# AEM – Final Task Newsroom Portfolio Website

**Header Implementation**

* Use the OOTB Header Component to create a navigation bar with:
  + Home
  + News (Dropdown with 5 pages)
  + About Me
  + Contact Me
* Ensure the News menu has a dropdown.
* Implement navigation and styling using:
  + \_cq\_dialog/.content.xml → For navigation items
  + header.html → Structure of the header
  + header.css → Styling
  + header.js → Dropdown functionality

# Code

**Header.html**

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

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

<header class="header-container">

<nav>

<ul class="nav-links">

<li><a href="/content/myTraining/us/en/home.html">Home</a></li>

<li class="dropdown">

<a href="/content/myTraining/us/en/news">News</a>

<ul class="dropdown-menu">

<li><a href="/content/myTraining/us/en/news/live-news.html">Live- News</a></li>

<li><a href="/content/myTraining/us/en/news/sports-news.html">Sports- News</a></li>

<li><a href="/content/myTraining/us/en/news/tech-news.html">Tech- News</a></li>

<li><a href="/content/myTraining/us/en/news/breaking-news.html">Breaking- News</a></li>

<li><a href="/content/myTraining/us/en/news/economic-news.html">Economic- News</a></li>

</ul>

</li>

<li><a href="/content/myTraining/us/en/about-me.html">About Me</a></li>

<li><a href="/content/myTraining/us/en/contact-us.html">Contact Me</a></li>

</ul>

</nav>

</header>

<sly data-sly-call="${clientLib.js @ categories='myTraining.header'}"/>

<sly data-sly-use.newsModel="com.myTraining.core.models.HeaderModel"/>

# Header.js

document.addEventListener("DOMContentLoaded", function () { const dropdownMenus = document.querySelectorAll(".nav-links li");

dropdownMenus.forEach(menu => {

const dropdown = menu.querySelector(".dropdown");

if (dropdown) {

menu.addEventListener("mouseenter", () => { dropdown.style.display = "block";

});

menu.addEventListener("mouseleave", () => { dropdown.style.display = "none";

});

}

});

});

# Hesder.css

**.header-container { display: flex;**

**justify-content: space-between; align-items: center;**

**padding: 15px 30px; background-color: #002b5c;**

**}**

**.nav-links {**

**list-style: none; display: flex; gap: 20px;**

**}**

**.nav-links li {**

**position: relative;**

**}**

**.nav-links a {**

**text-decoration: none; color: white;**

**font-size: 16px; padding: 10px;**

**}**

**.nav-links li:hover .dropdown-menu { display: block;**

**}**

**.dropdown-menu { display: none;**

**position: absolute; top: 100%;**

**left: 0;**

**background-color: white;**

**box-shadow: 0px 4px 6px rgba(0, 0, 0, 0.1); list-style: none;**

**padding: 10px; min-width: 150px;**

**}**

**.dropdown-menu li { padding: 5px 10px;**

**}**

**.dropdown-menu li a {**

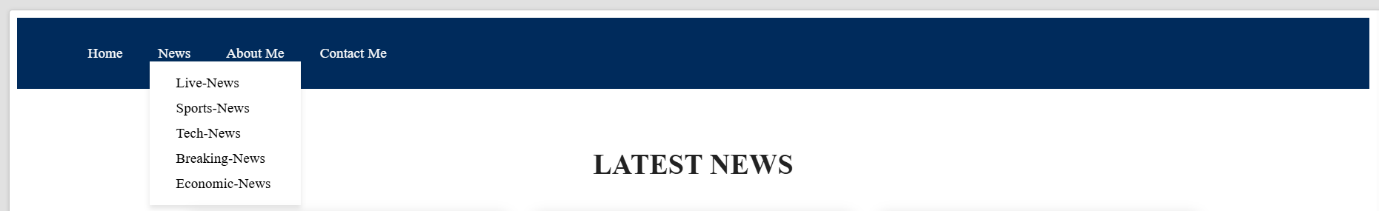
**color: black;**

**}**

**Screenshots Header**



**Header With Dropdown**



**Footer Implementation**

The footer should include the following sections:

1. About Me
2. Contact Me
3. Latest 5 News (with hyperlinks opening in a new tab)
4. Social Media Links To achieve this, we will:
   * Use an Experience Fragment (/content/experience-

fragments/myTraining/footer) to create a reusable footer.

* + Author the footer once and use it across multiple pages and templates.

