1. **Create 5 News Article Pages**

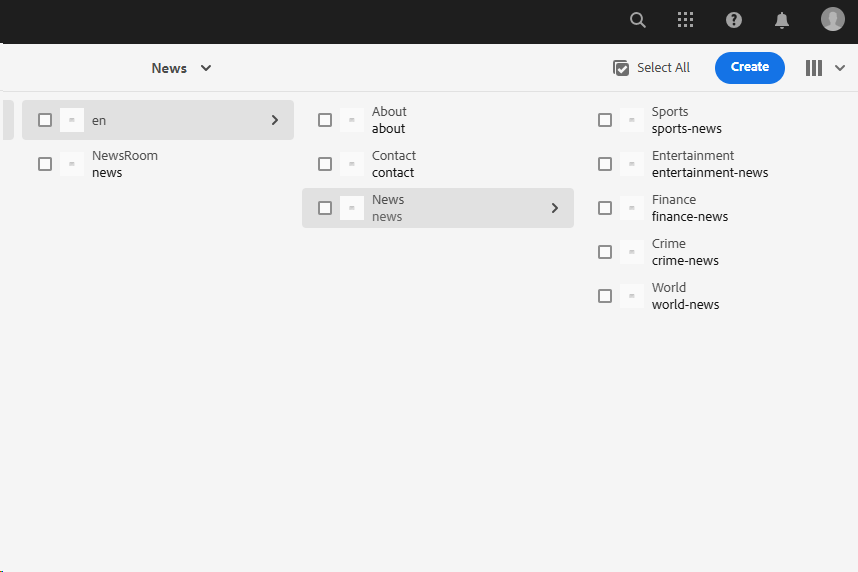
**Objective**: Create 5 unique news article pages within AEM to display content.

# Steps:

* 1. **Navigate to Content Folder**:
     + Go to **/content/us/en/news** in AEM’s CRX/DE or AEM Author instance.

# Create News Article Pages:

* + - Right-click on the /news folder and create 5 new pages.
    - Each page should have a unique title, for example:
      * News Article 1
      * News Article 2
      * News Article 3
      * News Article 4
      * News Article 5
    - Ensure each page has distinct content for each article.



1. **Use News Component**

**Objective**: Add a previously created **News Component** to each news page to display content like title, detail, and published date.

