**1.Create a New News Component Using Sling Model**

You need to create a Sling Model for the News component, which includes:

* News Title (h2)
* News Detail (<p>)
* Published Date
* Source

Path : C:\Users\LENOVO\Training\aem\codebase\news\core\src\main\java\com\myTraining\core\models

#### ****Sling Model (NewsComponent.java)****

package com.myTraining.core.models;

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

import org.apache.sling.models.annotations.DefaultInjectionStrategy;

import org.apache.sling.models.annotations.Model;

import org.apache.sling.models.annotations.injectorspecific.ValueMapValue;

import javax.inject.Inject;

import java.text.SimpleDateFormat;

import java.util.Calendar;

@Model(adaptables = Resource.class, defaultInjectionStrategy = DefaultInjectionStrategy.OPTIONAL)

public class NewsModel {

@Inject

private String text;

@Inject

private String newsDetail;

@Inject

private Calendar publishedDate; // FIXED: Changed to Calendar type

public String getText() {

return text;

}

public String getNewsDetail() {

return newsDetail;

}

public String getFormattedPublishedDate() {

if (publishedDate != null) {

SimpleDateFormat sdf = new SimpleDateFormat("dd MMM yyyy");

return sdf.format(publishedDate.getTime()); // FIXED: Now works with Calendar

}

return "N/A";

}

}

**News.html**

Path: C:\Users\LENOVO\Training\aem\codebase\news\ui.apps\src\main\content\jcr\_root\apps\news\components\news

<div class="cmp-news" data-cmp-is="news" data-sly-use.model="com.myTraining.core.models.NewsModel">

<h2 class="cmp-news\_\_title">${model.newsTitle}</h2>

<div class="cmp-news\_\_item" data-sly-test="${model.newsDetail}">

<p class="cmp-news\_\_label">News Detail:</p>

<pre class="cmp-news\_\_output">${model.newsDetail}</pre>

</div>

<div class="cmp-news\_\_item" data-sly-test="${model.formattedPublishedDate}">

<p class="cmp-news\_\_label">Published Date:</p>

<pre class="cmp-news\_output cmp-news\_date">${model.formattedPublishedDate}</pre>

