BÁO CÁO ĐẦY ĐỦ LẬP TRÌNH WEB

NHÓM 01: Ngọ Văn Trọng
Trương Vĩnh Tiến
Nguyễn Như Thiệu
Nguyễn Anh Huân

Cách thức tạo một trang web application với java (servlet + jsp)

Phần 1 : Tổng quan về Servlet và JSP.

1.Khái niệm, kiến trúc và cách hoạt động Servlet.

Java Servlet là chương trình chạy trên một Web Server . Nó hoạt động như một lớp trung gian giữa một yêu cầu đến từ một trình duyệt Web hoặc HTTP khách (Client) khác và cơ sở dữ liệu hoặc các ứng dụng trên máy chủ HTTP (HTTP Server).

Giúp tạo web động , có nhiều interface và các lớp trong API servlet như , HttpServlet, Servlet Request, ServletResponse, …

Servlets được xây dựng từ hai gói:

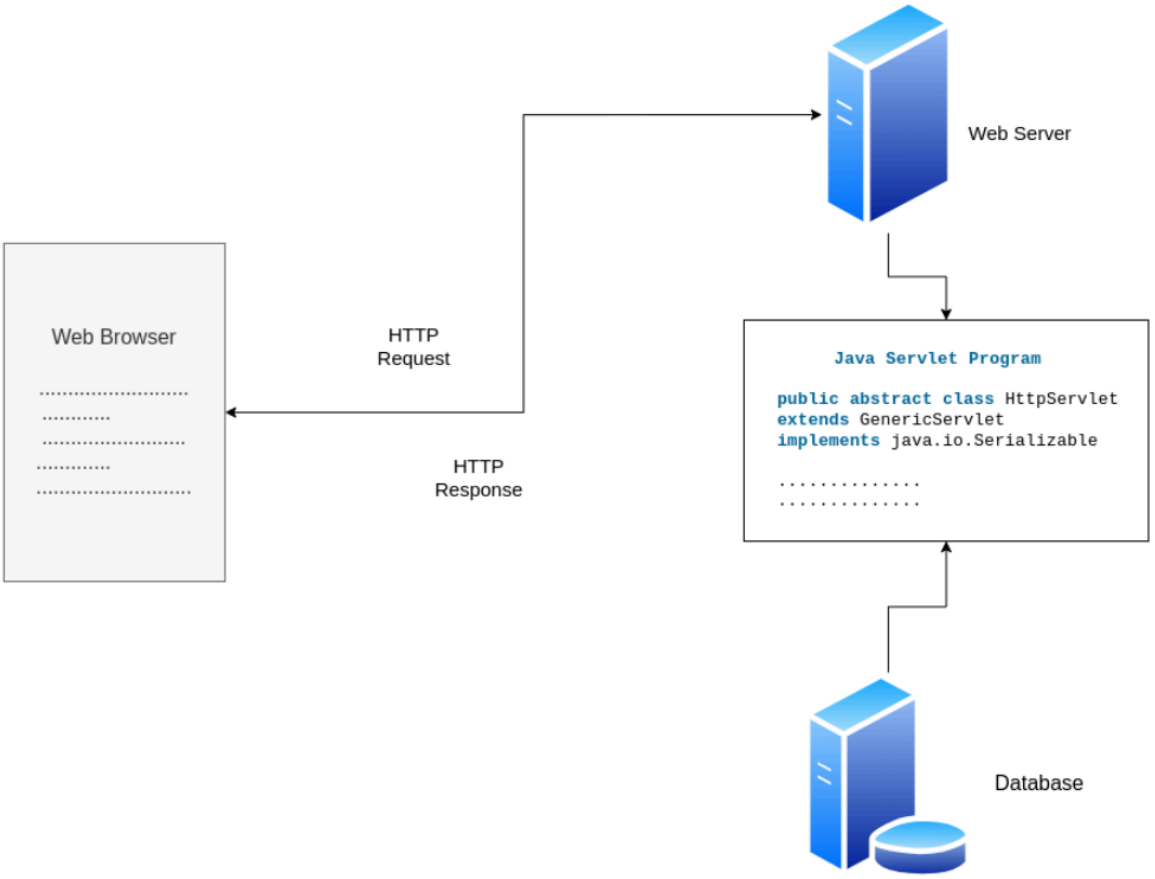
javax.servlet(Cơ bản)
javax.servlet.http(Nâng cao)

