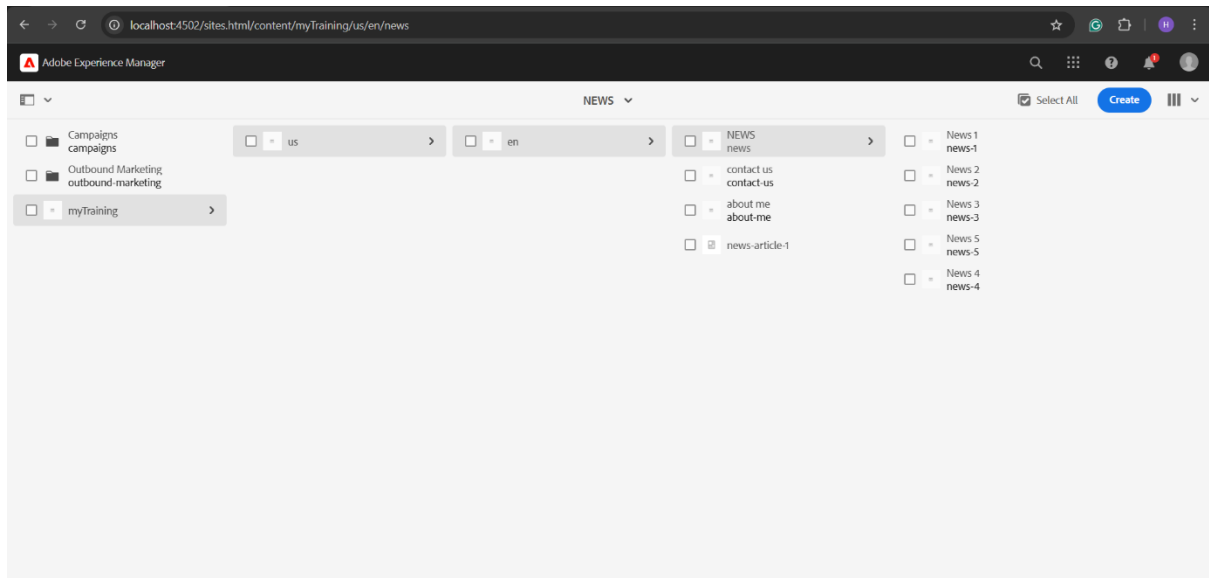


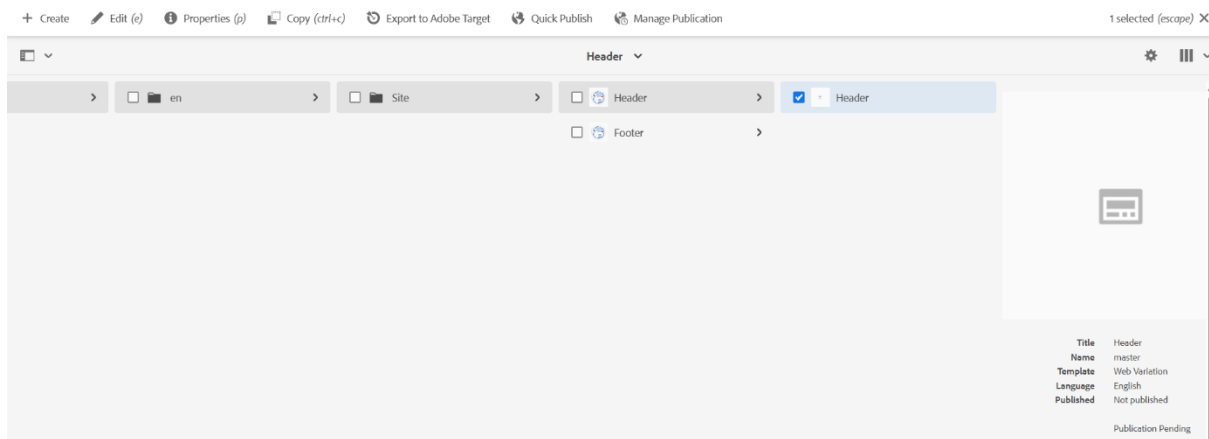
Newsroom Implementation and Custom Features

Creating News Article Pages:

Five unique news article pages are created under /content/us/en/news. Each page uses the News Component to display the title, news details, and published date.