</div>

<div class="cmp-news\_\_item" data-sly-test="${model.newsSource}">

<p class="cmp-news\_\_label">Source:</p>

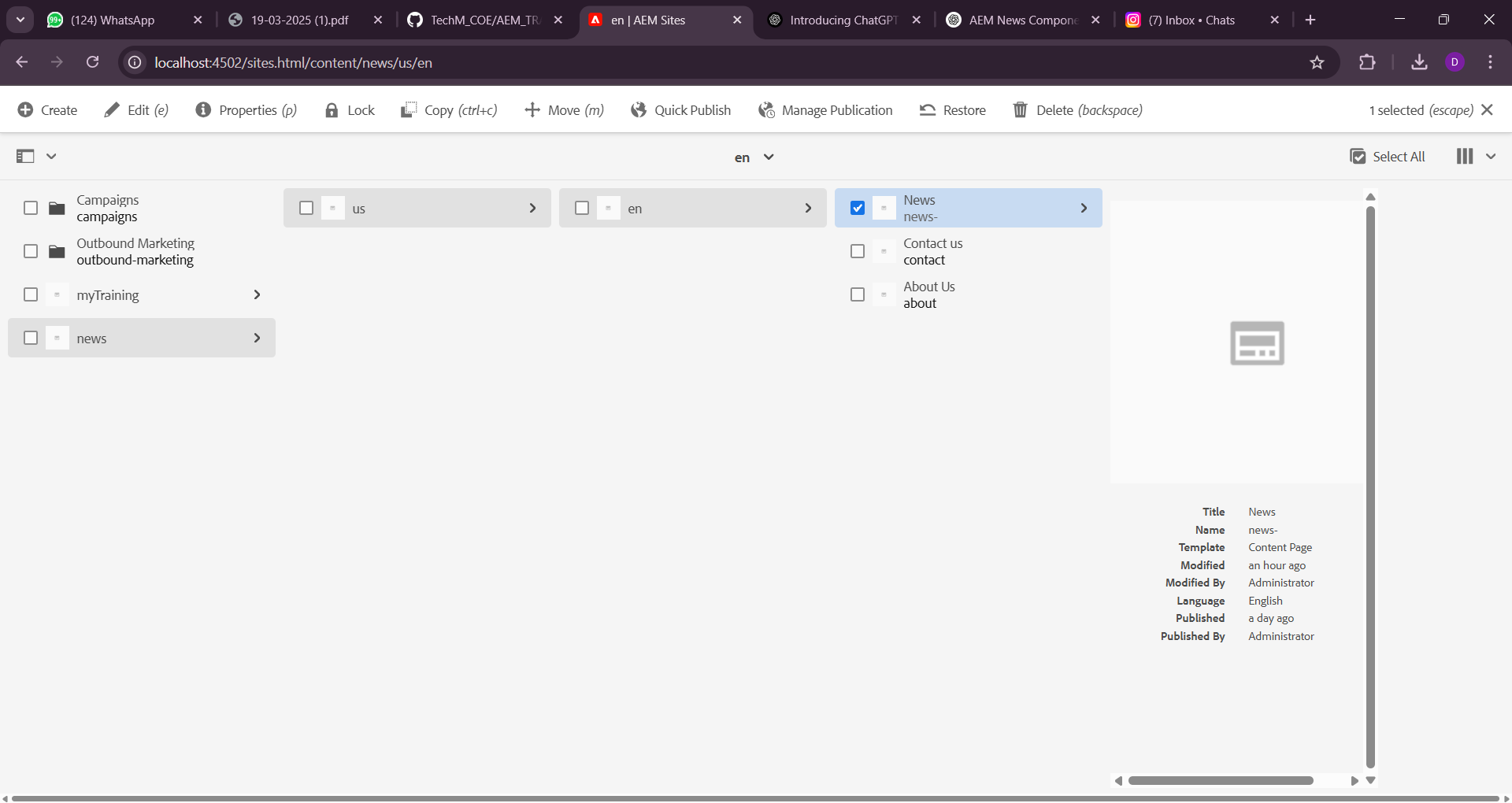
<pre class="cmp-news\_\_output">${model.newsSource}</pre>