Gói javax.servlet cung cấp các cơ sở cơ bản và nền tảng cho việc xử lý yêu cầu không phụ thuộc vào giao thức, trong khi javax.servlet.http cung cấp các

chức năng cụ thể cho giao thức HTTP, làm cho việc phát triển các ứng dụng web hiệu quả và linh hoạt hơn.

Component	Type	Package
Servlet	Interface	javax.servlet.*
ServletRequest	Interface	javax.servlet.*
ServletResponse	Interface	javax.servlet.*
GenericServlet	Class	javax.servlet.*
HttpServlet	Class	javax.servlet.http.*
HttpServletRequest	Interface	javax.servlet.http.*
HttpServletResponse	Interface	javax.servlet.http.*
Filter	Interface	javax.servlet.*
ServletConfig	Interface	javax.servlet.*

ứng với các lớp sẽ có rất nhiều phương thức , điều này sẽ được giới thiệu ở phần demo



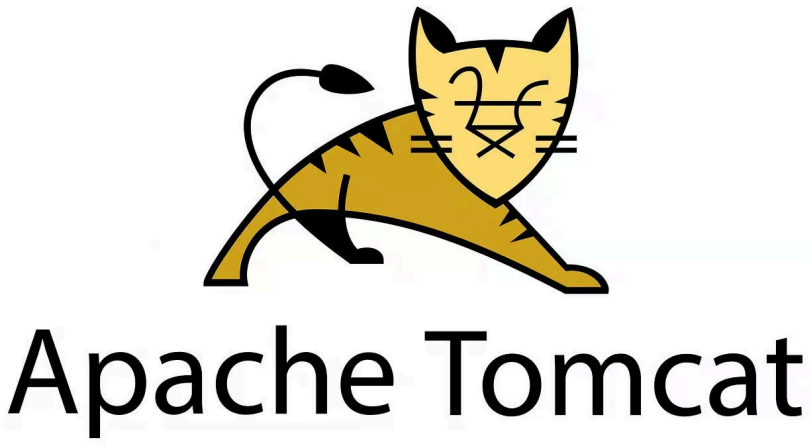
Về cơ bản, việc thực thi Servlets bao gồm Sáu bước cơ bản:

1. Khách hàng gửi yêu cầu đến Máy chủ Web.
2. Máy chủ Web nhận được yêu cầu.
3. Máy chủ Web chuyển yêu cầu đến servlet tương ứng.
4. Servlet xử lý yêu cầu và tạo phản hồi ở dạng đầu ra.
5. Servlet gửi phản hồi trở lại máy chủ web.
6. Máy chủ Web gửi phản hồi lại cho máy khách và trình duyệt máy khách hiển thị nó trên màn hình.