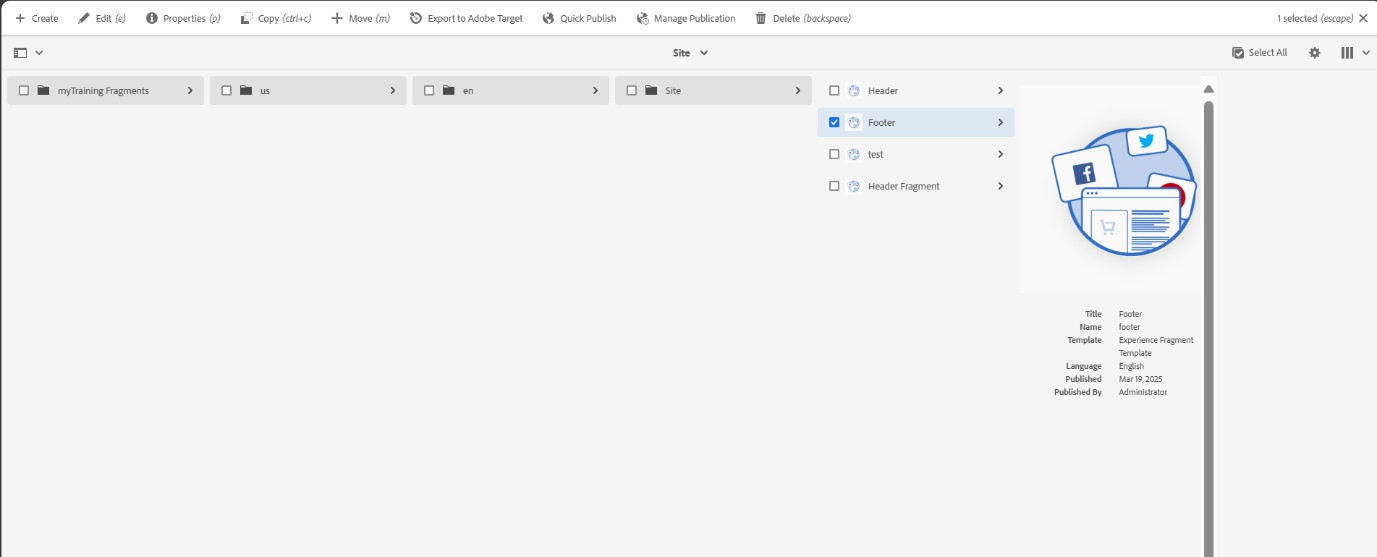
1. Creating the Footer Experience Fragment
   1. Navigate to AEM > Experience Fragments.
   2. Click on Create > Experience Fragment.
   3. Provide a title (e.g., Footer) and select a Master Variation.
   4. Choose the Web Variation Template.
   5. Click Create & Open.
   6. Edit the fragment by adding the required components:
      * Text Component for "About Me"
      * Text Component for "Contact Me"
      * List Component for latest 5 news
      * Teaser Component for social media links
   7. Click Save & Close.
2. Configuring Latest 5 News Using a List Component To dynamically fetch the latest 5 news items:
   1. Add a List Component to the Experience Fragment.
   2. Open the List Component’s properties.
   3. Configure it to fetch content from /content/news.
   4. Set Limit to 5 items.
   5. Check Open links in a new tab.
   6. Save the configuration.

# Screenshots

* **Footer in pages**

****

**Footer Experience Fragment**



****

# Home Page Implementation

The home page should include:

1. A Teaser Component to display an award ceremony for a journalist.
2. The teaser should have:
   * An Image
   * A Title
   * A Description
   * A Call-to-Action (CTA) linking to the full news page. To achieve this, we will:

* Use the OOTB Content Page Template

(/conf/myTraining/settings/wcm/templates/content-page).

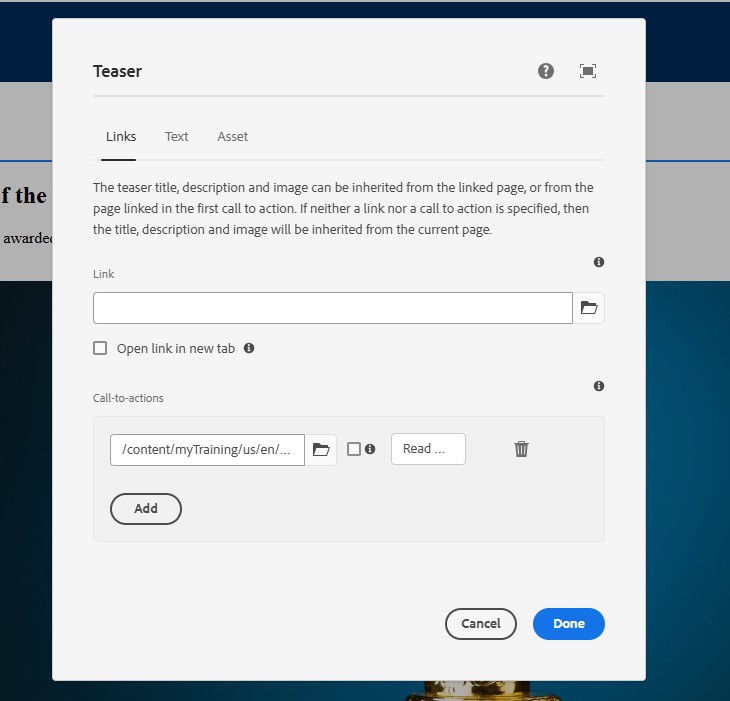
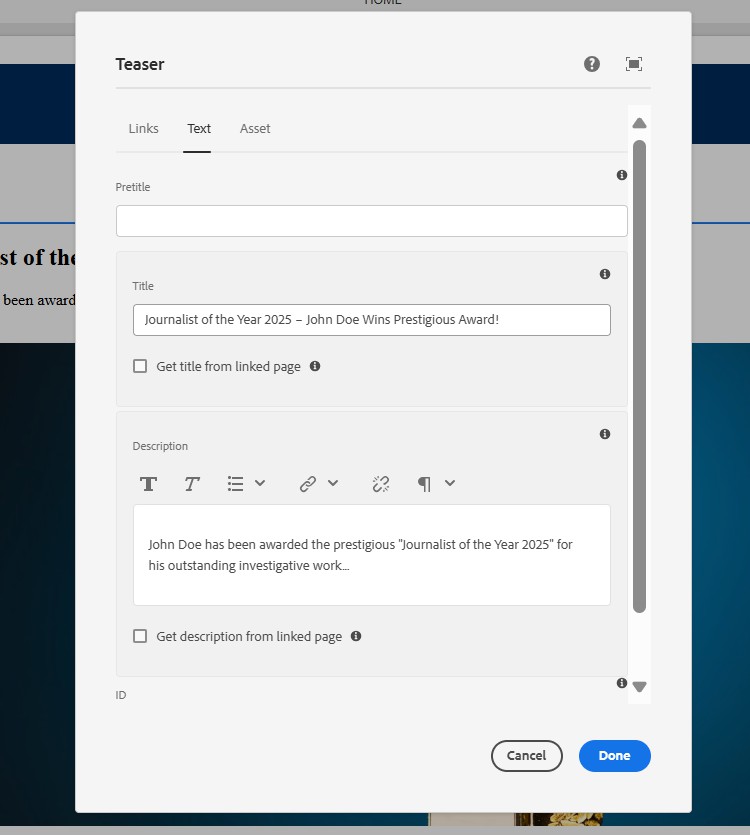
* Author the Teaser Component with award details.
* Ensure that the CTA links to a separate news article page containing full details.