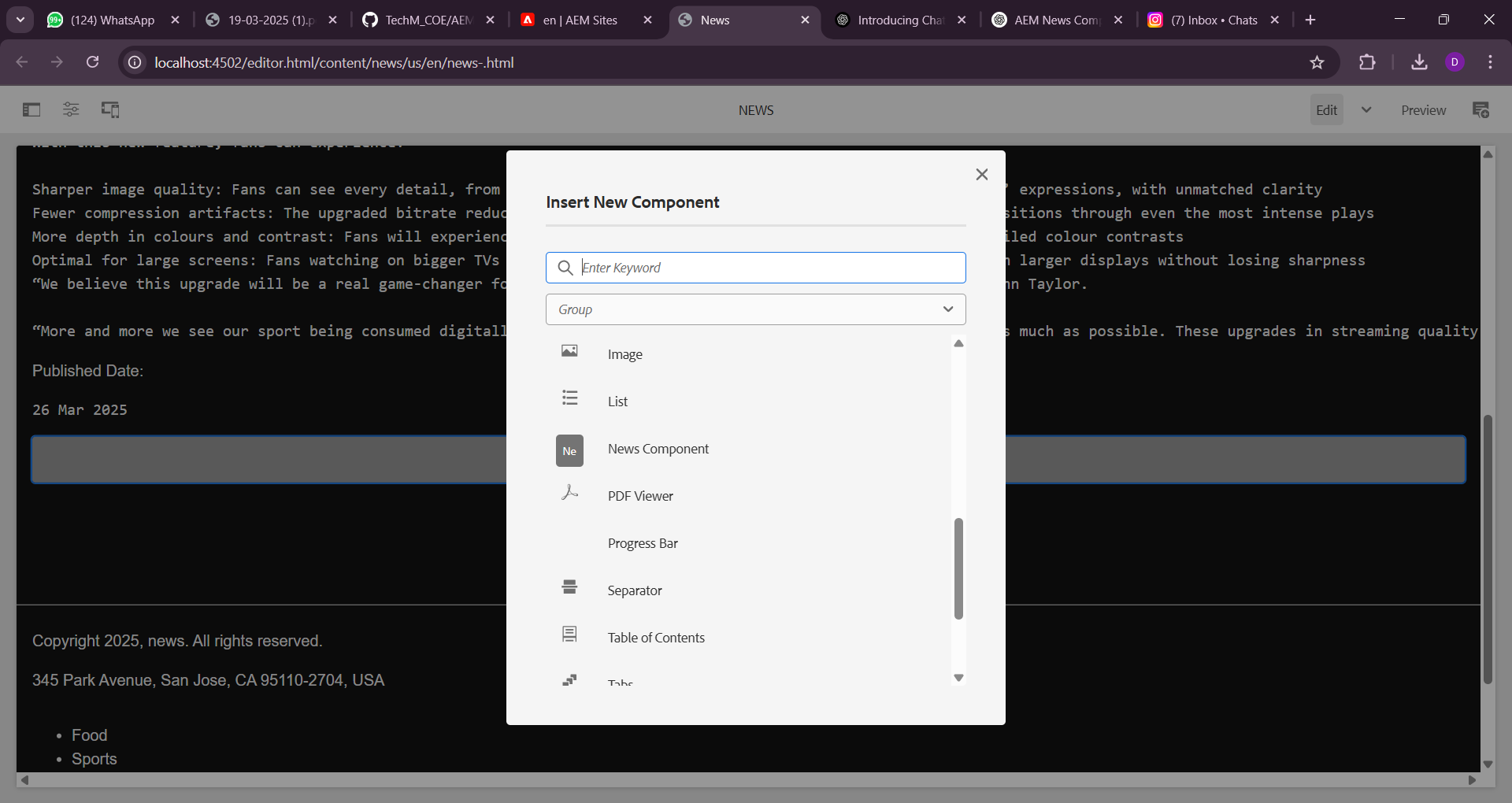
</div>

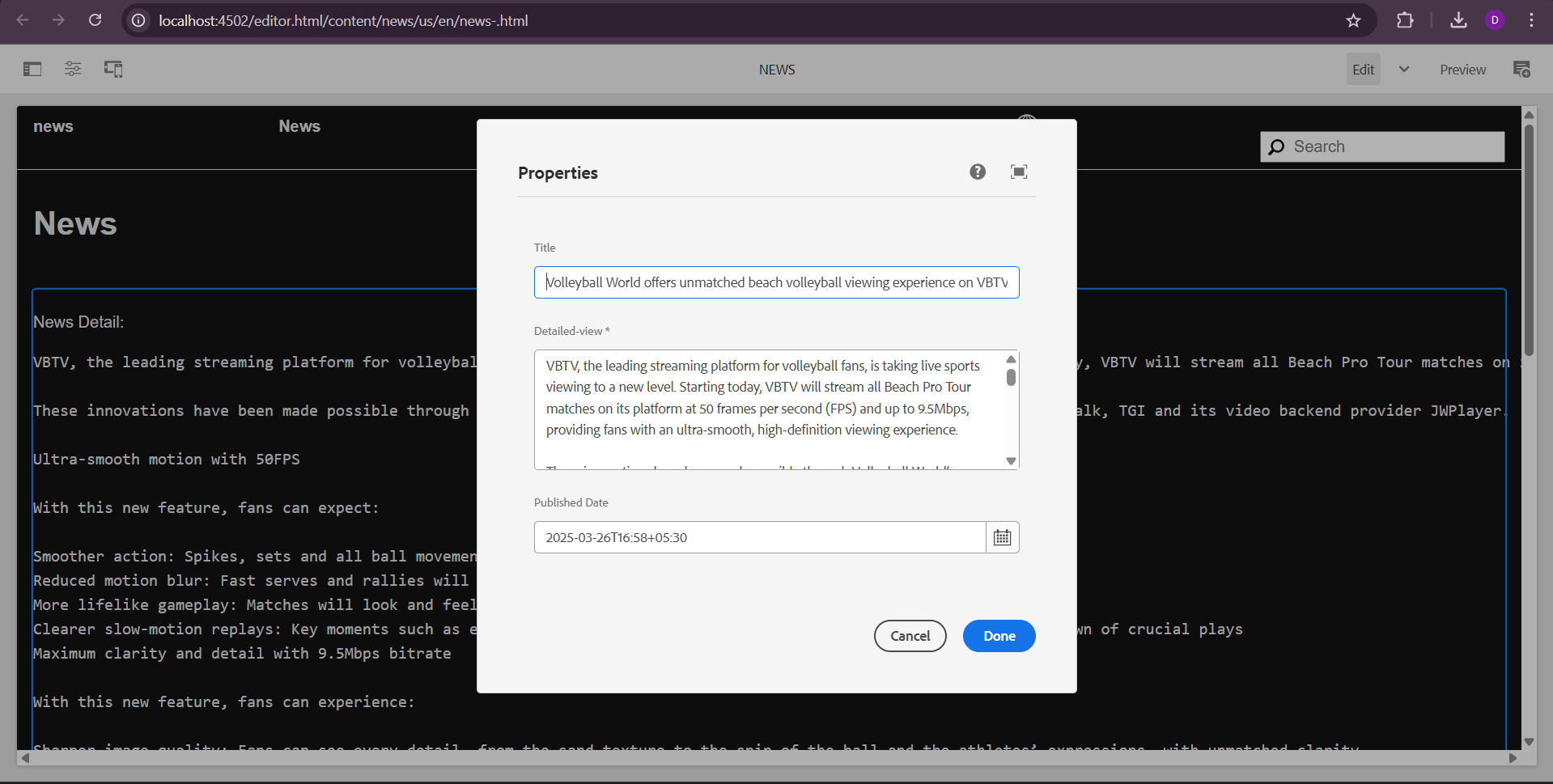
<sly data-sly-use.clientLib="/libs/granite/sightly/templates/clientlib.html" />