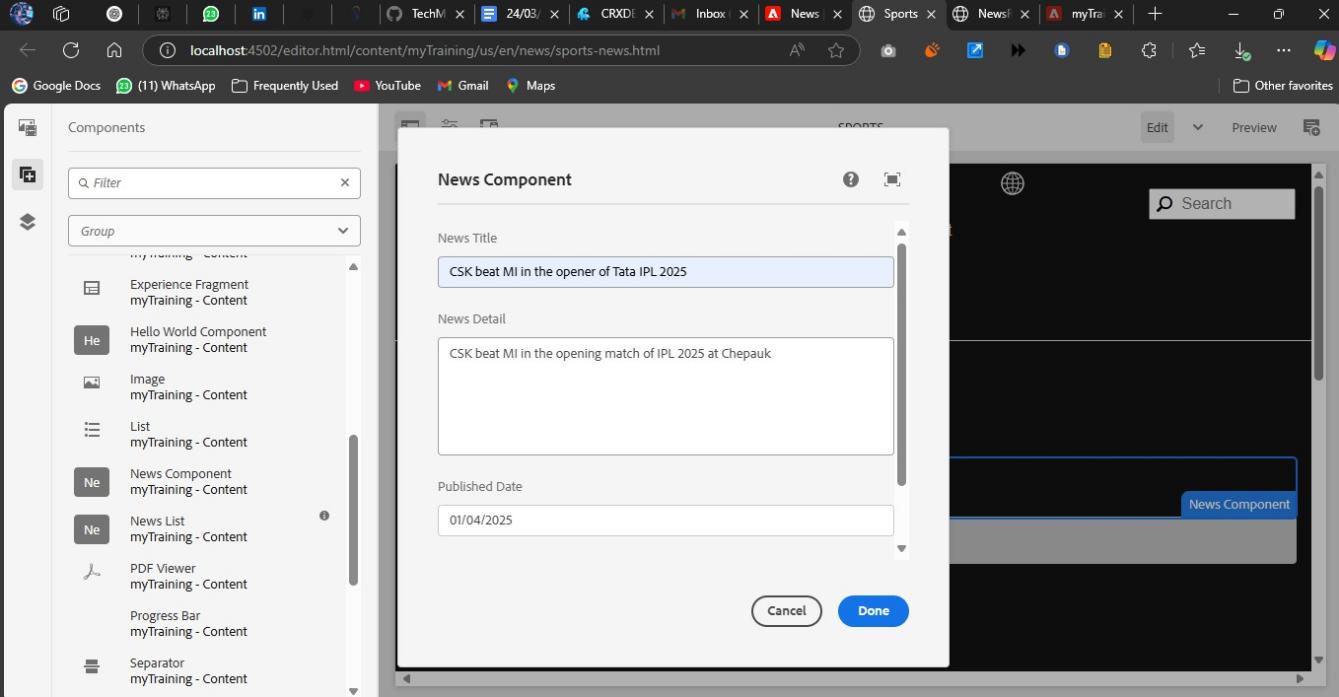
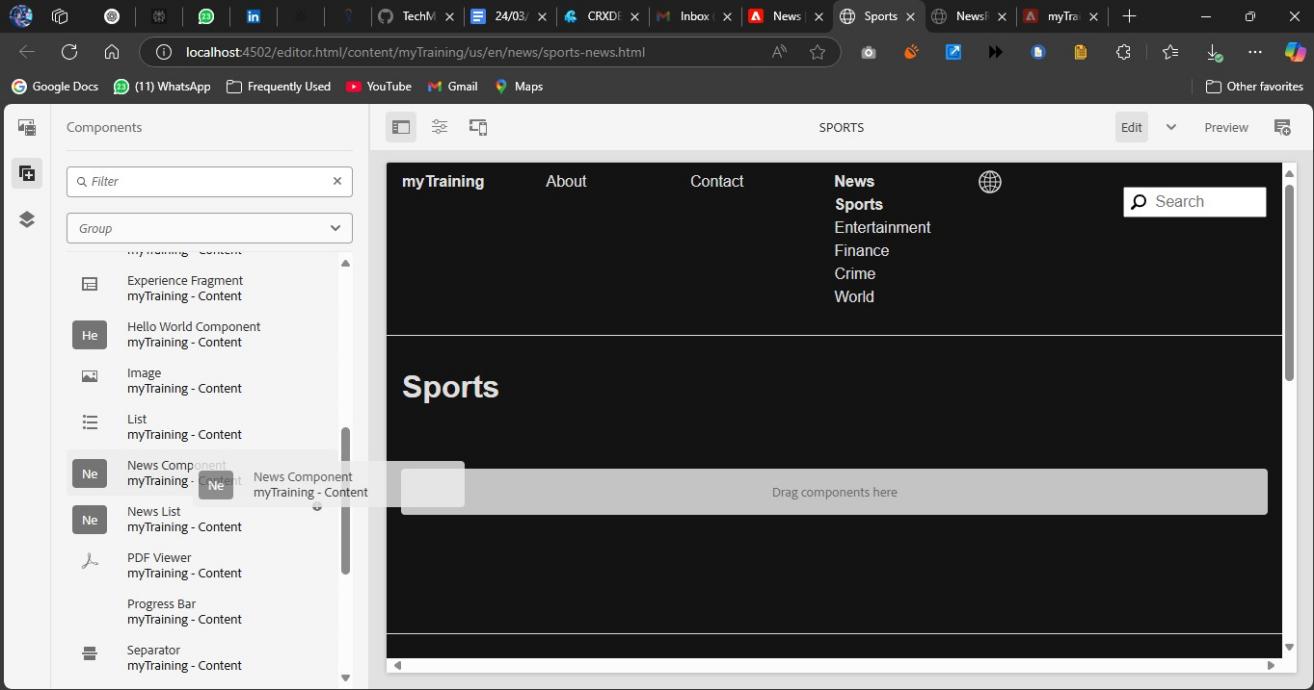
# Steps:

* 1. **Create or Locate the News Component**:
     + If not already created, ensure the **News Component** is available under

/apps/myTraining/components/structure/news.

# Add Component to Each News Page:

* + - Edit each of the 5 news article pages.
    - Add the **News Component** to the page by dragging it from the sidekick (or adding it via the component tab in the page editor).