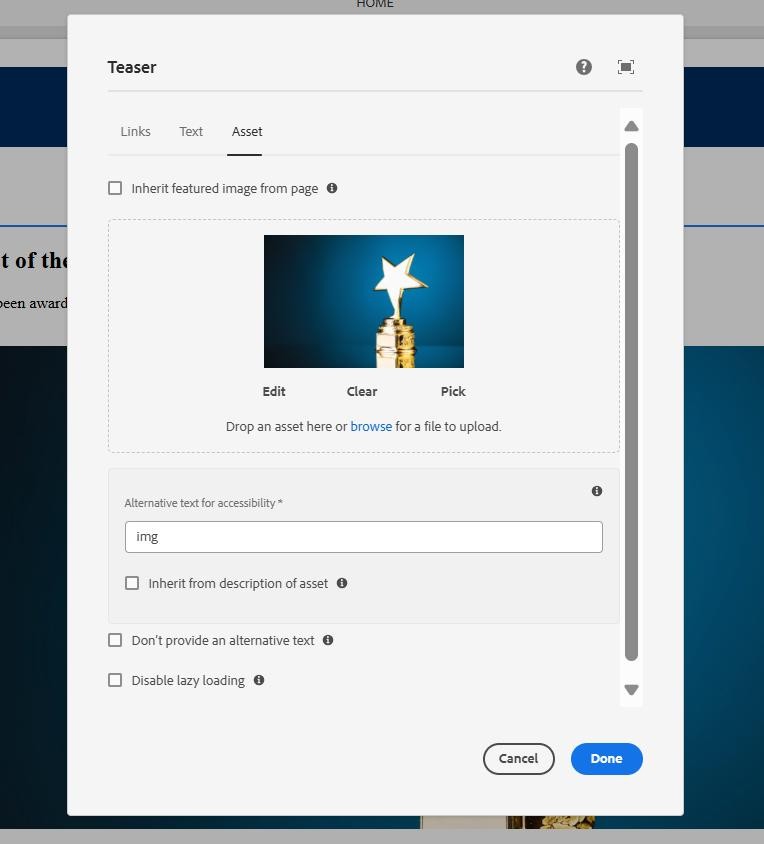
1. Creating the Home Page
   1. Navigate to AEM > Sites.
   2. Click on Create > Page.
   3. Select Content Page (OOTB Template).
   4. Provide a title (e.g., Home).
   5. Click Create & Open.
2. Configuring the Teaser Component
   1. Open the Home Page in the AEM editor.
   2. Drag and drop a Teaser Component onto the page.
   3. Configure the teaser:
      * Image: Upload an image of the journalist’s award ceremony.
      * Title: Enter a meaningful title (e.g., *Journalist Wins Prestigious Award*).
      * Description: Provide a short summary of the award.
      * CTA: Link to the detailed news page (/content/news/award- ceremony).
   4. Click Save & Close.
3. Linking the Teaser to a Detailed News Page

* The CTA Button links to /content/news/award-ceremony.
* You must author a news article page under /content/news/award- ceremony.
* That page should include:
  + A Title Component for the award’s headline.
  + A Text Component for full news details.
  + An Image Component with relevant images.

# Screenshots

* **Adding contents in teaser**

****



* **Home Page**



****

# News Page Implementation

The News Page should:

1. Be created under /content/news.
2. Use a custom NewsRoom Component to fetch all authored news articles from /content/news.
3. Display news articles in a card layout (each card should contain an image, title, and short description).
4. Ensure that clicking on a news card redirects the user to the respective news article page.

Each News Article Page should:

1. Be created under /content/news/{article-title}.
2. Include a Title Component for the article headline.
3. Use an Image Component for the news image.
4. Use a Text Component for the article content.
5. Creating the News Page
   1. Navigate to AEM > Sites.
   2. Click on Create > Page.
   3. Select Content Page (OOTB Template).
   4. Provide a title (News).
   5. Click Create & Open.
6. Creating the Custom NewsRoom Component

# Newsroom.html

<sly data-sly-use.newsroomModel="com.myTraining.core.models.NewsRoomModel"/>

<div class="newsroom">

<h2>Latest News</h2>

<ul>

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

<li class="news-card">

<img src="${newsItem.image}" alt="${newsItem.title}">

<h3>${newsItem.title}</h3>

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

<a href="${newsItem.pagePath}.html">Read More</a>

</li>

</sly>

</ul>

</div>

# Newsroom.css

.newsroom-container { text-align: center; padding: 50px;

}

.news-list { display: flex;

flex-wrap: wrap;

justify-content: center; gap: 20px;

}

.news-card { width: 300px;

background: white; padding: 15px;

border-radius: 10px;

box-shadow: 0px 4px 6px rgba(0, 0, 0, 0.1); text-align: left;

}

.news-card img { width: 100%;

border-radius: 10px;

}

.news-card h3 { font-size: 20px; margin: 10px 0;

}

.news-card p {

font-size: 14px; color: #555;

}

.read-more {

display: inline-block; margin-top: 10px; padding: 8px 12px;

background: #007BFF; color: white;

text-decoration: none; border-radius: 5px;

}

.read-more:hover {

background: #0056b3;