2. Servlet Container.

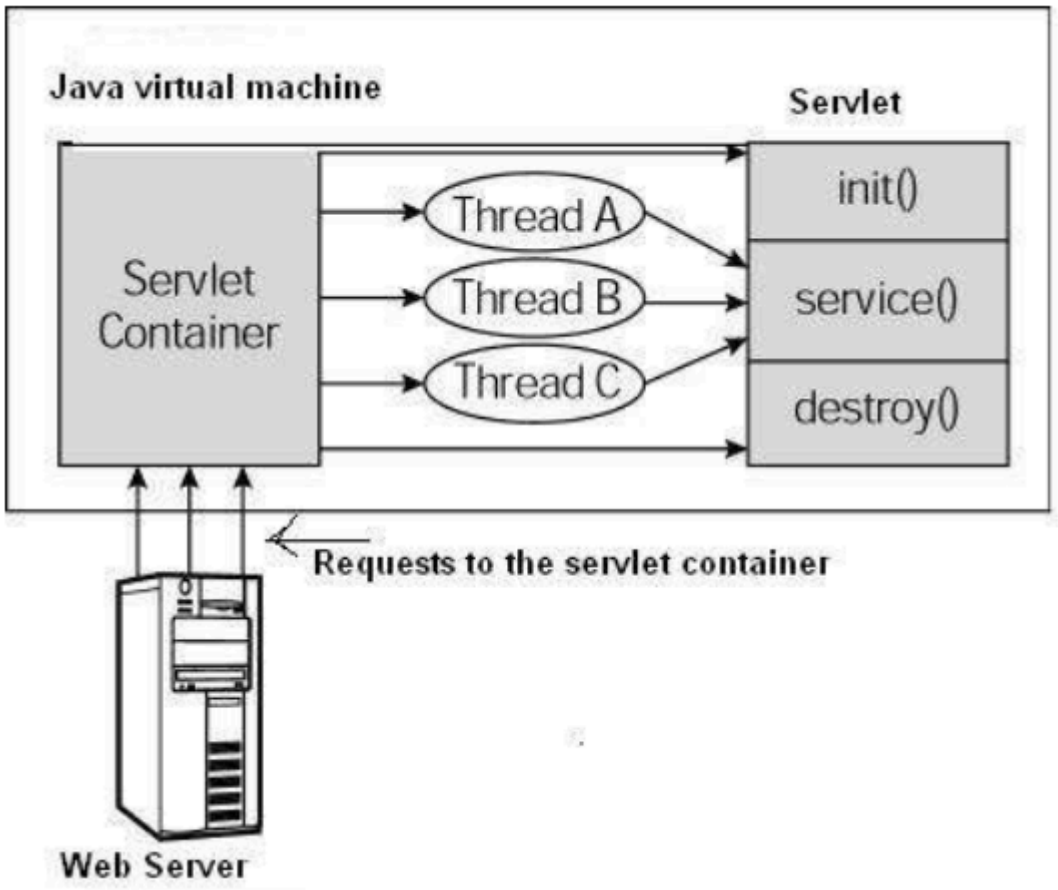
a.Khái niệm

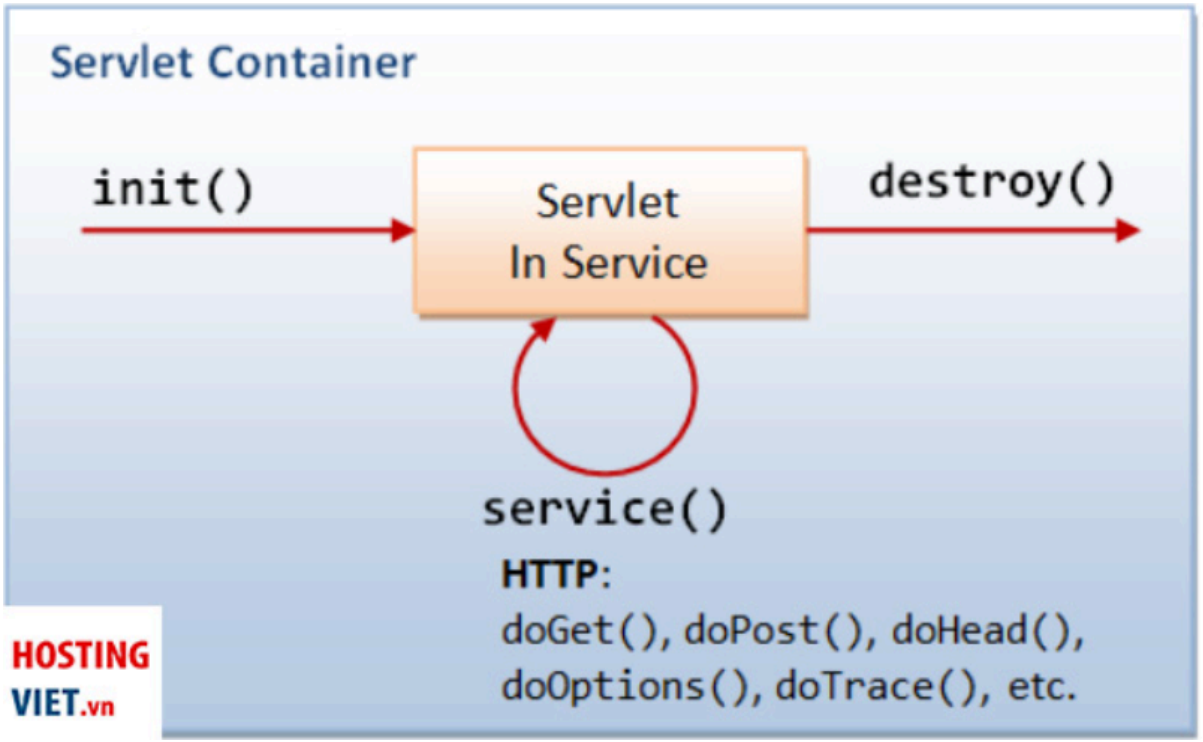
Servlet container là một phần của server, đóng vai trò cung cấp môi trường để chạy servlet , quản lý các hoạt động liên quan đến vòng đời và xử lý yêu cầu của servlet
Một số Servlet container như Tomcat server , Glassfish Server