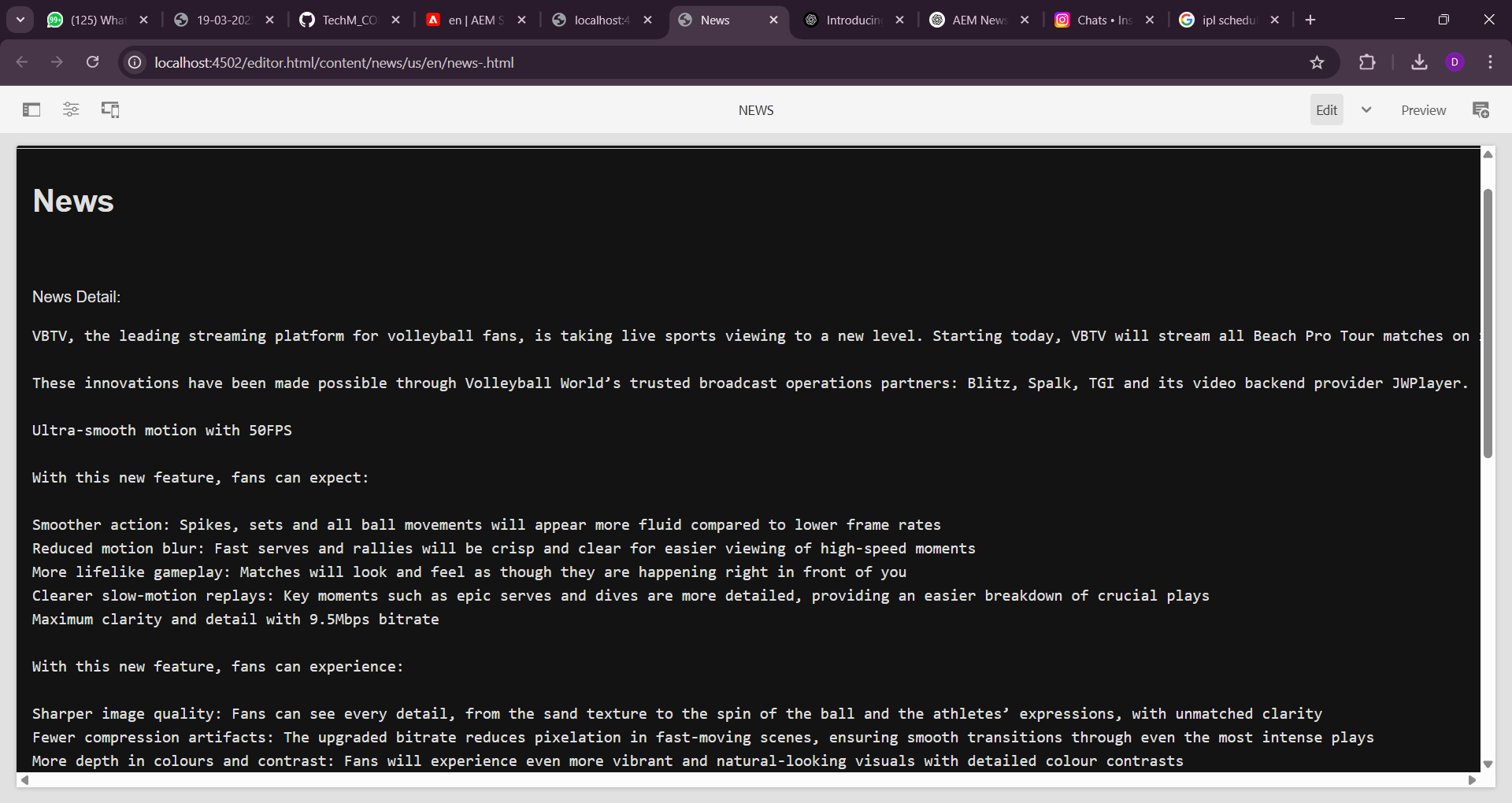
<sly data-sly-call="${clientLib.css @ categories='project.news'}" />