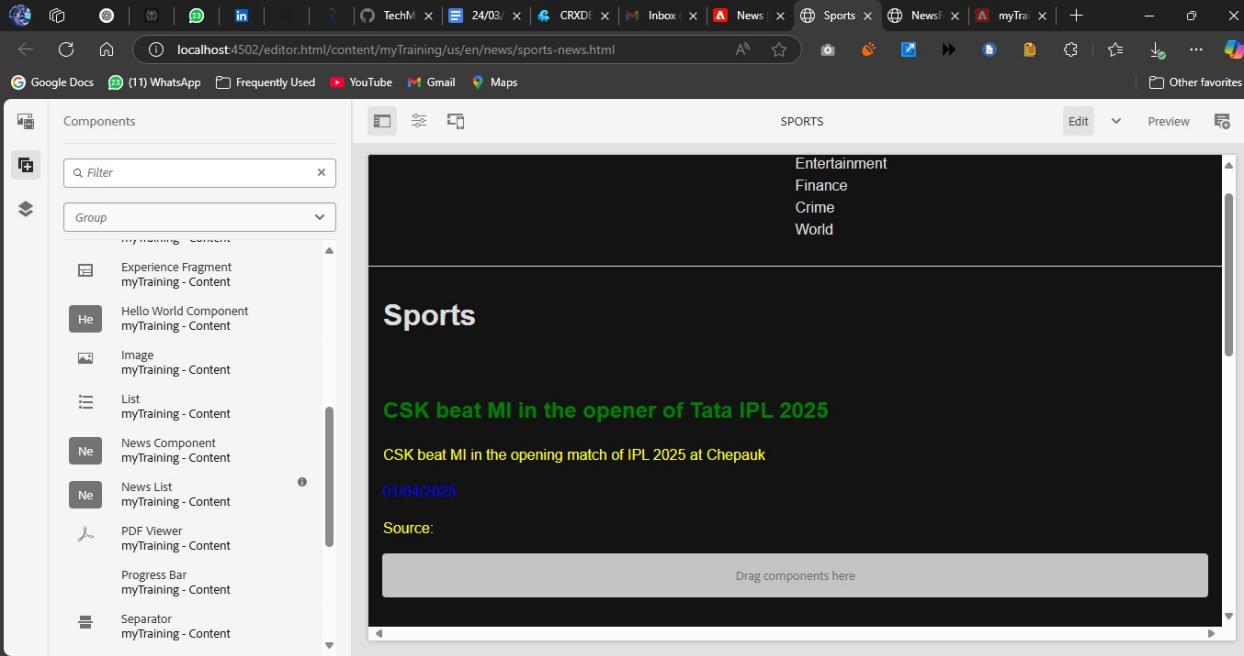
# Customize the News Component:

<div class="news-item">

<h2 style="color: green;">${title}</h2>

<p style="color: yellow;">${newsDetail}</p>

<span style="color: black;">Published on: ${publishedDate}</span></div>



1. **Create Header Experience Fragment**

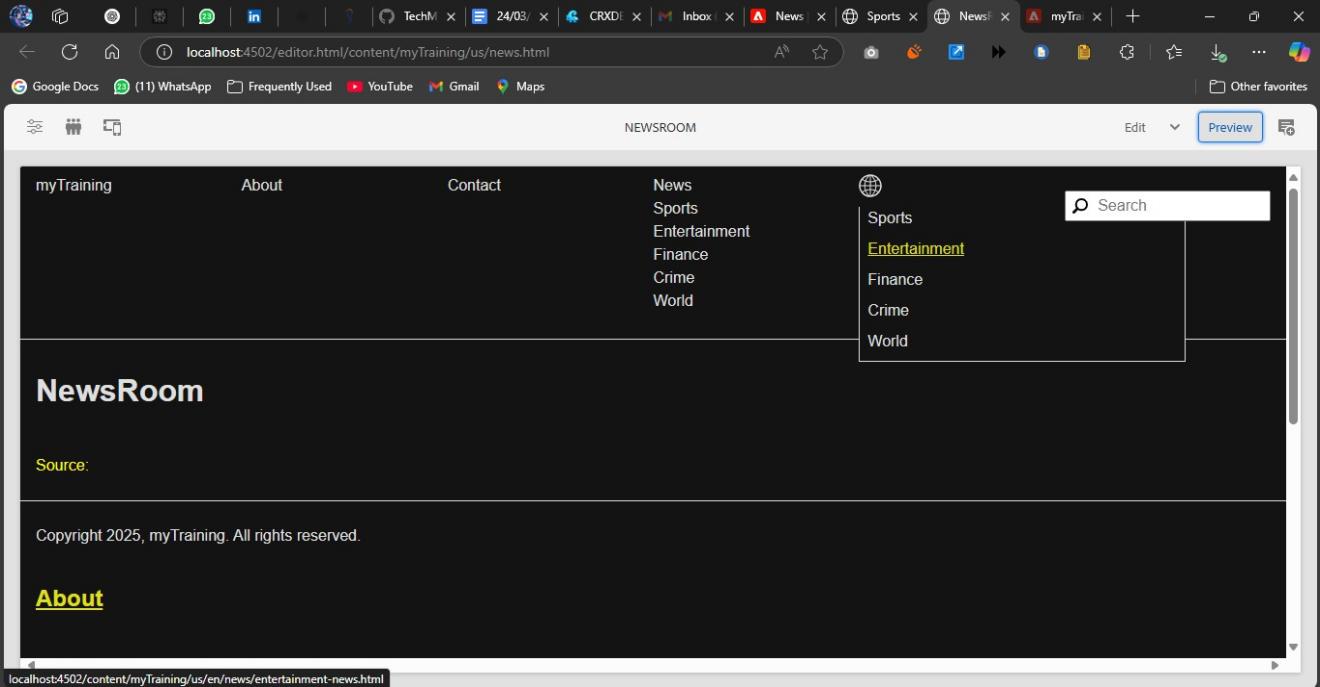
**Objective**: Design a **Header Experience Fragment** to contain the navigation menu and important links.