.

b.Vòng đời của servlet :





Khởi Tạo (Init)

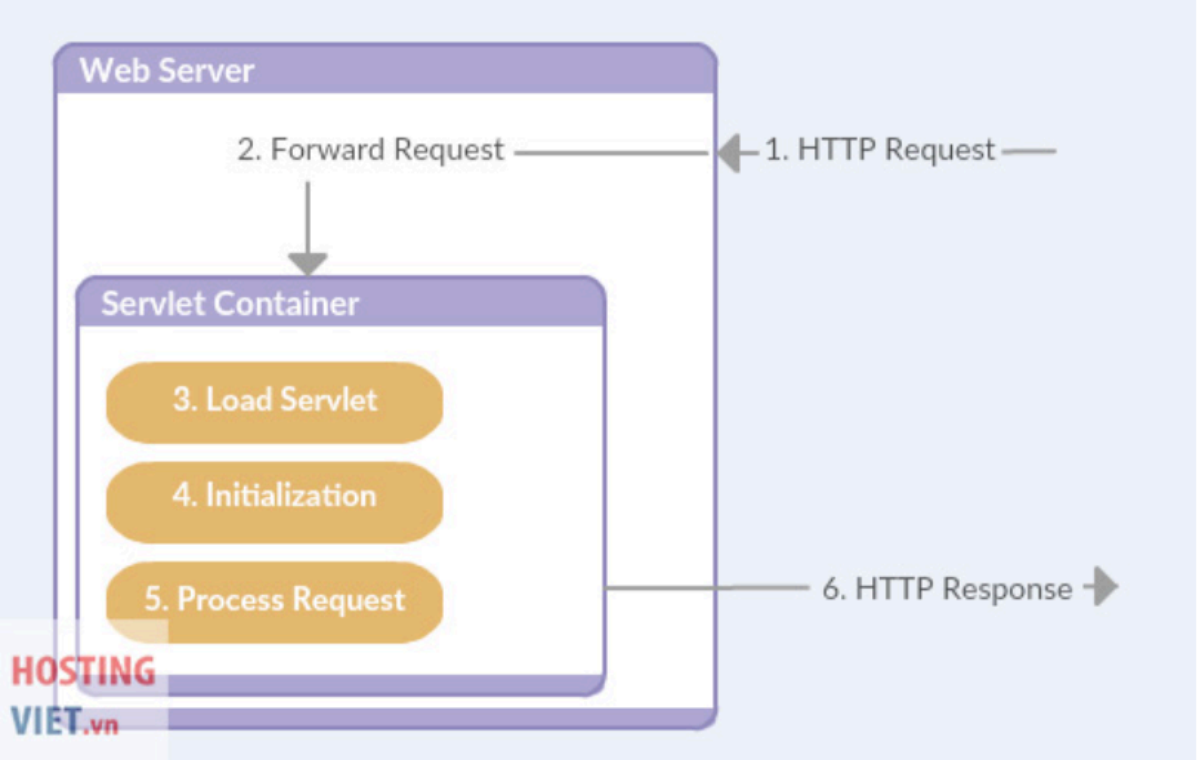
- Phương thức **init()**: Khi servlet được chạy lần đầu, servlet container sẽ gọi phương thức init() của servlet. Đây là nơi để thực hiện tất cả các cấu hình ban đầu cần thiết cho servlet, như khởi tạo tài nguyên, thiết lập kết nối cơ sở dữ liệu, đọc các tham số cấu hình, v.v. Phương thức init() chỉ được gọi một lần.

