**1.Create SampleServlet extend the SlingAllMethod Servlet and register it using resourceType**

package com.myTraining.core.servlets;  
  
import org.apache.sling.api.SlingHttpServletRequest;  
import org.apache.sling.api.SlingHttpServletResponse;  
import org.apache.sling.api.servlets.SlingAllMethodsServlet;  
import org.osgi.service.component.annotations.Component;  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
  
import javax.servlet.Servlet;  
import javax.servlet.ServletException;  
import java.io.IOException;  
  
@Component(  
 service = Servlet.class,  
 property = {  
 "sling.servlet.resourceTypes=myTraining/components/sample",  
 "sling.servlet.methods=GET"  
 }  
)  
public class SampleServlet extends SlingAllMethodsServlet {  
  
 private static final Logger *LOG* = LoggerFactory.*getLogger*(SampleServlet.class);  
  
 @Override  
 protected void doGet(SlingHttpServletRequest request, SlingHttpServletResponse response)  
 throws ServletException, IOException {  
 *LOG*.info("✅ SampleServlet Executed!");  
 response.getWriter().write("Sample Servlet Response!");  
 }  
}

**2.Create CreatePageServlet extend the SlingSafeMethod Servlet and register it using path**

package com.myTraining.core.servlets;  
  
import com.day.cq.wcm.api.Page;  
import com.day.cq.wcm.api.PageManager;  
import org.apache.sling.api.SlingHttpServletRequest;  
import org.apache.sling.api.SlingHttpServletResponse;  
import org.apache.sling.api.servlets.SlingSafeMethodsServlet;  
import org.osgi.service.component.annotations.Component;  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
  
import javax.servlet.Servlet;  
import javax.servlet.ServletException;  
import java.io.IOException;  
  
@Component(  
 service = Servlet.class,  
 property = {  
 "sling.servlet.paths=/bin/createPage",  
 "sling.servlet.methods=POST"  
 }  
)  
public class CreatePageServlet extends SlingSafeMethodsServlet {  
  
 private static final Logger *LOG* = LoggerFactory.*getLogger*(CreatePageServlet.class);  
 private static final String *PAGE\_PATH* = "/content/us/en/";  
  
 @Override  
 protected void doPost(SlingHttpServletRequest request, SlingHttpServletResponse response)  
 throws ServletException, IOException {  
  
 String pageName = request.getParameter("pageName");  
 if (pageName == null || pageName.isEmpty()) {  
 response.getWriter().write("❌ Page name is required!");  
 return;  
 }  
  
 PageManager pageManager = request.getResourceResolver().adaptTo(PageManager.class);  
 if (pageManager != null) {  
 Page newPage = pageManager.create(*PAGE\_PATH*, pageName, "/conf/wknd/settings/wcm/templates/content-page", pageName);  
 if (newPage != null) {  
 *LOG*.info("✅ Page Created: {}", newPage.getPath());  
 response.getWriter().write("✅ Page Created at: " + newPage.getPath());  
 } else {  
 response.getWriter().write("❌ Page creation failed!");  
 }  
 }  
 }  
}

**3.Take page name from user and create pages in aem using above servlet**

package com.myTraining.core.servlets;  
  
import com.day.cq.wcm.api.Page;  
import com.day.cq.wcm.api.PageManager;  
import com.day.cq.wcm.api.WCMException;  
import org.apache.sling.api.SlingHttpServletRequest;  
import org.apache.sling.api.SlingHttpServletResponse;  
import org.apache.sling.api.servlets.SlingAllMethodsServlet;  
import org.osgi.service.component.annotations.Component;  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
  
import javax.servlet.Servlet;  
import javax.servlet.ServletException;  
import java.io.IOException;  
  
@Component(  
 service = Servlet.class,  
 property = {  
 "sling.servlet.paths=/bin/createPage",  
 "sling.servlet.methods=POST"  
 }  
)  
public class CreatePageServlet extends SlingAllMethodsServlet {  
  
 private static final Logger *LOG* = LoggerFactory.*getLogger*(CreatePageServlet.class);  
 private static final String *PARENT\_PATH* = "/content/us/en";  
 private static final String *TEMPLATE\_PATH* = "/conf/wknd/settings/wcm/templates/content-page";  
  
 @Override  
 protected void doPost(SlingHttpServletRequest request, SlingHttpServletResponse response)  
 throws ServletException, IOException {  
  