# Steps:

* 1. **Navigate to Experience Fragments**:
     + Go to **/content/experience-fragments** in AEM.
     + Create a new experience fragment for the header (e.g., header-fragment).

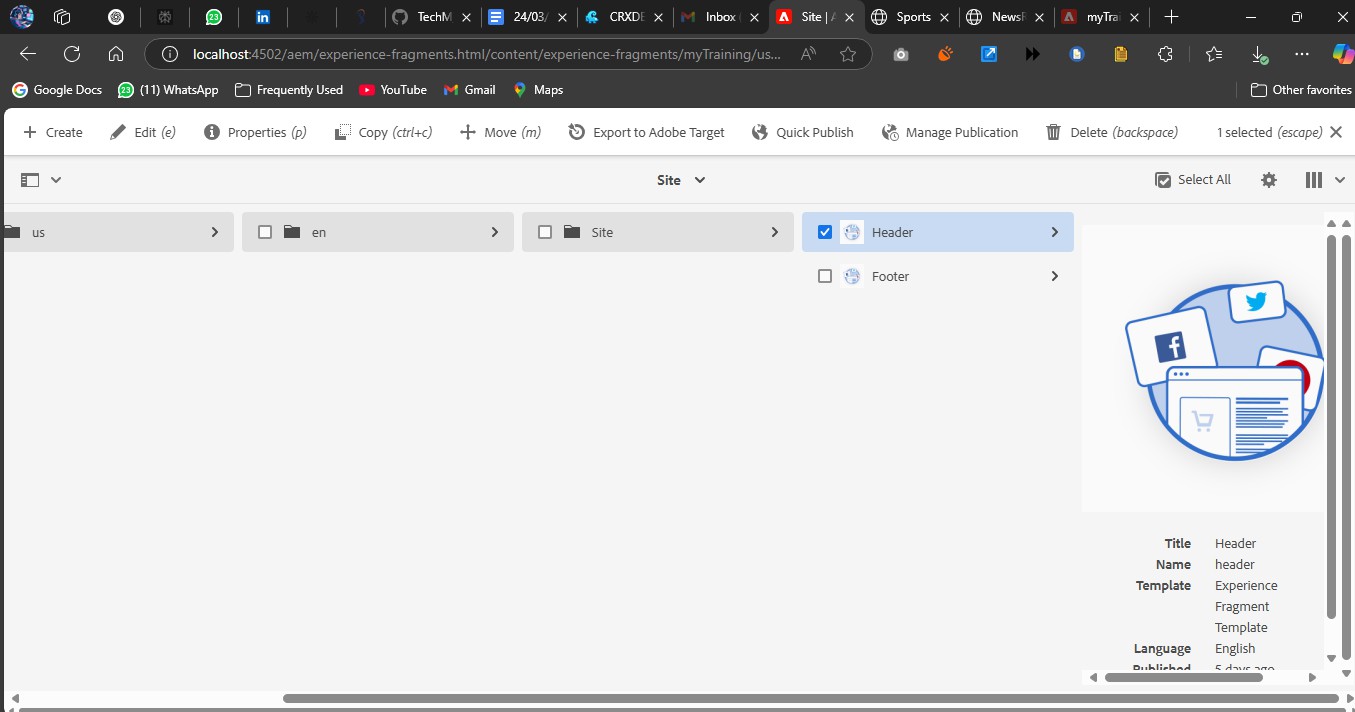
# Design the Header:

* + - Add a **Navigation Menu** component.
    - Link the following pages:
      * News Menu (Links to news pages)
      * Contact Us Page
      * About Me Page



# Publish the Experience Fragment:

* + - Once designed, make sure the experience fragment is published and ready for use across pages.