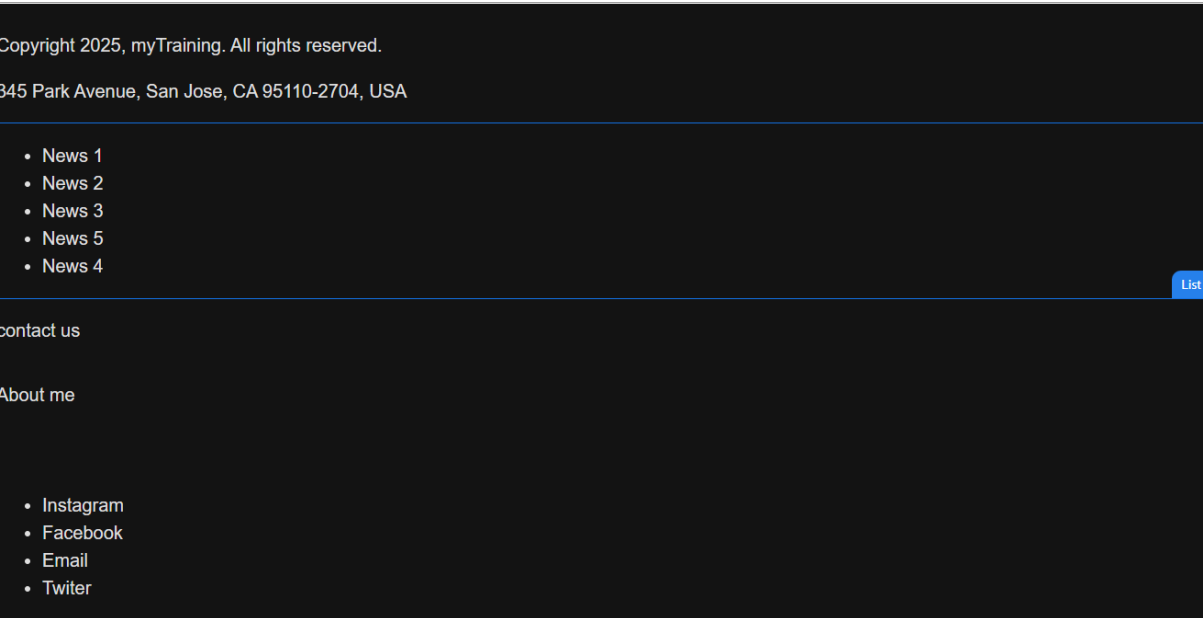
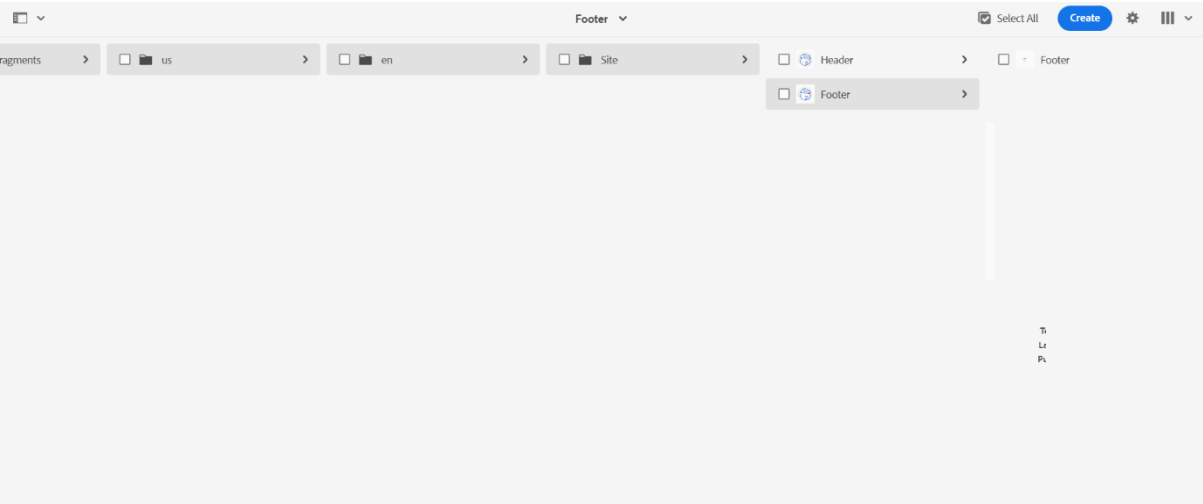
Create Header Experience fragment for header and use these page as menu:



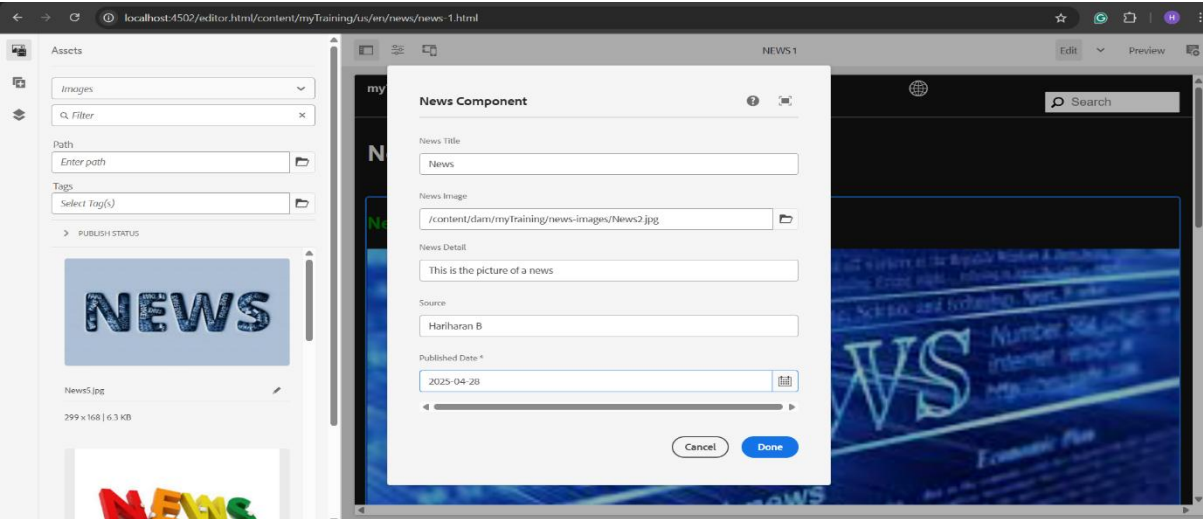
- [News 1](#)
- [News 2](#)
- [News 3](#)
- [News 5](#)
- [News 4](#)

Navigation

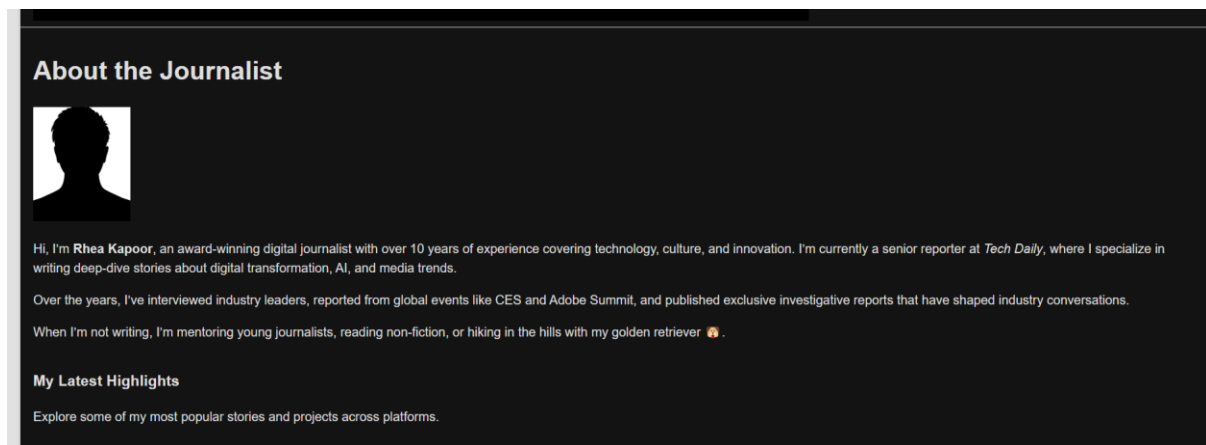
Create footer XF and it could have 4 sections