}

# Xml file

<?xml version="1.0" encoding="UTF-8"?>

<jcr:root xmlns:sling="<http://sling.apache.org/jcr/sling/1.0>" xmlns:cq[="http://www.day.com/jcr/cq/1.0](http://www.day.com/jcr/cq/1.0)"

xmlns:jcr="[http://www.jcp.org/jcr/1.0"](http://www.jcp.org/jcr/1.0) jcr:primaryType="cq:Dialog"

jcr:title="NewsRoom Component"

sling:resourceType="cq/gui/components/authoring/dialog">

<content jcr:primaryType="nt:unstructured"

sling:resourceType="granite/ui/components/coral/foundation/container">

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

<newsPath jcr:primaryType="nt:unstructured"

sling:resourceType="granite/ui/components/coral/foundation/form/pathfield" fieldLabel="News Parent Path"

name="./newsPath"/>

</items>

</content>

</jcr:root>

# NewsRoomModel.java

package com.myTraining.core.models;

import com.adobe.cq.wcm.core.components.models.ListItem;

import com.adobe.cq.wcm.core.components.util.AbstractComponentImpl; import org.apache.sling.api.resource.Resource;

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

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

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

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

import javax.annotation.PostConstruct; import java.util.ArrayList;

import java.util.List;

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

public class NewsRoomModel extends AbstractComponentImpl {

@Self

private Resource resource;

@ValueMapValue

private String newsPath;

private List<NewsItem> newsItems; @PostConstruct

protected void init() {

newsItems = new ArrayList<>();

if (newsPath != null && !newsPath.isEmpty()) {

ResourceResolver resolver = resource.getResourceResolver(); Resource newsRoot = resolver.getResource(newsPath);

if (newsRoot != null) {

for (Resource newsResource : newsRoot.getChildren()) {

String title = newsResource.getValueMap().get("jcr:title", "Untitled News"); String image = newsResource.getValueMap().get("image", "");

String description = newsResource.getValueMap().get("text", "No description available.");

newsItems.add(new NewsItem(title, image, description, newsResource.getPath()));

}

}

}

}

public List<NewsItem> getNewsItems() { return newsItems;

}

public static class NewsItem { private final String title;

private final String image;

private final String description; private final String pagePath;

public NewsItem(String title, String image, String description, String pagePath) { this.title = title;

this.image = image; this.description = description; this.pagePath = pagePath;

}

public String getTitle() { return title; } public String getImage() { return image; }

public String getDescription() { return description; } public String getPagePath() { return pagePath; }