1. **Create Footer Experience Fragment**

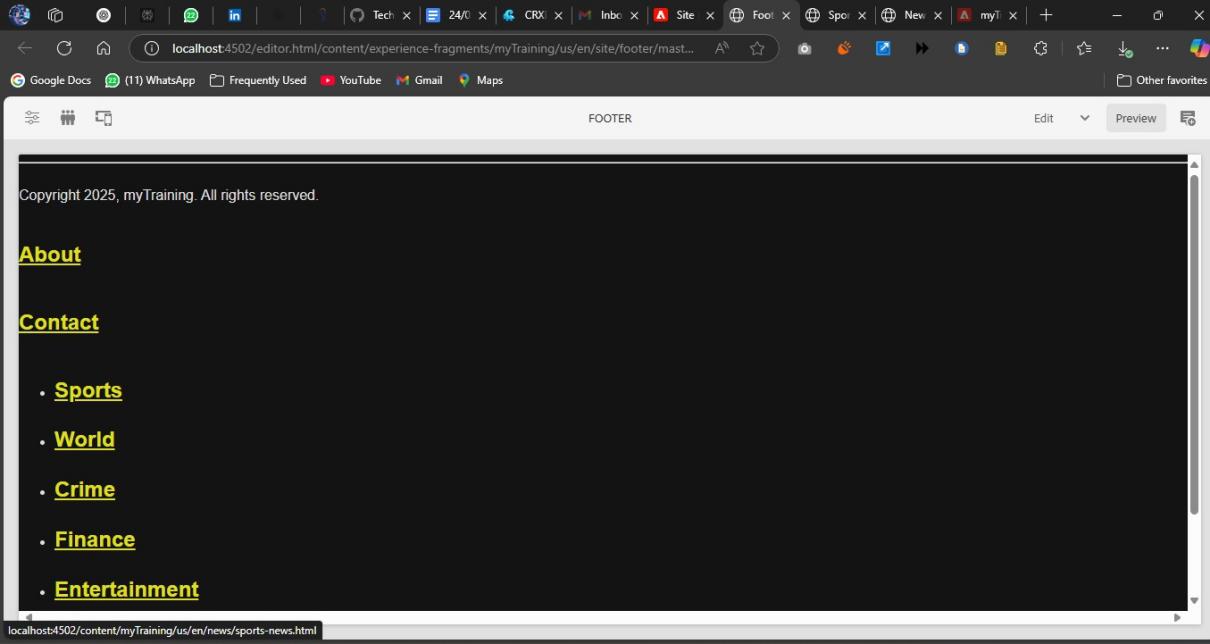
**Objective**: Create a **Footer Experience Fragment** with multiple sections, including news articles, contact information, and social media links.