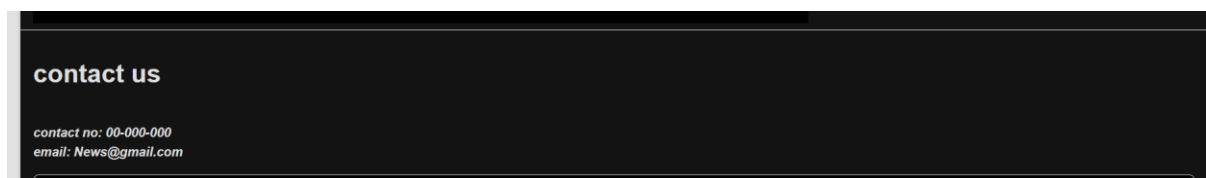
Create news component



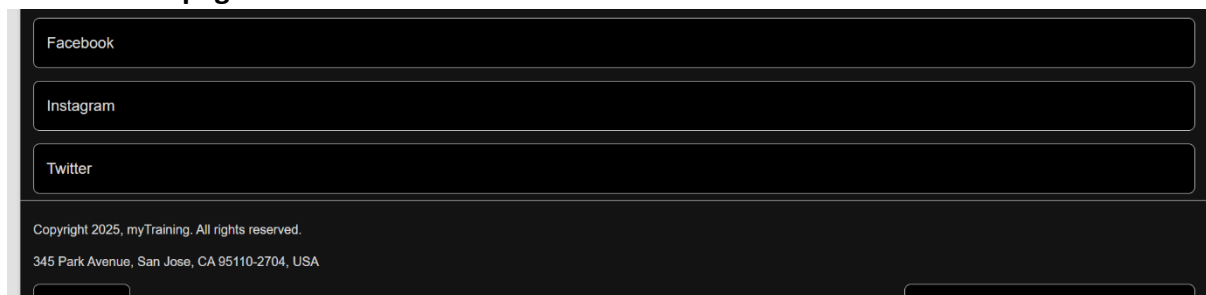
About us page