 String pageName = request.getParameter("pageName");  
 if (pageName == null || pageName.trim().isEmpty()) {  
 response.setStatus(SlingHttpServletResponse.*SC\_BAD\_REQUEST*);  
 response.getWriter().write("❌ Error: Page name is required!");  
 return;  
 }  
  
 try {  
 PageManager pageManager = request.getResourceResolver().adaptTo(PageManager.class);  
 if (pageManager != null) {  
 Page newPage = pageManager.create(*PARENT\_PATH*, pageName, *TEMPLATE\_PATH*, pageName);  
 if (newPage != null) {  
 *LOG*.info("✅ Page Created Successfully: {}", newPage.getPath());  
 response.getWriter().write("✅ Page Created at: " + newPage.getPath());  
 } else {  
 response.getWriter().write("❌ Error: Page creation failed!");  
 }  
 } else {  
 response.getWriter().write("❌ Error: PageManager is null!");  
 }  
 } catch (WCMException e) {  
 *LOG*.error("❌ Page creation error: {}", e.getMessage());  
 response.getWriter().write("❌ Error: " + e.getMessage());  
 }  
 }  
}

**4.Create one SearchServlet to search the content using PredicateMap to search the content from pages**

package com.myTraining.core.servlets;  
  
import com.day.cq.search.Query;  
import com.day.cq.search.QueryBuilder;  
import com.day.cq.search.Result;  
import com.day.cq.search.predicates.PredicateGroup;  
import org.apache.sling.api.SlingHttpServletRequest;  
import org.apache.sling.api.SlingHttpServletResponse;  
import org.apache.sling.api.servlets.SlingAllMethodsServlet;  
import org.apache.sling.api.resource.ResourceResolver;  
import org.osgi.service.component.annotations.Component;  
import org.osgi.service.component.annotations.Reference;  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
  
import javax.servlet.Servlet;  
import javax.servlet.ServletException;  
import java.io.IOException;  
import java.util.HashMap;  
import java.util.Map;  
  
@Component(  
 service = Servlet.class,  
 property = {  
 "sling.servlet.paths=/bin/searchContent",  
 "sling.servlet.methods=GET"  
 }  
)  
public class SearchServlet extends SlingAllMethodsServlet {  
  
 private static final Logger *LOG* = LoggerFactory.*getLogger*(SearchServlet.class);  
  
 @Reference  
 private QueryBuilder queryBuilder;  
  
 @Override  
 protected void doGet(SlingHttpServletRequest request, SlingHttpServletResponse response)  
 throws ServletException, IOException {  
  
 String searchTerm = request.getParameter("q");  
 if (searchTerm == null || searchTerm.trim().isEmpty()) {  
 response.setStatus(SlingHttpServletResponse.*SC\_BAD\_REQUEST*);  
 response.getWriter().write("❌ Error: Search term is required!");  
 return;  
 }  
  
 // Get Resource Resolver  
 ResourceResolver resourceResolver = request.getResourceResolver();  
  
 // Create PredicateMap for QueryBuilder API  
 Map<String, String> predicateMap = new HashMap<>();  
 predicateMap.put("path", "/content/us/en"); // Searching in this path  
 predicateMap.put("type", "cq:Page"); // Only search pages  
 predicateMap.put("fulltext", searchTerm); // Search content  
 predicateMap.put("p.limit", "10"); // Limit results to 10  
  
 try {  
 Query query = queryBuilder.createQuery(PredicateGroup.create(predicateMap), resourceResolver.adaptTo(Session.class));  
 Result result = query.getResult();  
  
 StringBuilder responseText = new StringBuilder();  
 result.getHits().forEach(hit -> {  
 try {  
 responseText.append("🔹 Found: ").append(hit.getPath()).append("\n");  
 } catch (Exception e) {  
 *LOG*.error("Error fetching search results: {}", e.getMessage());  
 }  
 });  
  
 if (responseText.length() == 0) {  
 responseText.append("⚠️ No results found for: ").append(searchTerm);  
 }  
  
 response.getWriter().write(responseText.toString());  
  
 } catch (Exception e) {  
 *LOG*.error("❌ Error executing search: {}", e.getMessage());  
 response.getWriter().write("❌ Error: " + e.getMessage());  
 }  
 }  
}