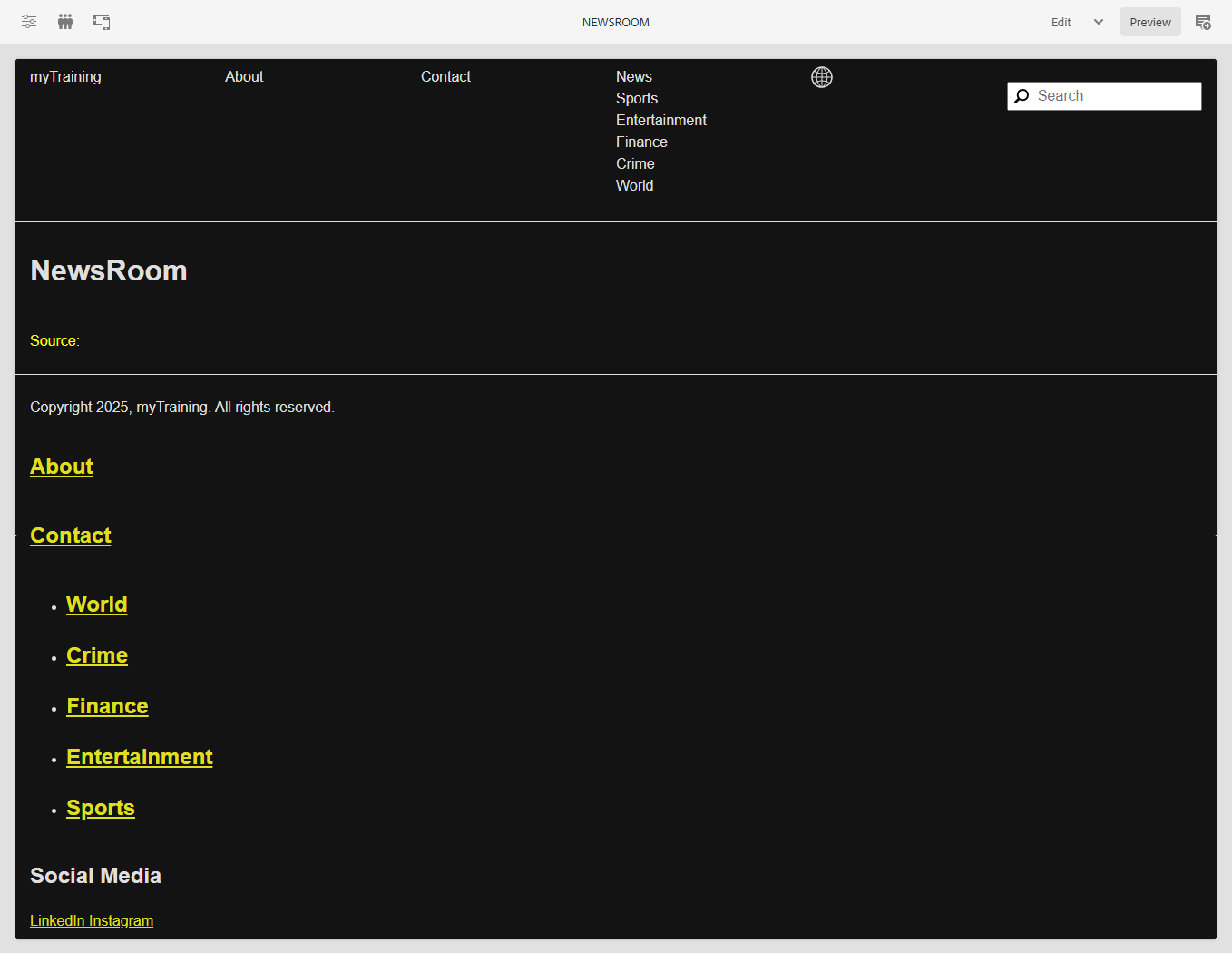
# Steps:

* 1. **Navigate to Experience Fragments**:
     + Go to **/content/experience-fragments** in AEM.
     + Create a new experience fragment for the footer (e.g., footer-fragment).

# Design the Footer:

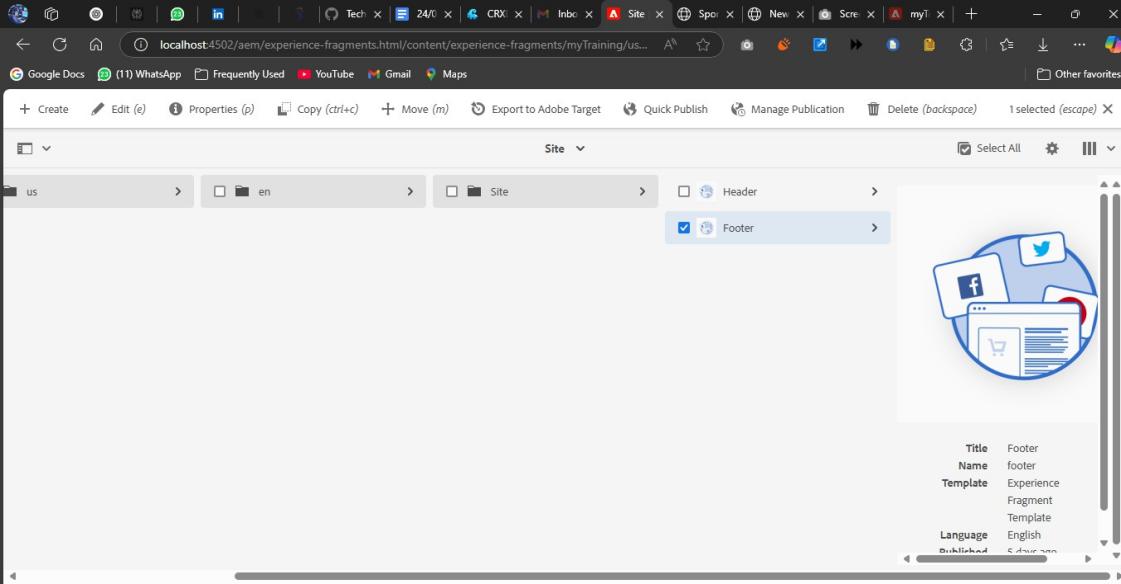
* + - Add the following sections using appropriate components:
      * **News Menu Section**: Add a **List Component** to display the 4 most recent news articles.
      * **About Me Section**: Add a **Text Component** to provide brief information about the journalist.
      * **Contact Us Section**: Add a **Text Component** to list the contact details (email, phone, office address).
      * **Social Media Section**: Add a **List Component** for links to social media accounts.