</div>









### ****2. Create a Multi-Field Component Using Sling Model****

You need to create a multi-field component that allows adding multiple news items.

#### ****Sling Model for Multi-Field News Component (MultiNewsComponent.java)****

Path:

C:\Users\LENOVO\Training\aem\codebase\news\core\src\main\java\com\myTraining\core\models

package com.myTraining.core.models;

import com.adobe.cq.sightly.WCMUsePojo;

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

import javax.inject.Inject;

import java.util.ArrayList;

import java.util.List;

import java.util.Optional;

import org.apache.sling.models.annotations.DefaultInjectionStrategy;

import org.apache.sling.models.annotations.Model;

@Model(adaptables = Resource.class, defaultInjectionStrategy = DefaultInjectionStrategy.OPTIONAL)

public class MultiNewsModel {

@Inject

private List<NewsItem> newsItems;

public List<NewsItem> getNewsItems() {

return Optional.ofNullable(newsItems).orElse(new ArrayList<>());

}

public static class NewsItem {

@Inject

private String title;

@Inject

private String source;

public String getTitle() {

return title;

}

public String getSource() {

return source;

}

}

}

**MultiNews.html**

<div class="multi-news">

<sly data-sly-list.newsItem="${news.newsItems}">

<div class="news-item">

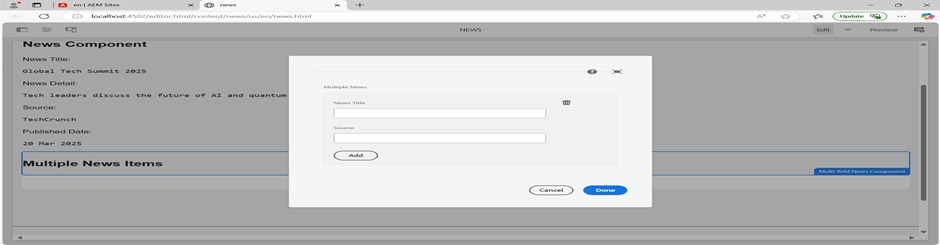
<h2>${newsItem.title}</h2>

<p>${newsItem.source}</p>

</div>

</sly>

</div>



### ****3. Create Clientlibs for the News Component****

#### ****Define Clientlib (****news-clientlib****)****

Create news.css under /apps/project/clientlibs/news/css/news.css

.cop-news-component h2 {

color: green;

}

.cop-news-component p {

color: yellow;

}

.cop-news-component .date {

color: black;

