26/03/2025

Tasks

1. Create SampleServlet using SlingAllMethodServlet
2. Create CreatePageServlet using SlingSafeMethodServlet
3. Create AEM pages dynamically using servlet
4. Use Page Manager APIs for page creation
5. Create SearchServlet using PredicateMap for content search

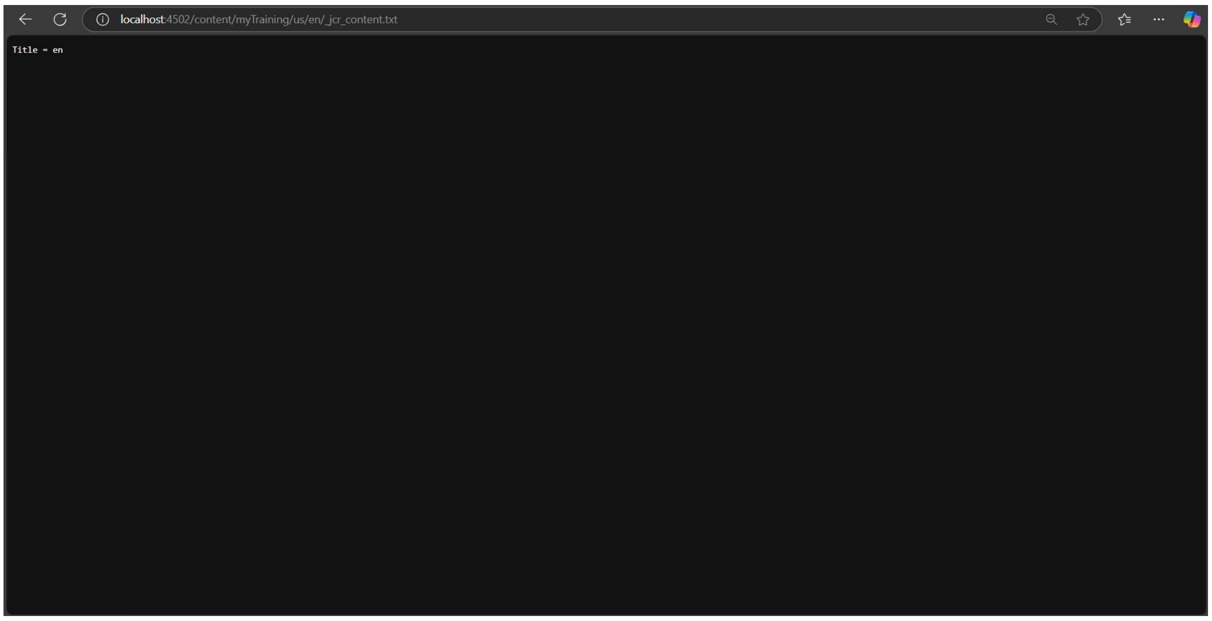
1. Create SampleServlet using SlingAllMethodServlet

Steps:

1. **Create Java Class:**
   - Extend SlingAllMethodsServlet to support both GET and POST requests.
2. **Register Servlet using Resource Type:**
   - Use @SlingServletResourceTypes annotation.
   - Define resourceTypes and methods properties.
3. **Override doGet and doPost Methods:**
   - Implement logic for handling GET and POST requests.
   - Print sample responses to verify execution.
4. **Deploy and Test Servlet:**
   - Register servlet in AEM.
   - Access the servlet via the defined resource type.