# Publish the Experience Fragment:

* + - Once complete, publish the footer experience fragment.



1. **Create Custom Service**

**Objective**: Develop a **Custom Service** in AEM that prints Hello World and is called within the **News Component's Sling Model**.

# Steps:

* 1. **Create a Service Interface**:

In your core module, create a service interface like NewsService: public interface NewsService {

String sayHello();

}

# Create the Service Implementation:

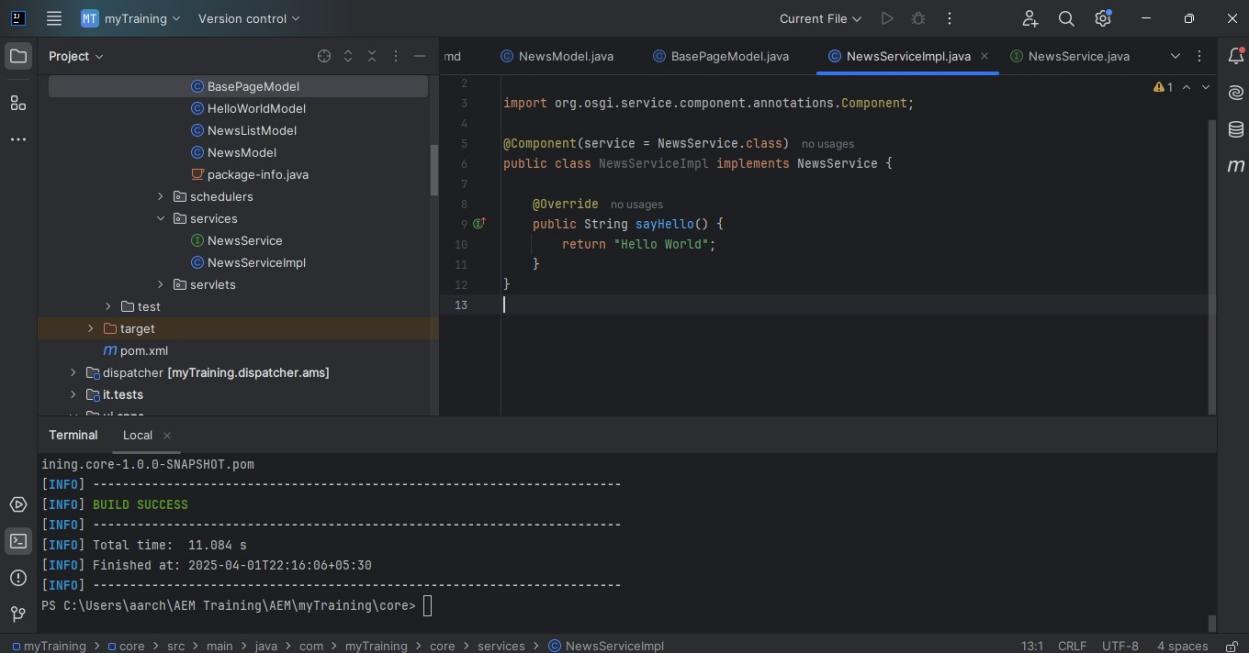
Implement the service interface in a new class NewsServiceImpl: @Component(service = NewsService.class)

public class NewsServiceImpl implements NewsService { @Override

public String sayHello() { return "Hello World";

}

}



# Inject and Call the Service in Sling Model:

In the BasePageModel or a new NewsComponentModel, inject and use the service: @Inject

private NewsService newsService;

public String getGreeting() { return newsService.sayHello();