Xử lý Yêu cầu (Service)

- Phương thức **service()**: Sau khi servlet đã được khởi tạo. Mỗi lần nhận được một yêu cầu HTTP, servlet container sẽ gọi phương thức service() của servlet. Phương thức này sẽ kiểm tra loại yêu cầu HTTP (GET, POST, PUT, DELETE, v.v.) và chuyển tiếp yêu cầu đến phương thức thích hợp (như doGet(), doPost(), v.v.). Phương thức service() là nơi điều phối chính cho việc xử lý các yêu cầu.

Hủy bỏ (Destroy)

- Phương thức destroy(): Khi servlet không còn được cần đến nữa, hoặc khi servlet container đang được tắt, phương thức destroy() sẽ được gọi. Đây là cơ hội để servlet làm sạch và giải phóng các tài nguyên đã được sử dụng, như đóng các kết nối cơ sở dữ liệu, dọn dẹp các bộ nhớ tạm, v.v. Giống như init(), phương thức destroy() cũng chỉ được gọi một lần.



Từ đó suy ra các bước đầy đủ để xử lý yêu cầu request:

- Bước 1: Đầu tiên, máy chủ Web sẽ thực hiện nhận HTTP request.
- Bước 2: Web server chuyển tiếp yêu cầu đã nhận đến Servlet Container.
- Bước 3: Servlet tự động tiến hành lấy yêu cầu rồi tải chúng lên địa chỉ không gian Container (áp dụng trong trường hợp nó thuộc Container).
- Bước 4: Container thực hiện lệnh gọi init () method của Servlet (chỉ gọi một lần khi Servlet tải lên lần đầu) để khởi tạo.
- Bước 5: Container tiến hành gọi service () method của Servlet nhằm mục đích xử lý HTTP request. Điều này có nghĩa, chúng thực hiện việc đọc toàn bộ dữ liệu có trong yêu cầu, sau đó hình thành một response.
- Bước 6: Cuối cùng, máy chủ Web trả lại kết quả động tương ứng với vị trí yêu cầu.
- Bước 7: Nếu hủy servlet container , phương thức destroy() sẽ được gọi một lần duy nhất

3. JSP

a. Khái niệm

```
 */
@WebServlet(name = "NewServlet", urlPatterns = {"/NewServlet"})
public class NewServlet extends HttpServlet {

    /**
     * Processes requests for both HTTP <code>GET</code> and <code>POST</code>
     * methods.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    protected void processRequest(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        try (PrintWriter out = response.getWriter()) {
            /* TODO output your page here. You may use following sample code. */
            out.println("<!DOCTYPE html>");
            out.println("<html>");
            out.println("<head>");
            out.println("<title>Servlet NewServlet</title>");
            out.println("</head>");
            out.println("<body>");
            out.println("<h1>Servlet NewServlet at " + request.getContextPath() + "</h1>");
            out.println("</body>");
            out.println("</html>");
        }
    }

    HttpServlet methods. Click on the + sign on the left to edit the code.

}
```

Trên đấy là một New Servlet được tạo ra , ta có thể thấy servlet nó sẽ bao gồm code logic và code giao diện , nhưng do quá trình code giao diện trên servlet rất khó , từ đó jsp ra đời để thay thế cho việc design.

JSP hay còn gọi là Java Server Page, là công nghệ phía server để tạo web page bằng java.Nó chứa mã HTML và Java xen kẽ để tạo ra giao diện cách thức hiển thị dữ liệu.