Code:

```
package com.myTraining.core.servlets;

import com.day.cq.commons.jcr.JcrConstants;
import org.apache.sling.api.SlingHttpServletRequest;
import org.apache.sling.api.SlingHttpServletResponse;
import org.apache.sling.api.resource.Resource;
import org.apache.sling.api.servlets.HttpConstants;
import org.apache.sling.api.servlets.SlingAllMethodsServlet;
import org.apache.sling.api.servlets.SlingSafeMethodsServlet;
import org.apache.sling.servlets.annotations.SlingServletResourceTypes;
import org.osgi.service.component.annotations.Component;
import org.osgi.service.component.propertytypes.ServiceDescription;

import javax.servlet.Servlet;
import javax.servlet.ServletException;
import java.io.IOException;

/**
 * Servlet that writes some sample content into the response. It is mounted for
 * all resources of a specific Sling resource type. The
 * {@link SlingSafeMethodsServlet} shall be used for HTTP methods that are
 * idempotent. For write operations use the {@link SlingAllMethodsServlet}.
 */
@Component(service = { Servlet.class })
@SlingServletResourceTypes(
        resourceTypes="myTraining/components/page",
        methods=HttpConstants.METHOD_GET,
        extensions="txt")
@ServiceDescription("Simple Demo Servlet")
public class SimpleServlet extends SlingAllMethodsServlet {

    private static final long serialVersionUID = 1L;

    @Override
    protected void doGet(final SlingHttpServletRequest req,
                    final SlingHttpServletResponse resp) throws ServletException, IOException {
        final Resource resource = req.getResource();
        resp.setContentType("text/plain");
        resp.getWriter().write("Title = " +
resource.getValueMap().get(JcrConstants.JCR_TITLE));
    }
}
```



```
Title = en
```

2. Create CreatePageServlet using SlingSafeMethodServlet

Steps:

1. **Create Java Class:**
    - Extend SlingSafeMethodsServlet to handle safe HTTP methods.
2. **Register Servlet using Path:**
    - Use @SlingServletPaths annotation.
    - Define paths and methods properties.
3. **Implement Logic to Create Page:**
    - Retrieve pageName parameter from request.
    - Use PageManager to create a new page under /content.
    - Use an existing template (/conf/template).
4. **Deploy and Verify Servlet:**
    - Access servlet via /bin/createpage?pageName=samplePage.
    - Check if the page is created in AEM under /content/samplePage.

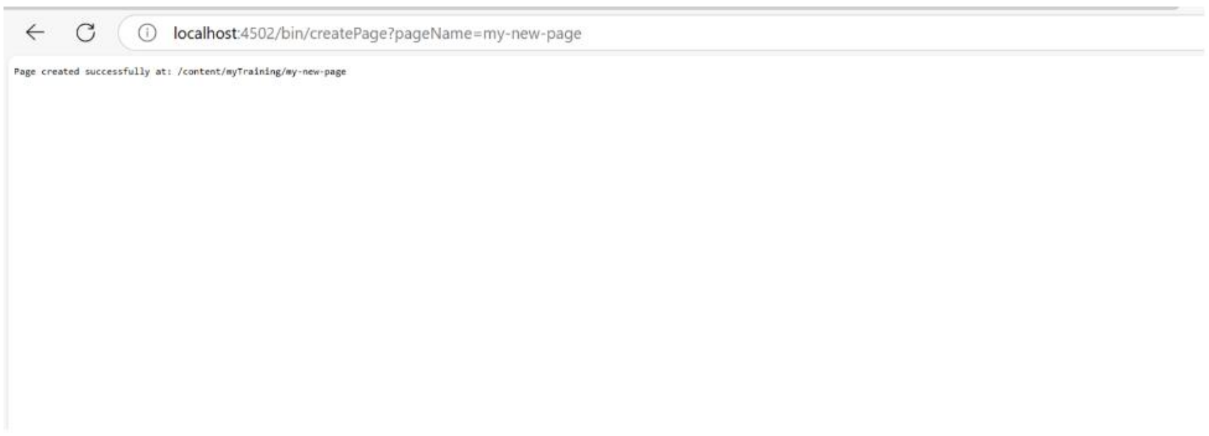
Code:

```
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.SlingSafeMethodsServlet;
import org.apache.sling.models.annotations.DefaultInjectionStrategy;
import org.apache.sling.models.annotations.Model;
import org.osgi.service.component.annotations.Component;
import org.osgi.service.component.annotations.Reference;

import javax.servlet.Servlet;
import javax.servlet.ServletException;
import java.io.IOException;

import static
org.apache.sling.api.servlets.ServletResolverConstants.SLING_SERVLET_PATHS;
import static
org.apache.sling.api.servlets.ServletResolverConstants.SLING_SERVLET_METHODS;

@Component(
    service = Servlet.class,
    property = {
        SLING_SERVLET_PATHS + "=/bin/createPage",
        SLING_SERVLET_METHODS + "=GET"
    }
)
public class CreatePageServlet extends SlingSafeMethodsServlet {

  private static final String PARENT_PATH = "/content/myTraining"; // Set your parent path
  private static final String TEMPLATE_PATH =
"/conf/myTraining/settings/wcm/templates/page-content"; // Set your template path
```

```
    @Reference
    private org.apache.sling.api.resource.ResourceResolverFactory resourceResolverFactory;

    @Override
    protected void doGet(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;
        }

        org.apache.sling.api.resource.ResourceResolver resolver =
request.getResourceResolver();
        PageManager pageManager = resolver.adaptTo(PageManager.class);

        if (pageManager != null) {
            try {
                Page newPage = pageManager.create(PARENT_PATH, pageName,
TEMPLATE_PATH, pageName);
                response.getWriter().write("Page created successfully at: " + newPage.getPath());
            } catch (WCMException e) {
                response.getWriter().write("Error creating page: " + e.getMessage());
            }
        } else {
            response.getWriter().write("PageManager is null. Unable to create page.");
        }
    }
```