}

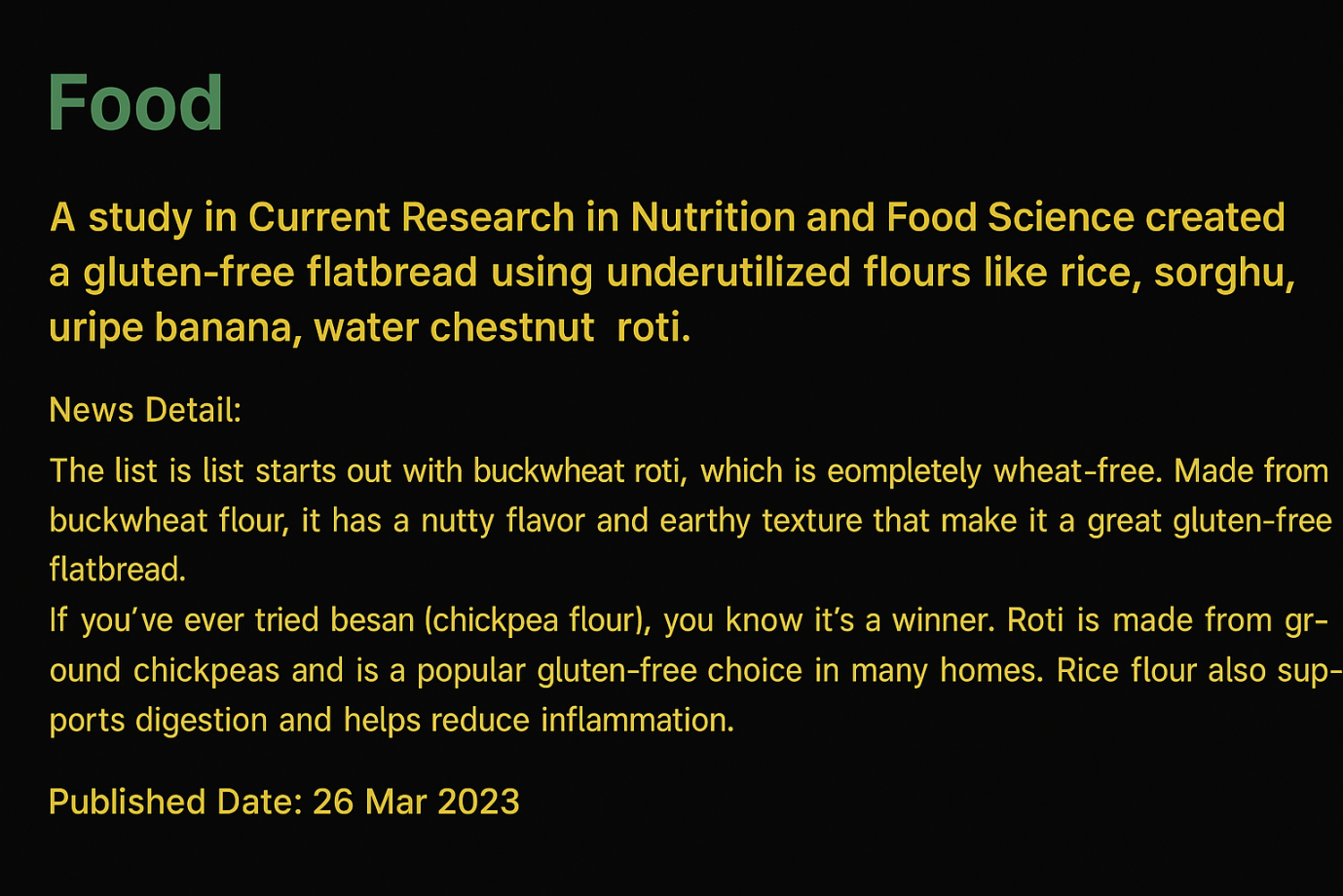
#### ****Add Clientlibs in News Component****