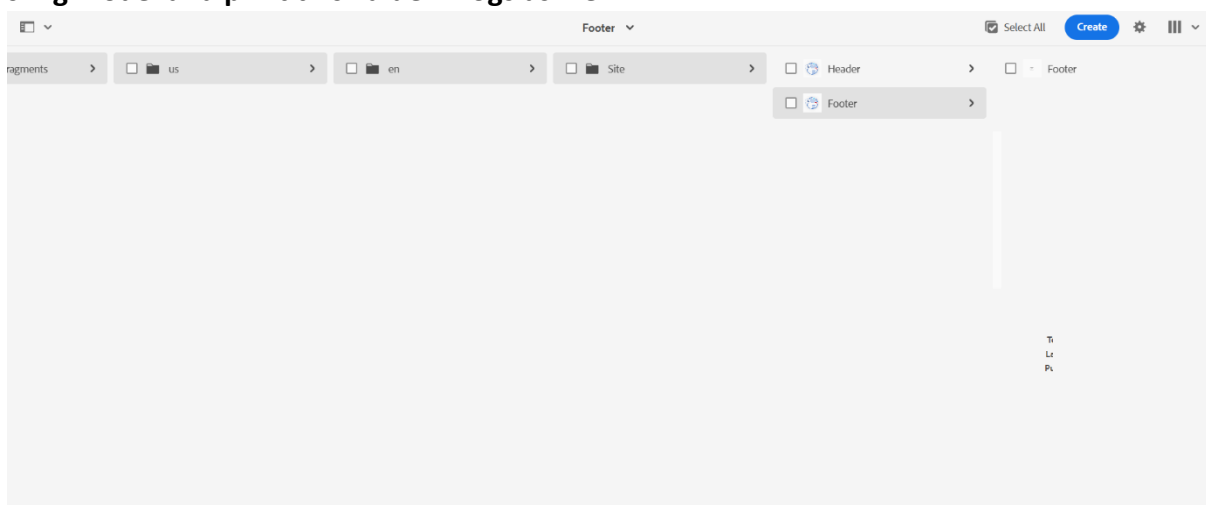
Contact us



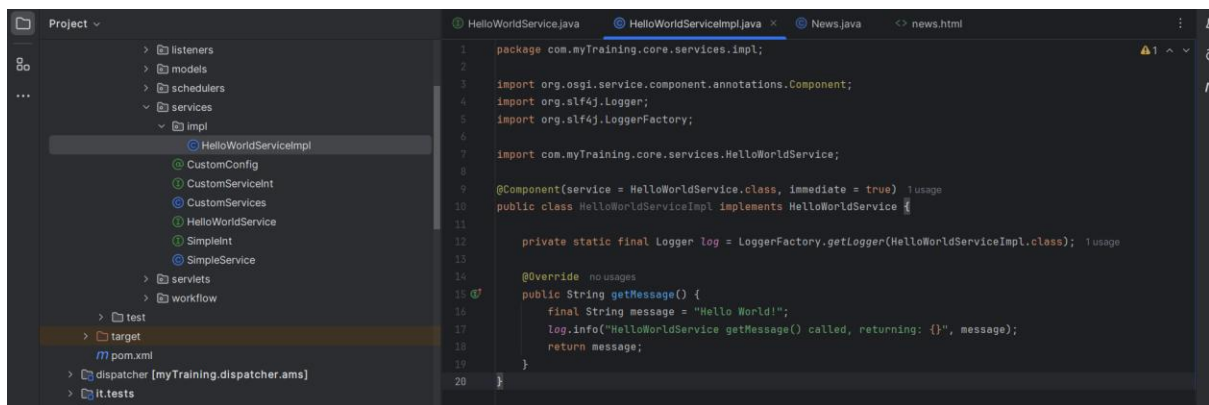
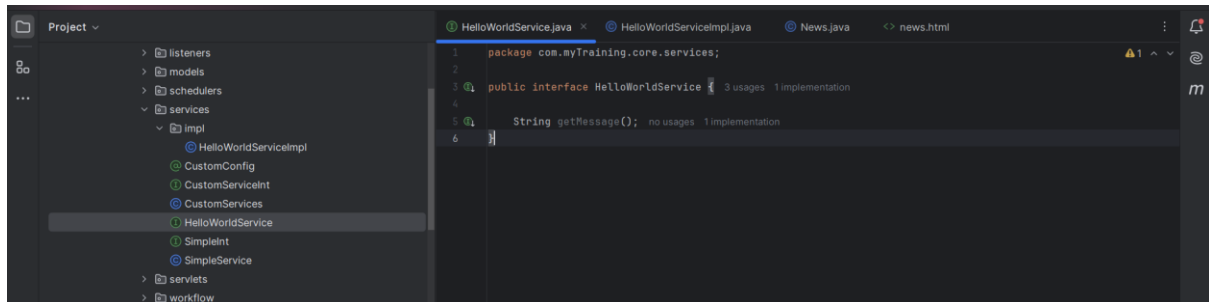
Social media page