}

# Log the Output:

Log the service output in the AEM logs to confirm it’s working: @Activate

@Modified

public void logGreeting() {

String greeting = getGreeting(); LOGGER.info(greeting); // Logs "Hello World"

}

1. **Create Custom Configuration**

**Objective**: Create a **Custom Configuration** to store a third-party API URL and fetch JSON data from it.

# Steps:

* 1. **Create the Configuration Interface**:

Define a Sling Model or OSGi configuration to store the API URL. @Designate(ocd = MyConfig.class)

public class MyConfig { @Activate @Modified

public void activate() {

String apiUrl = config.apiUrl(); LOGGER.info("Configured API URL: " + apiUrl);

}

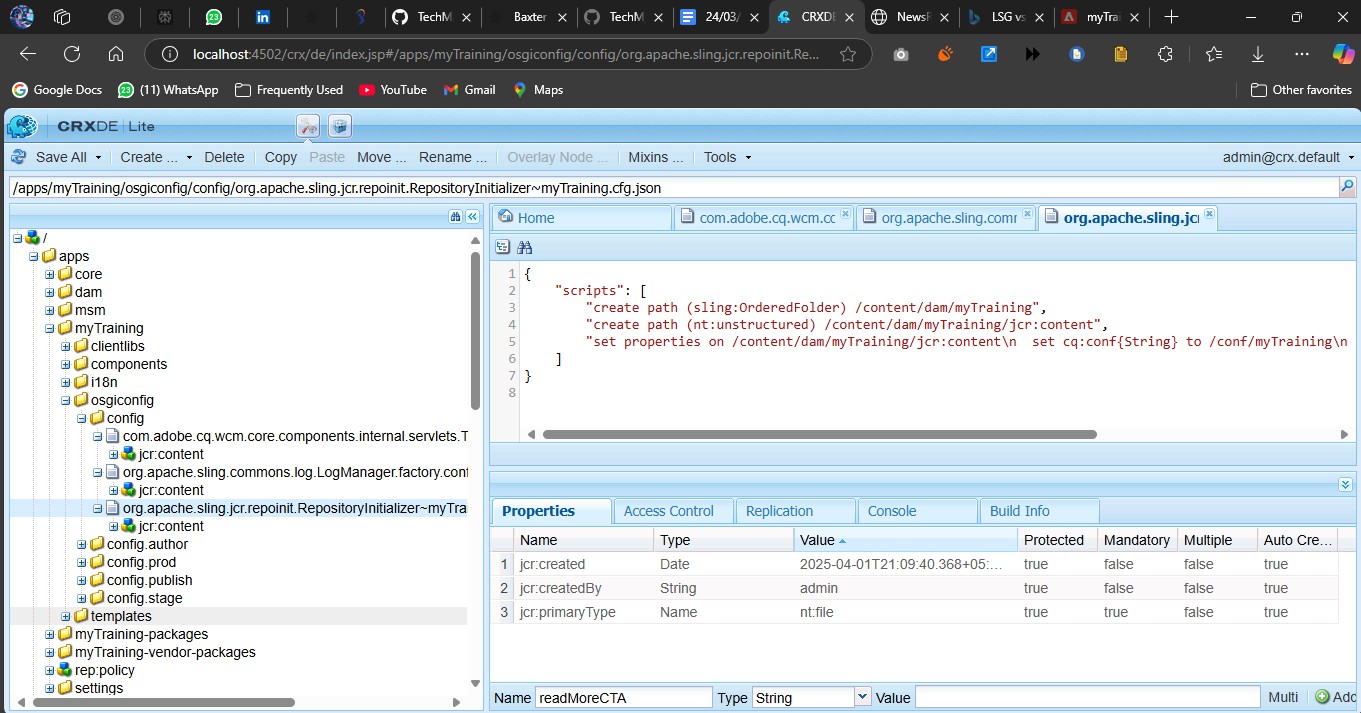
@Activate @Modified @Property

private String apiUrl;

}

# Create the Configuration Dialog:

* + - Add a configuration dialog under /apps/myTraining/configs where you can input the third-party API URL, such as <https://jsonplaceholder.typicode.com/posts>.



# Fetch API Data:

In the service or Sling model, fetch data from the configured API URL: HttpClient client = HttpClients.createDefault();

HttpGet request = new HttpGet(apiUrl); HttpResponse response = client.execute(request);

String jsonResponse = EntityUtils.toString(response.getEntity()); LOGGER.info("Fetched API Response: " + jsonResponse);