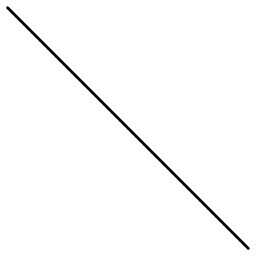
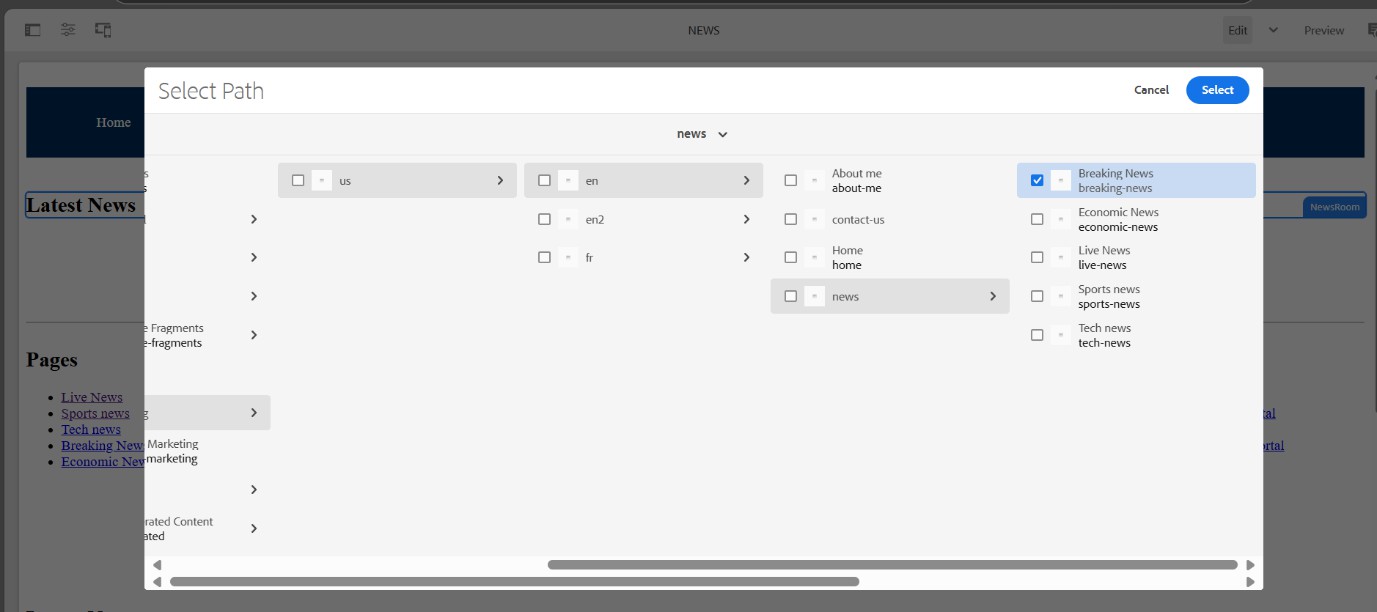
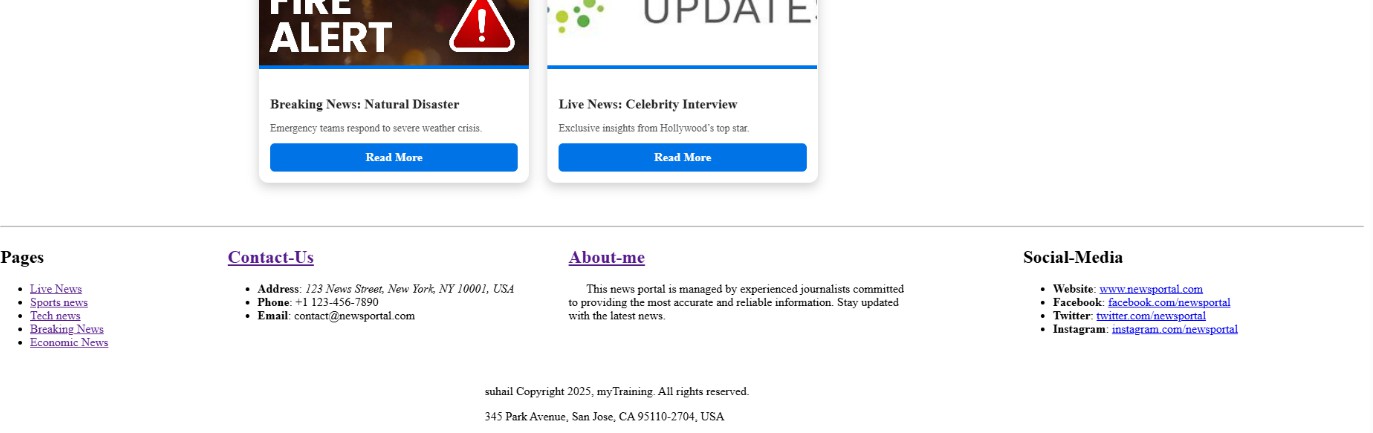
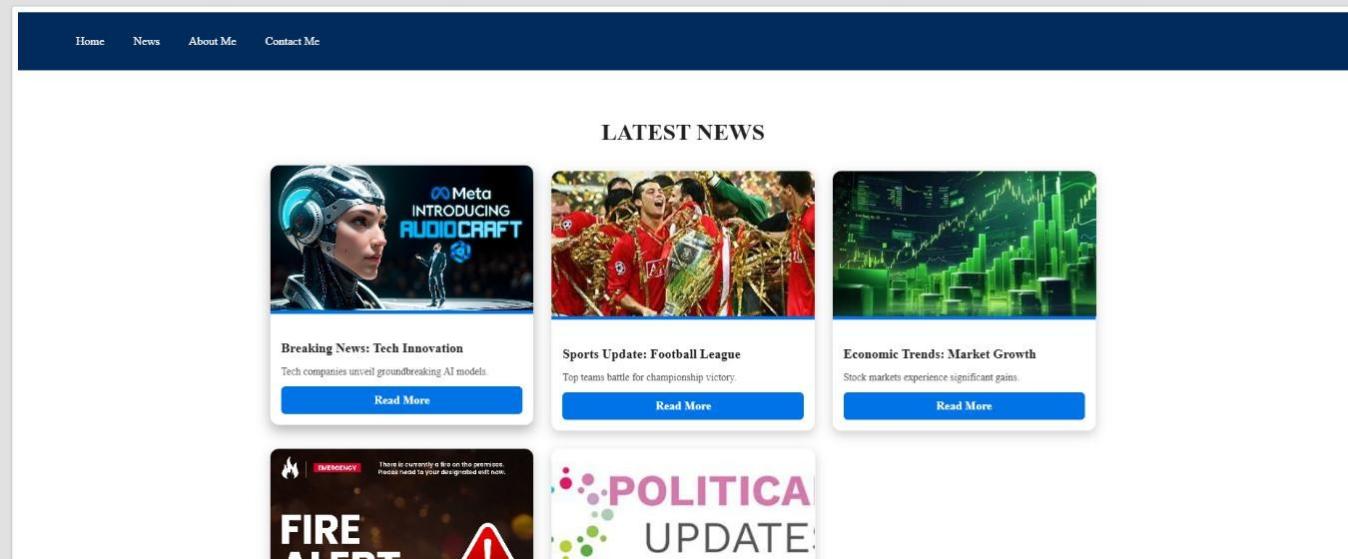
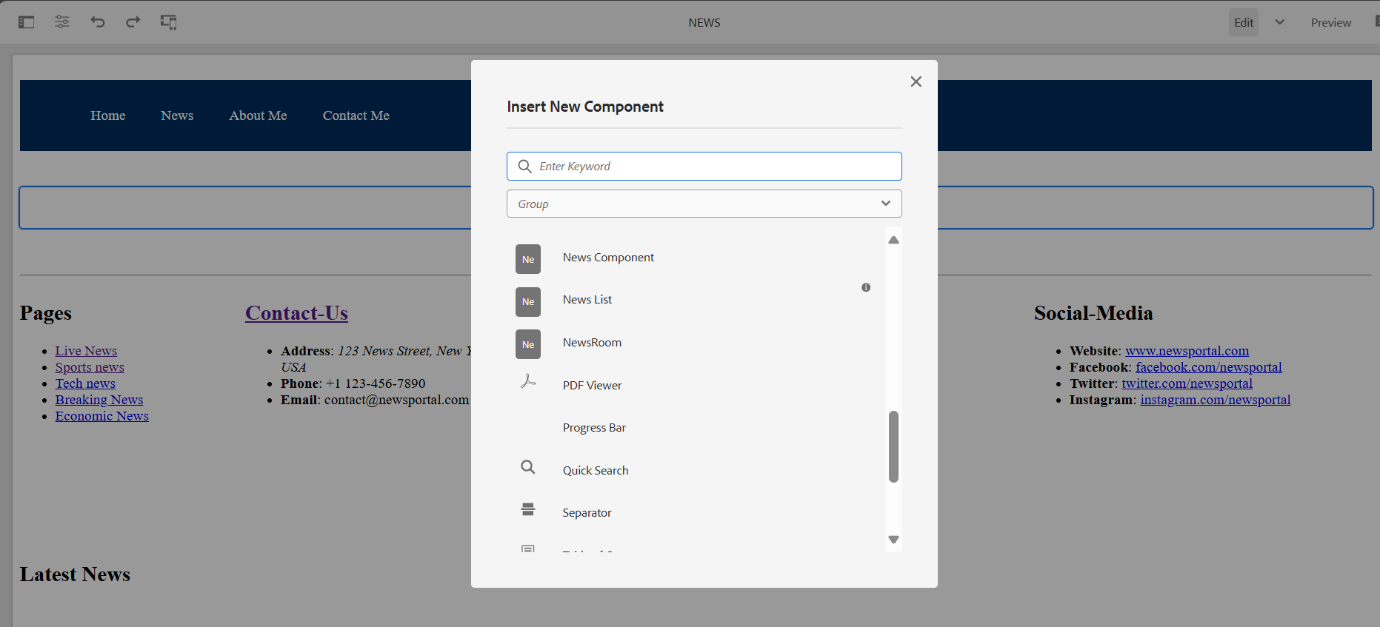
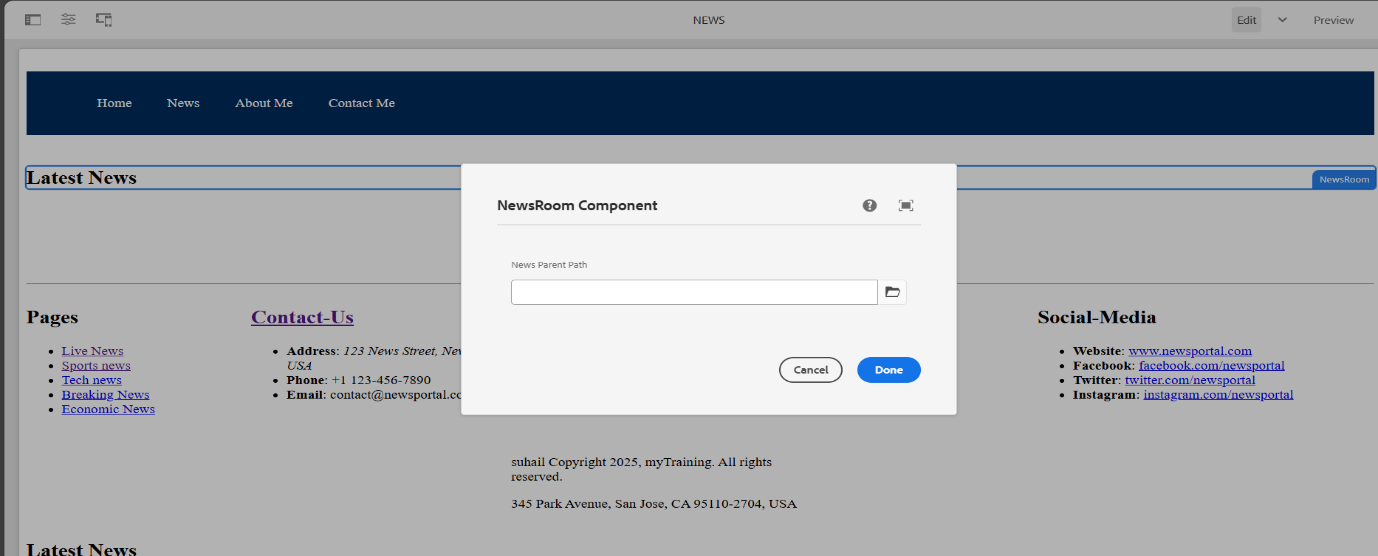
}