}

Output


localhost:4502/bin/createPage?pageName=my-new-page

```
Page created successfully at: /content/myTraining/my-new-page
```

3. Create AEM pages dynamically using servlet

Steps:

1. **Accept User Input:**
    - Pass pageName as a request parameter.
2. **Invoke CreatePageServlet:**
    - Call /bin/createpage?pageName=dynamicPage.
3. **Verify Page Creation:**
    - Navigate to /content/dynamicPage in AEM’s Site Admin.

4. Use Page Manager APIs for page creation

Best Practices:

- **Ensure Service User Permissions:**
    - Use system users with proper permissions.
- **Use Templates & Components:**
    - Assign pages a meaningful template.
- **Handle Naming Conflicts:**
    - Verify if the page already exists before creating it

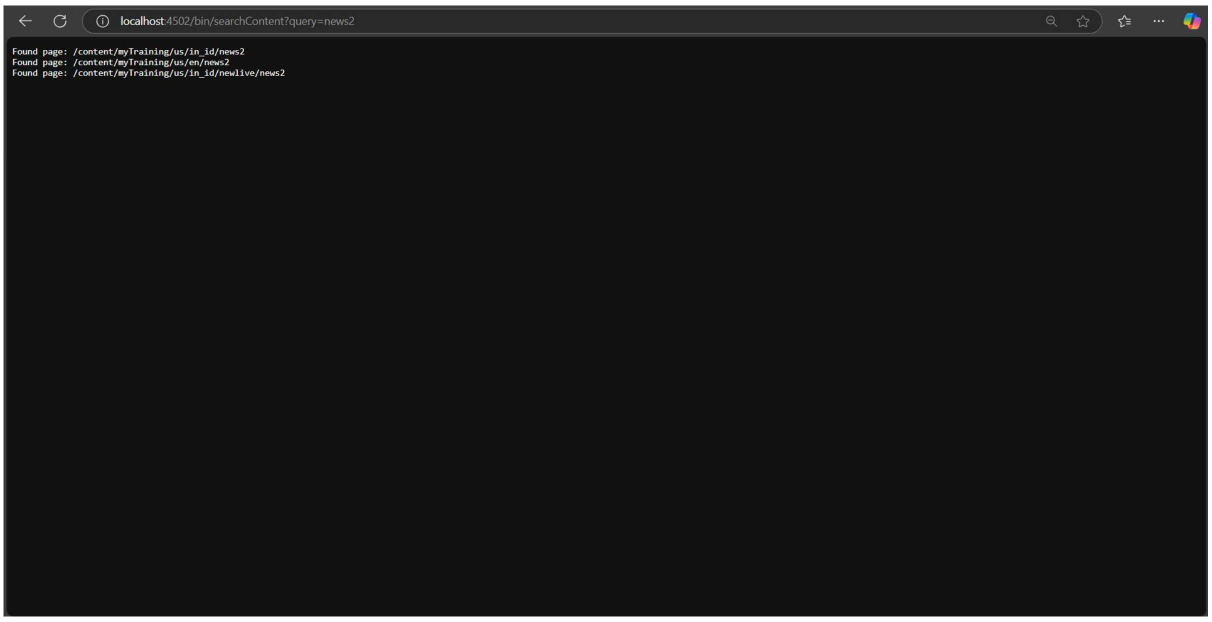
5. Create SearchServlet using PredicateMap for content search

