString()**, **getInt()**, etc., to retrieve the data.

7. Close the resources: Use the **close()** method for this purpose.

Q3.What are the types of statements in JDBC in java?

ans)
There are three types of statements in JDBC in Java:

1. Statement
2. PreparedStatement
3. CallableStatement

- Statement is the most basic type of statement in JDBC. It is used to execute simple SQL statements, such as SELECT, INSERT, UPDATE, and DELETE.
- PreparedStatement is a more powerful type of statement. It is used to execute SQL statements that contain parameters. This allows the statement to be executed multiple times with different values for the parameters. This can improve performance, as the statement does not have to be parsed each time it is executed.

- CallableStatement is used to execute stored procedures. Stored procedures are precompiled SQL statements that are stored in the database. They can be used to encapsulate complex functionality, such as business logic or validation.

Q4.What is Servlet in Java?

ans)A servlet is a Java class that is used to extend the capabilities of a web server. Servlets can respond to any type of request, but they are most commonly used to handle HTTP requests and responses. Servlets are typically used to create dynamic web pages, but they can also be used to perform other tasks, such as processing form data, accessing databases, and sending email.

Q5.Explain the life Cycle of servlet?

ans)The life cycle of a servlet is the sequence of events that occur from the time a servlet is created until the time it is destroyed. The life cycle of a servlet is managed by the servlet container. The following are the main stages of the servlet life cycle:

1. **Loading:** The servlet container loads the servlet class into memory.
2. **Initialization:** The servlet container calls the servlet's init() method.
3. **Request handling:** The servlet container calls the servlet's service() method to handle each HTTP request.
4. **Destruction:** The servlet container calls the servlet's destroy() method when the servlet is no longer needed.

The servlet life cycle can be divided into two main phases: the initialization phase and the request handling phase. The initialization phase occurs once, when the servlet is first loaded by the servlet container. The request handling phase occurs each time the servlet receives an HTTP request.

Q6.Explain the difference between the RequestDispatcher.forward() and HttpServletResponse.sendRedirect() methods?

ans)

RequestDispatcher.forward()	HttpServletResponse.sendRedirect()
Server-side redirect	Client-side redirect

The request is internally forwarded to a different resource within the same server	The browser is instructed to send a new request to a different URL
The URL in the browser's address bar remains unchanged	The URL in the browser's address bar changes to the redirected URL
The request attributes and parameters are preserved and accessible in the forwarded resource	The request attributes and parameters are not preserved in the redirected resource
The forward is performed within the server and the client is unaware of the internal redirection	The redirect is performed by the client browser and the client is aware of the new URL
Suitable for server-side processing or including content from different resources	Suitable for redirecting the client to a different page or external URL
The forwarded resource is typically within the same web application or context	The redirected URL can be within the same server or a different server

Q7.What is the purpose of the doGet() and doPost() methods in a servlet?

ans)

⇒The doGet() method is called when the client makes a GET request to the servlet. GET requests are typically used to retrieve information from the server. The doGet() method can be used to retrieve data from a database, generate a web page, or perform

any other task that requires retrieving information from the server.
⇒The doPost() method is called when the client makes a POST request to the servlet. POST requests are typically used to send information to the server. The doPost() method can be used to process form data, upload files, or perform any other task that requires sending information to the server.

Q8.Explain the JSP Model-View-Controller (MVC) architecture.

ans)The JSP Model-View-Controller (MVC) architecture is a design pattern that separates the presentation layer, the business logic layer, and the data access layer of a web application. This separation of concerns makes it easier to develop, maintain, and test web applications.

The MVC architecture consists of three main components:

⇒The model represents the data of the application. It is responsible for storing and retrieving data from the database.
⇒The view represents the presentation of the application. It is responsible for displaying the data to the user.
⇒The controller acts as a mediator between the model and the view. It is responsible for handling user requests and updating the model and view accordingly.

Q9.What are some of the advantages of Servlets?

ans)Servlets have many advantages over other server-side technologies, such as CGI scripts and ASP pages. Some of the key advantages of servlets include:

1. Performance: Servlets are more efficient than CGI scripts because they do not create a new process for each request. This can improve performance significantly, especially for high-traffic web applications.
2. Portability: Servlets are platform-independent, meaning that they can be run on any web server that supports the Servlet API. This makes it easy to deploy servlet-based web applications.
3. Scalability: Servlets are scalable and can be easily extended to accommodate new features or changes to the application. This makes them a good choice for large and complex web applications.
4. Security: Servlets are secure because they are written in Java, which is a compiled language. This means that security vulnerabilities are less likely to occur in servlets than in scripting languages, such as PHP or ASP.
5. Reusability: Servlets can be reused in different web applications, which can save time and development effort.
6. Extensibility: Servlets can be extended to provide new functionality or to integrate with other web services.

Q10.What are the limitations of JSP?

ans)Some of the limitations of JSP:

1. Performance: JSP pages are compiled into Java servlets on the server, which can add some overhead. This can be a problem for high-traffic web applications.

2. Complexity: JSP pages can be complex, especially when they involve a lot of logic or interaction with the database. This can make them difficult to develop and maintain

3. Security: JSP pages can be vulnerable to security attacks, such as cross-site scripting (XSS) and SQL injection. This is because JSP pages are executed on the server, which means that they have access to the underlying code and data.

4. Lack of flexibility: JSP pages are not as flexible as other server-side technologies, such as PHP or ASP. This is because JSP pages are based on the Java programming language, which can be more complex than other scripting languages.