Bản chất nó chính là phần mở rộng của servlet chưa cả code code logic và code giao diện được tách biệt .Trong JSP, chúng ta định nghĩa thẻ bằng "<% %>".

- Scriptlet: Được định nghĩa bằng <% %> - cho phép chèn mã Java trực tiếp vào HTML.
- Expression: Được định nghĩa bằng <%= %> - được dùng để in giá trị biểu thức Java vào HTML.

- Directive: Được định nghĩa bằng <%@ %> - cung cấp thông tin chỉ đạo cho JSP engine (ví dụ, thư viện thẻ).

JSP là một giao diện nằm trên đầu Servlets.
Theo cách khác, chúng ta có thể nói rằng JSPs là mở rộng của servlet để giảm thiểu nỗ lực của các nhà phát triển để viết giao diện người dùng bằng cách sử dụng lập trình Java.

4. Sự kết hợp giữa Servlet và Jsp , mô hình MVC

Cách thức hoạt động

1. Nhận Request từ Trình Duyệt: Khi một người dùng nhấp vào liên kết trên trình duyệt , yêu cầu sẽ được gửi về địa chỉ chứa servlet và gọi tên servlet để xử lý.

2. Xử Lý Logic trong Servlet: Servlet được gọi bởi servlet container (như Tomcat hoặc GlassFish). Tại đây, servlet xử lý các nghiệp vụ cần thiết, có thể bao gồm:
    - Truy xuất hoặc cập nhật dữ liệu từ cơ sở dữ liệu thông qua DAO (Data Access Object).
    - Thực hiện các phép tính hoặc xử lý logic khác.
    - Chuẩn bị dữ liệu cần thiết để hiển thị.

set thuộc tính và đối tượng cần thiết cho request , response

3. Sử Dụng **getRequestDispatcher** để điều hướng yêu cầu từ trình duyệt sang JSP.

4. Forward Request và Response:
    - Servlet sau đó sử dụng phương thức **forward()** của **RequestDispatcher** để chuyển tiếp cả đối tượng HttpServletRequest và HttpServletResponse đã được set các thuộc tính cần thiết sang trang JSP.

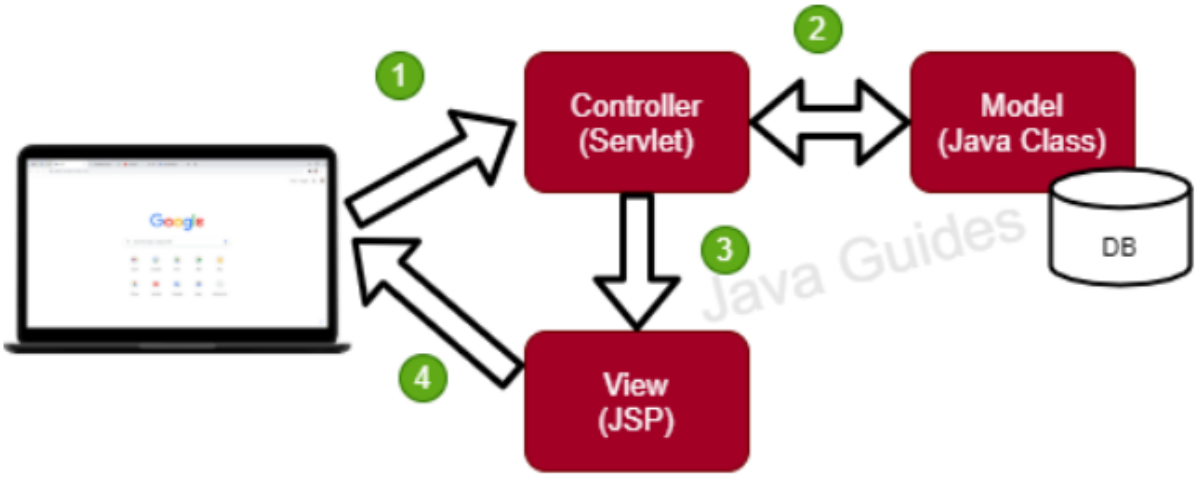
5. JSP Sinh Ra Nội Dung Động:
    - Trang JSP sử dụng các thuộc tính được set trong đối tượng request để sinh ra nội dung động, thường là dưới dạng HTML. JSP có thể sử dụng các tag library , java để hiển thị nội dung động cho người dùng.