<sly data-sly-use.clientlib="/libs/granite/sightly/templates/clientlib.html" />

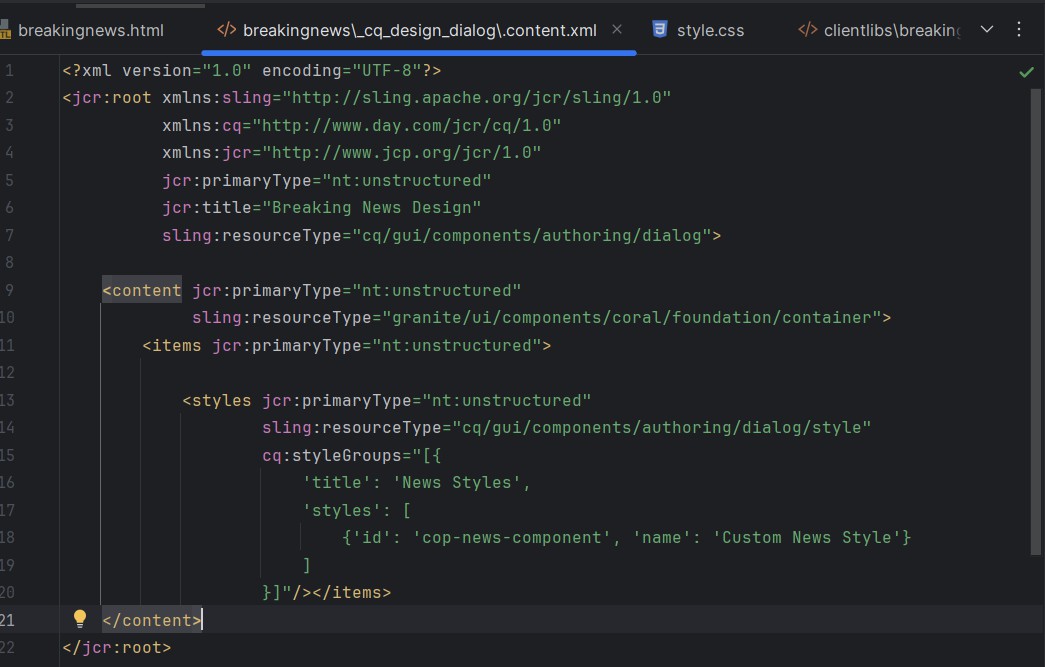
<sly data-sly-call="${clientlib.js @ categories='news-component'}" />

<sly data-sly-call="${clientlib.css @ categories='news-component'}" />

**4.** **Apply Green colour to the heading(h2) and yellow colour to news detail(<p) and date should be black in colour.**



**5. Apply Custom Styling (cop-news-component)**



**6. Create a Base Page Component and Metadata File**

Add a new metadata file to output Open Graph meta tags.

**HTML File (basepage.html)**

<head>

<meta property="og:title" content="${pageProperties['og:title']}">

<meta property="og:description" content="${pageProperties['og:description']}">

<meta property="og:image" content="${pageProperties['og:image']}">

</head>

**7. Create Custom Page Properties (Global Properties)**

**dialog.xml**

<jcr:root xmlns:jcr="<http://www.jcp.org/jcr/1.0>" xmlns:sling="[http://sling.apache.org/jcr/sling/1.0"](http://sling.apache.org/jcr/sling/1.0) jcr:primaryType="cq:Dialog" cq:dialogMode="floating">

<content jcr:primaryType="nt:unstructured">

<items jcr:primaryType="nt:unstructured">

<ogTitle

jcr:primaryType="nt:unstructured"

sling:resourceType="granite/ui/components/coral/foundation/form/textfiel d"

fieldLabel="OG Title" name="./ogTitle"/>

<ogDescription

jcr:primaryType="nt:unstructured"

sling:resourceType="granite/ui/components/coral/foundation/form/textare a"

fieldLabel="OG Description" name="./ogDescription"/>

<ogImage

jcr:primaryType="nt:unstructured"

sling:resourceType="granite/ui/components/coral/foundation/form/textfiel d"

fieldLabel="OG Image" name="./ogImage"/>

</items>

</content>

</jcr:root>

**8. What is ExtraClientlib?**

ExtraClientLibs is a property in AEM that allows additional client libraries (CSS/JS) to be included dynamically for a component without modifying the component's default client libraries.

Why Use extraClientLibs?

* + Helps extend functionality without modifying core component clientlibs.
  + Useful for adding custom JavaScript or styles specific to a component.
  + Allows separation of concerns, keeping core and additional scripts/styles modular.