}

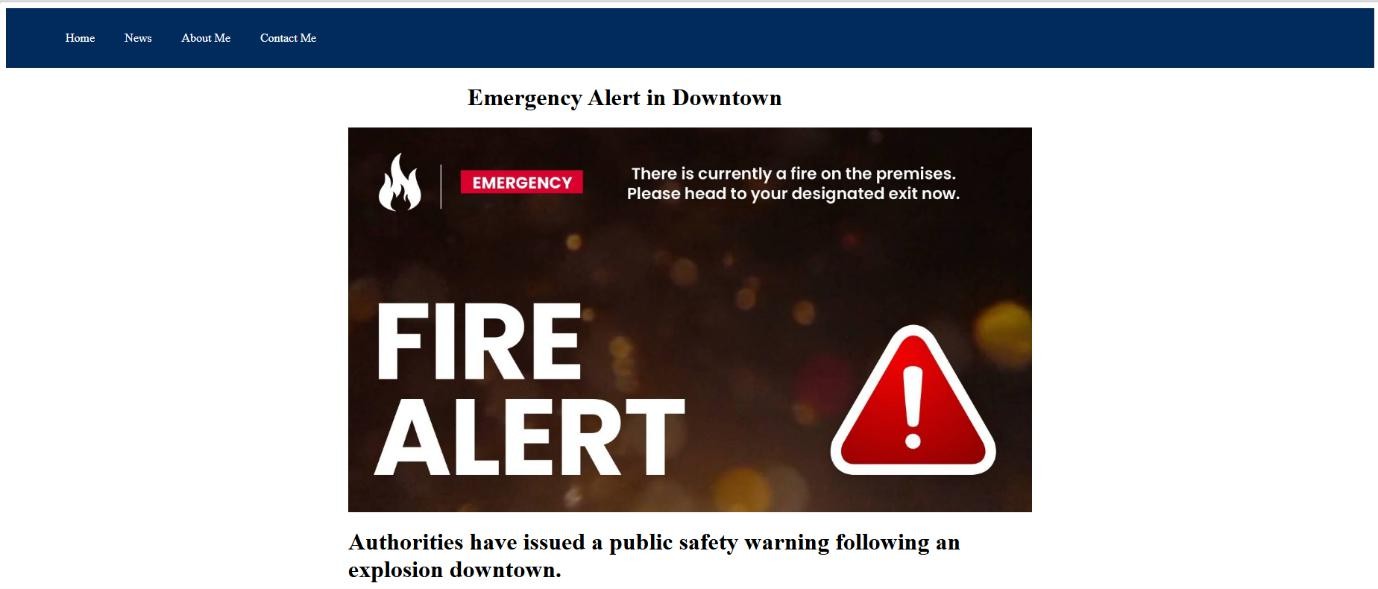
1. Creating News Article Pages
   1. Navigate to AEM > Sites > News.
   2. Click Create > Page.
   3. Select Content Page (OOTB Template).
   4. Provide a title (e.g., "Breaking News - Major Event").
   5. Click Create & Open.
   6. Edit the page by adding:
      * Title Component for the headline.
      * Image Component for the article image.
      * Text Component for the news content.
   7. Click Save & Publish.