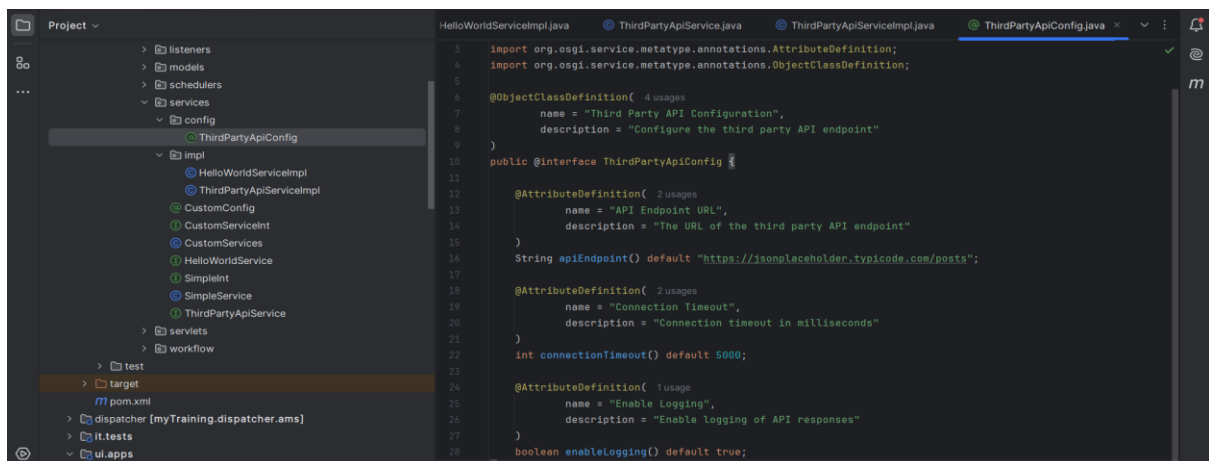
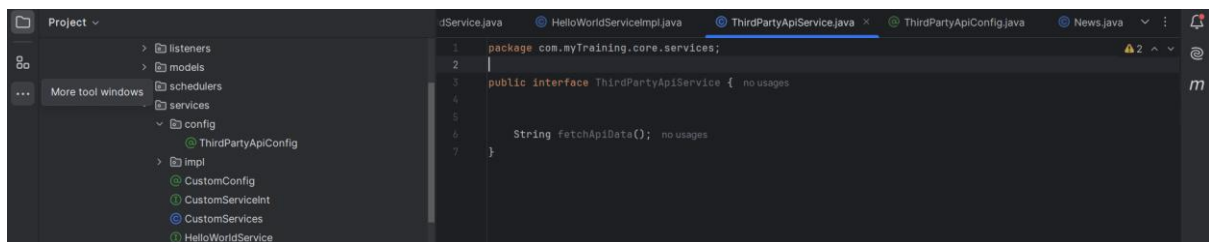
Create a custom service to print hello world and call this service from news component sling model and print this value in logs as well.



Create a custom service to print hello world and call this service from news component sling model and print this value in logs as well.