Steps:

1. **Create Java Class:**
   - Extend SlingSafeMethodsServlet to ensure secure operations.
2. **Register Servlet using Path:**
   - Use @SlingServletPaths annotation to define endpoint /bin/searchcontent.
3. **Use Query Builder API:**
   - Construct a PredicateMap to search content.
   - Set path, type, and fulltext search parameters.
4. **Process Search Results:**
   - Retrieve results from QueryBuilder.
   - Return formatted search results.

   Code:

```
package com.myTraining.core.servlets;

import com.day.cq.search.Query;
import com.day.cq.search.QueryBuilder;
import com.day.cq.search.result.Hit;
import com.day.cq.search.result.SearchResult;
import com.day.cq.search.PredicateGroup;
import org.apache.sling.api.servlets.SlingSafeMethodsServlet;
import org.apache.sling.api.SlingHttpServletRequest;
import org.apache.sling.api.SlingHttpServletResponse;
import org.apache.sling.api.resource.ResourceResolver;
import org.osgi.service.component.annotations.Component;
import org.osgi.service.component.annotations.Reference;
import org.osgi.framework.Constants;

import javax.jcr.Session;
import javax.servlet.Servlet;
import javax.servlet.ServletException;
import java.io.IOException;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

@Component(
        service = {Servlet.class},
        property = {
                Constants.SERVICE_DESCRIPTION + "= Search Content Servlet",
                "sling.servlet.paths=/bin/searchContent",
                "sling.servlet.methods=GET"
        }
)
public class SearchServlet extends SlingSafeMethodsServlet {

    @Reference
    private QueryBuilder queryBuilder; // Inject QueryBuilder service

    @Override
    protected void doGet(SlingHttpServletRequest request, SlingHttpServletResponse
response) throws ServletException, IOException {
        response.setContentType("text/plain");
        String searchText = request.getParameter("query");

        if (searchText == null || searchText.isEmpty()) {
            response.getWriter().write("No search query provided");
            return;
        }

        ResourceResolver resourceResolver = request.getResourceResolver();
        Session session = resourceResolver.adaptTo(Session.class); // Get JCR Session

        if (session == null) {
            response.getWriter().write("JCR Session is null");
            return;
        }

        if (queryBuilder == null) {
            response.getWriter().write("QueryBuilder service is unavailable");
            return;
        }

        // Build search predicates
        Map<String, String> predicates = new HashMap<>();
        predicates.put("path", "/content/myTraining");  // Change this to your content root
        predicates.put("type", "cq:Page");
        predicates.put("fulltext", searchText);
        predicates.put("p.limit", "10"); // Limit search results
        try {
            Query query = queryBuilder.createQuery(PredicateGroup.create(predicates), session);
            SearchResult result = query.getResult();

            List<Hit> hits = result.getHits();
            if (hits.isEmpty()) {
                response.getWriter().write("No results found for: " + searchText);
            } else {
                for (Hit hit : hits) {
                    response.getWriter().write("Found page: " + hit.getPath() + "\n");
                }
            }
        } catch (Exception e) {
            response.getWriter().write("Error executing search query: " + e.getMessage());
        }
    }
}
```



```
Found page: /content/myTraining/us/in_id/news2
Found page: /content/myTraining/us/en/news2
Found page: /content/myTraining/us/in_id/newlive/news2
```