Aem Task 7

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

```
@Component(
   service = Servlet.class,
   property = {
      Constants.SERVICE_DESCRIPTION + "= Sample Servlet",
      "sling.servlet.methods=GET",
      "sling.servlet.resourceTypes=myTraining/components/structure/page",
      "sling.servlet.extensions=json"
   }
)
public class SampleServlet extends SlingAllMethodsServlet {
   @Override
   protected void doGet(SlingHttpServletRequest request,
SlingHttpServletResponse response)
         throws ServletException, IOException {
      Resource resource = request.getResource();
      response.setContentType("application/json");
      response.getWriter().write("{\"message\": \"This is a servlet for: " +
resource.getResourceType() + "\"}");
   }
}
```

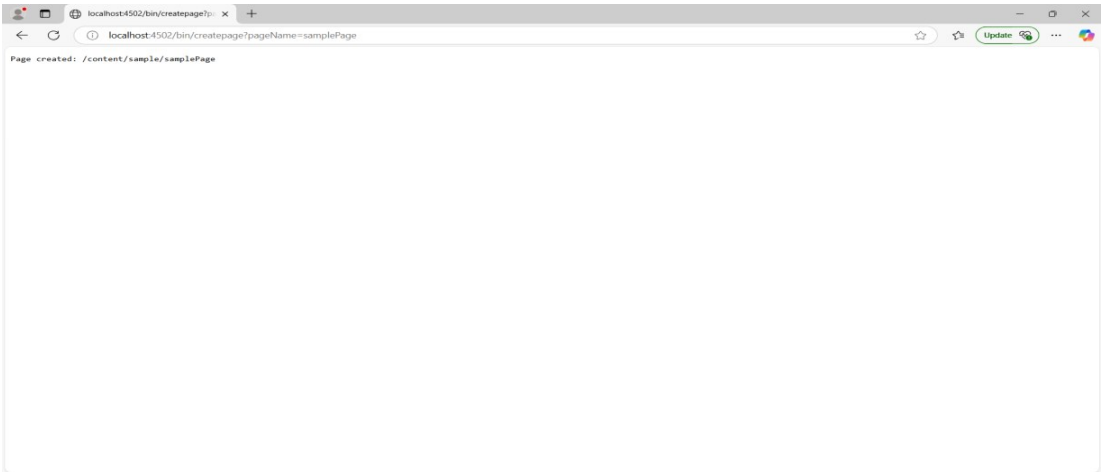
2.Create CreatePageServlet extend the SlingSafeMethod Servlet and register it using path

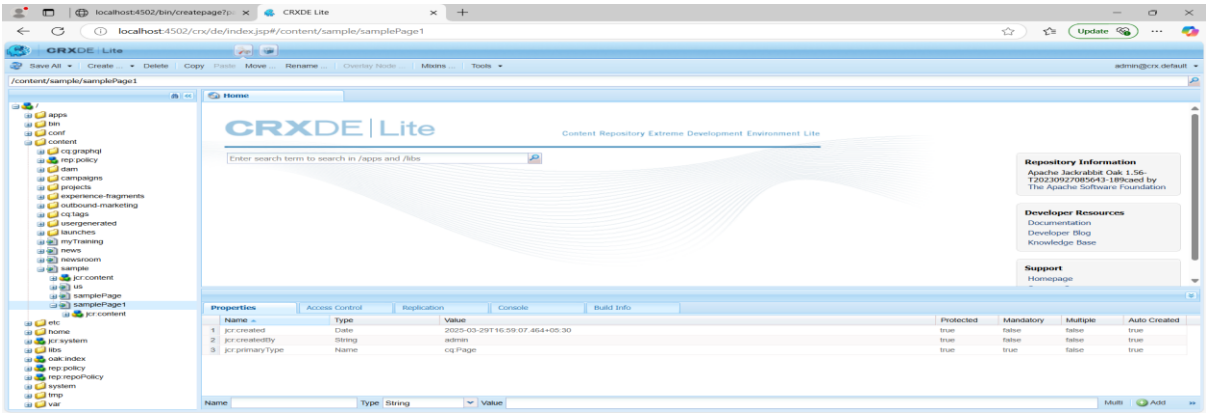
```
package com.sample.core.servlets;
import com.day.cq.wcm.api.Page;
import com.day.cq.wcm.api.PageManager;
import org.apache.sling.api.SlingHttpServletRequest;
import org.apache.sling.api.SlingHttpServletResponse;
import org.apache.sling.api.servlets.SlingSafeMethodsServlet;
import org.osgi.service.component.annotations.Component;
import org.osgi.framework.Constants;
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 = {
                Constants.SERVICE_DESCRIPTION + "= Create Page Servlet",
                "sling.servlet.paths=" + "/bin/createpage",
                "sling.servlet.methods=GET"
        }
)
```

```
public class CreatePageServlet extends SlingSafeMethodsServlet {

   private static final Logger LOG =
LoggerFactory.getLogger(CreatePageServlet.class);

   @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 missing!");
         return;
      }

      try {
         PageManager pageManager =
request.getResourceResolver().adaptTo(PageManager.class);

         if (pageManager != null) {
            Page newPage = pageManager.create("/content/sample", pageName,
"cq:Page", pageName);

            if (newPage != null) {
               response.getWriter().write("Page created: " + newPage.getPath());
```

```
                LOG.info("Page created at: {}", newPage.getPath());
            } else {
                response.getWriter().write("Page creation failed.");
            }
        } else {
            response.getWriter().write("Failed to adapt to PageManager.");
        }
    } catch (Exception e) {
        LOG.error("Error creating page", e);
        response.getWriter().write("Error creating page: " + e.getMessage());
    }
  }
}
```

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





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

```
package com.aemtasks.core.servlets;

import com.day.cq.search.PredicateGroup;

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.Result;

import org.apache.sling.api.SlingHttpServletRequest;

import org.apache.sling.api.SlingHttpServletResponse;

import org.apache.sling.api.resource.ResourceResolver;

import org.apache.sling.api.servlets.SlingSafeMethodsServlet;

import org.osgi.service.component.annotations.Component;

import org.osgi.service.component.annotations.Reference;

import javax.jcr.Session;

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"
})
public class SearchServlet extends SlingSafeMethodsServlet {
@Reference
private QueryBuilder queryBuilder;
@Override
protected
void
doGet(SlingHttpServletRequest
SlingHttpServletResponse response)
throws ServletException, IOException {
request,

    response.setContentType("text/plain");
    String searchText = request.getParameter("searchText");

    ResourceResolver resolver = request.getResourceResolver();
    Session session = resolver.adaptTo(Session.class);

    if (searchText != null && session != null) {
        Map<String, String> map = new HashMap<>();
        map.put("path", "/content");
        map.put("type", "cq:Page");
```

```
            map.put("fulltext", searchText);
            map.put("p.limit", "-1");

            Query query =
queryBuilder.createQuery(PredicateGroup.create(map), session);
            Result result = query.getResult();

            for (Hit hit : result.getHits()) {
                response.getWriter().write(hit.getPath() + "\n");
            }
        } else {
            response.getWriter().write("Missing searchText or session.");
        }
    }
}
```