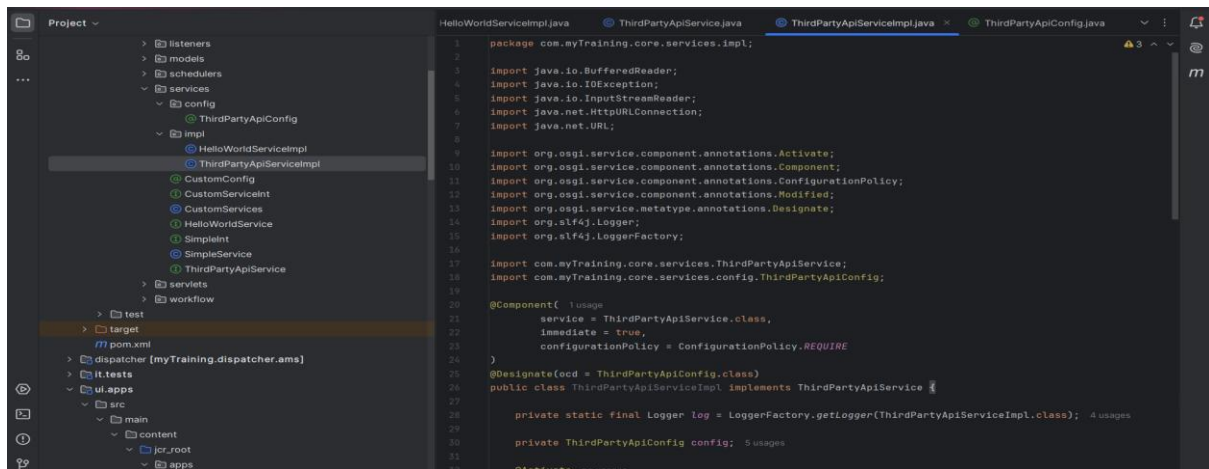


Create custom configurations where I can provide the 3rd party api for exmample(<https://jsonplaceholder.typicode.com/posts>) and I can see the data as json. And this data should be print in logs





```
1 <sly data-sly-use.newsModel="com.myTraining.core.models.News">
2   <div class="news-component cop-news-component">
12
13   </p>
14
15   <p class="news-source">Source: ${newsModel.source}</p>
16
17   <div class="hello-message" data-sly-test="${newsModel.helloMessage}">
18     <p>Service Message: ${newsModel.helloMessage}</p>
19   </div>
20   <div class="api-data" data-sly-test="${newsModel.hasApiData}">
21     <h3>API Data Preview</h3>
22     <button onclick="toggleApiData()">Show/Hide API Data</button>
23     <pre id="apiDataContent" style="display: none;">${newsModel.apiData @ context='html'}</pre>
24   </div>
25 </div>
34
```



```
1 package com.myTraining.core.services.impl;
2
3 import java.io.BufferedReader;
4 import java.io.IOException;
5 import java.io.InputStreamReader;
6 import java.net.HttpURLConnection;
7 import java.net.URL;
8
9 import org.osgi.service.component.annotations.Activate;
10 import org.osgi.service.component.annotations.Component;
11 import org.osgi.service.component.annotations.ConfigurationPolicy;
12 import org.osgi.service.component.annotations.Modified;
13 import org.osgi.service.metatype.annotations.Designate;
14 import org.slf4j.Logger;
15 import org.slf4j.LoggerFactory;
16
17 import com.myTraining.core.services.ThirdPartyApiService;
18 import com.myTraining.core.services.config.ThirdPartyApiConfig;
19
20 @Component(
21     service = ThirdPartyApiService.class,
22     immediate = true,
23     configurationPolicy = ConfigurationPolicy.REQUIRE
24 )
25 @Designate(ocd = ThirdPartyApiConfig.class)
26 public class ThirdPartyApiServiceImpl implements ThirdPartyApiService {
27
28     private static final Logger log = LoggerFactory.getLogger(ThirdPartyApiServiceImpl.class);
29
30     private ThirdPartyApiConfig config;
31
32     @Activate
33     @Modified
34     public void activate(BundleContext context) throws Exception {
35         log.info("ThirdPartyApiServiceImpl activated");
36         this.config = context.getServiceReference(ThirdPartyApiConfig.class)
37             .getService(ThirdPartyApiConfig.class);
38     }
39
40     @Override
41     public String getHelloMessage() {
42         return "Hello Message: " + config.getHelloMessage();
43     }
44
45     @Override
46     public boolean hasApiData() {
47         return config.hasApiData();
48     }
49
50     @Override
51     public String getApiData() {
52         return config.getApiData();
53     }
54
55     @Override
56     public void toggleApiData() {
57         log.info("Toggle API Data");
58     }
59
60     @Override
61     public String getSource() {
62         return config.getSource();
63     }
64
65     @Override
66     public void deactivate() {
67         log.info("ThirdPartyApiServiceImpl deactivated");
68     }
69 }
```