# Screenshots

* **NewsRoom component**

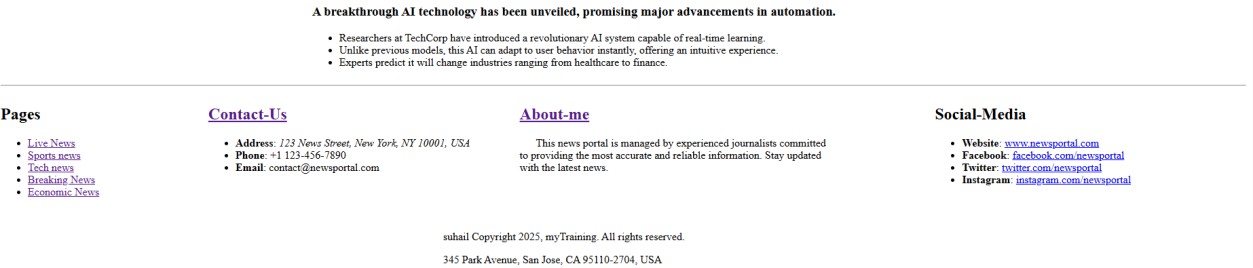
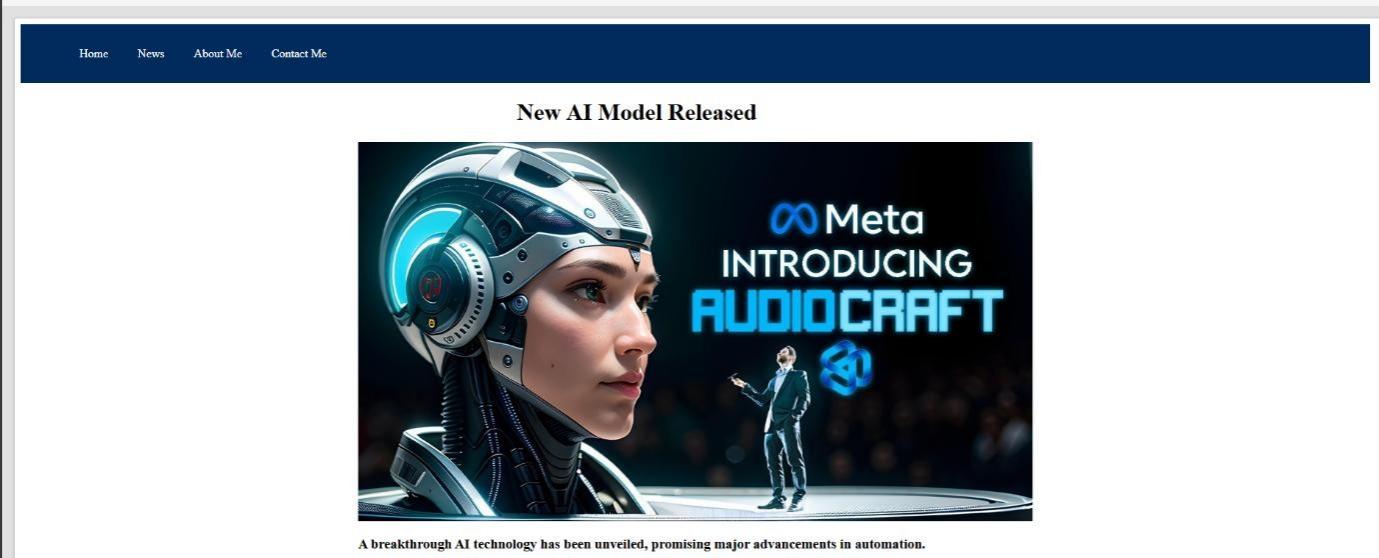