6. Gửi Phản Hồi Lại Cho Trình Duyệt:

- Sau khi JSP hoàn tất việc sinh nội dung, HTML được trả lại cho trình duyệt thông qua phản hồi HTTP, và người dùng nhìn thấy trang web được cập nhật.

Quá trình này là ví dụ về mô hình MVC (Model-View-Controller) trong đó:

- Model: Dữ liệu và các phương thức tương tác dữ liệu. (Database , java)
- View: Hiển thị nội dung (JSP).
- Controller: Điều khiển response/request và xử lý logic (servlet).

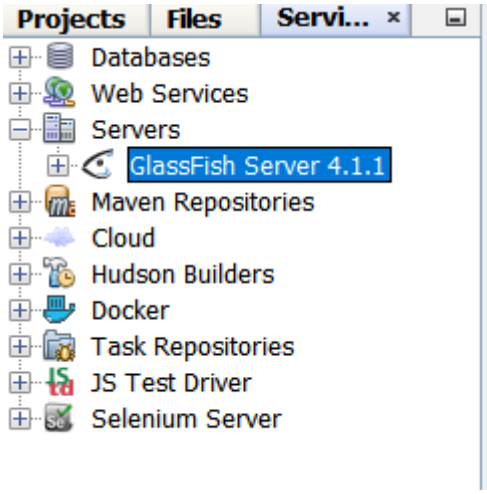


PROJECT VÍ DỤ

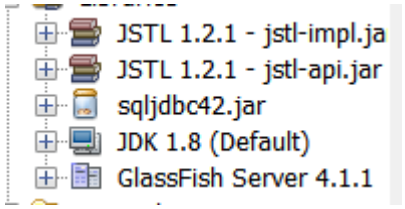
TẠO MODEL LẤY DỮ LIỆU TỪ DB

```
public List<Product> getAllProduct(){
    List<Product> list = new ArrayList<>();
    String query = "select * from product";
    try{
        conn= new DBContext().getConnection();
        ps = conn.prepareStatement(query);
        rs = ps.executeQuery();
        while(rs.next()){
            list.add(new Product(rs.getInt(1),
            rs.getString(2),
            rs.getString(3),
            rs.getDouble(4),
            rs.getString(5),
            rs.getString(6),
            rs.getString(7),
            rs.getString(8),
            rs.getString(9)));
        }
    }catch(Exception e){

    }
    return list;

}
```

Tạo Server để quản lý Servlet



Add Library để sử dụng



Tạo Servlet để trung chuyển dữ liệu , nhận request từ trình duyệt

```
 */
protected void processRequest(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
    response.setContentType("text/html;charset=UTF-8");
    DAO dao = new DAO();
    List<Product> list = dao.getAllProduct();
    request.setAttribute("listP", list);
    request.getRequestDispatcher("Home.jsp").forward(request, response);
}

// <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the + sign on th
```

Tạp JSP để hiển thị dữ liệu và đẩy response lên trình duyệt.

```
<c:forEach items="${listP}" var="o">
    <div class="product col-12 col-md-6 col-lg-4">
        <div class="card">
            <img class="card-img-top" src="${o.image}" alt="Card image cap">
            <div class="card-body">
                <h4 class="card-title show_txt"><a href="detail?pid=${o.id}" title="View Product">${o.name}</a></h4>
                <p class="card-text show_txt">${o.title}</p>
                <div class="row">
                    <div class="col">
                        <p class="btn btn-danger btn-block">${o.price}/tháng VND</p>
                    </div>

                    <div class="col">
                        <a href="detail?pid=${o.id}" class="btn btn-success btn-block">View Room</a>

                    </div>
                </div>
            </div>
        </div>
    </div>
</c:forEach>
```