* **NewsPage**
* **Page - 01**

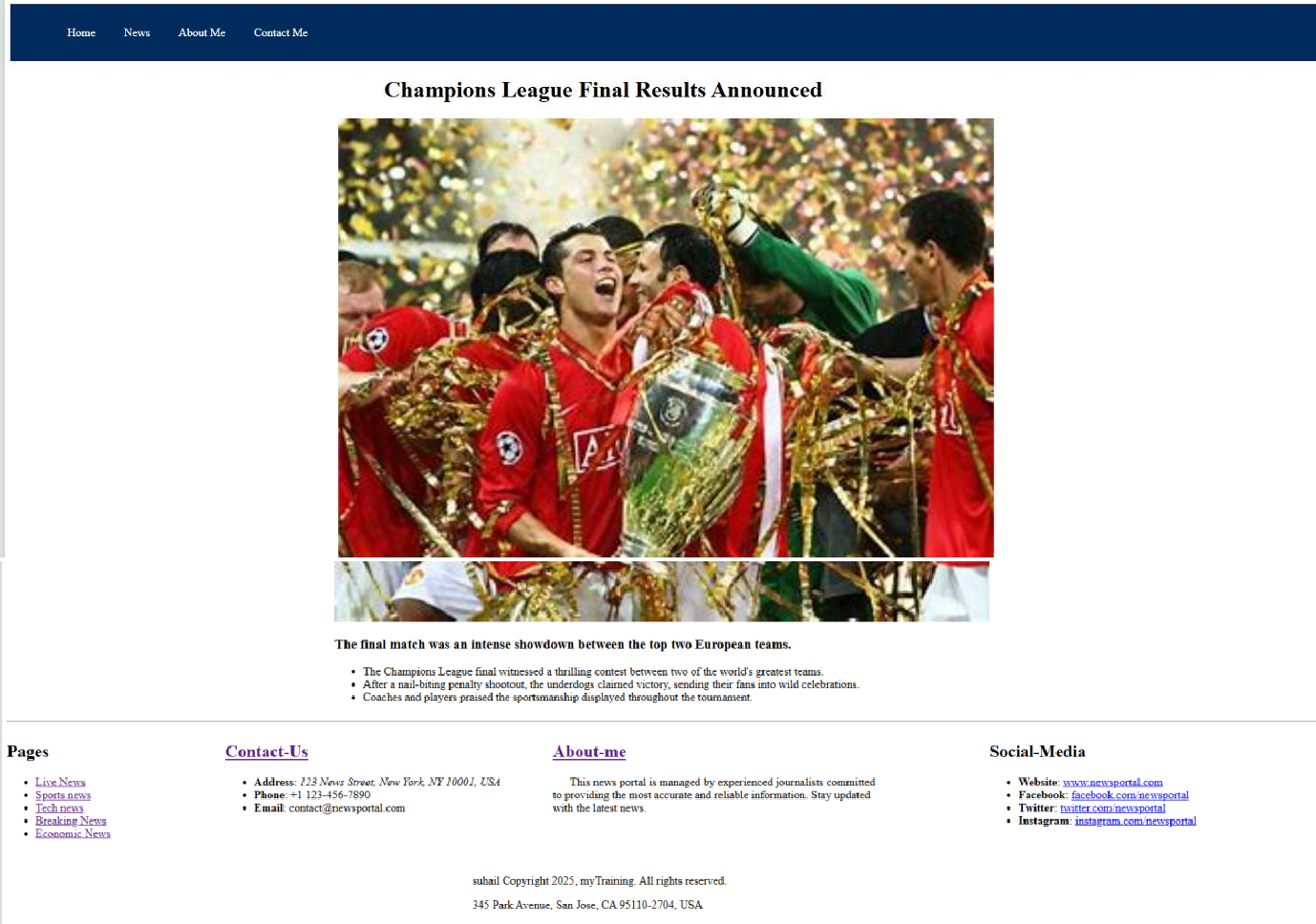


**Page – 02**

**Page – 03**

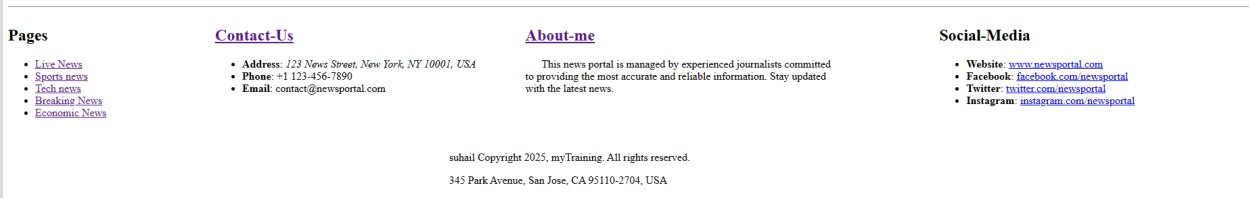


**Page – 04**



**Page – 05**



****

# About Me Page

The About Me Page should:

1. Be created under /content/about-me.
2. Use a custom Journalist Component to display journalist information.
3. Include:
   * Journalist’s Image
   * Name & Title
   * Biography / Description
   * Notable Achievements
4. Creating the About Me Page
   1. Navigate to AEM > Sites.
   2. Click on Create > Page.
   3. Select Content Page (OOTB Template).
   4. Provide a title (About Me).
   5. Click Create & Open.
5. Creating the Custom Journalist Component

# Code

* **Journalist.html**

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

<sly data-sly-call="${clientlib.all @ categories='myTraining.journalist'}" />

<sly data-sly-use.journalist="com.myTraining.core.models.JournalistModel"/>

<div class="journalist">

<div class="journalist image" data-sly-test="${journalist.image}">

<img src="${journalist.image}" alt="${journalist.name}" />

</div>

<div class="journalist content">

<h2 data-sly-test="${journalist.name}">${journalist.name}</h2>

<p data-sly-test="${journalist.bio}">${journalist.bio}</p>

</div>

</div>

# Xml file

<?xml version="1.0" encoding="UTF-8"?>

<jcr:root

xmlns:sling[="http://www.sling.apache.org/jcr/sling/1.0](http://www.sling.apache.org/jcr/sling/1.0)" xmlns:cq[="http://www.day.com/jcr/cq/1.0](http://www.day.com/jcr/cq/1.0)"

xmlns:jcr[="http://www.jcp.org/jcr/1.0"](http://www.jcp.org/jcr/1.0) jcr:primaryType="cq:Dialog"

cq:dialogMode="floating"

jcr:title="Journalist Properties">

<content jcr:primaryType="nt:unstructured"

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

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

<name jcr:primaryType="nt:unstructured"

sling:resourceType="granite/ui/components/coral/foundation/form/textfield" fieldLabel="Journalist Name"

name="./name"/>

<image jcr:primaryType="nt:unstructured"

sling:resourceType="granite/ui/components/coral/foundation/form/pathfield" fieldLabel="Journalist Image"

name="./image"

rootPath="/content/dam"/>

<bio jcr:primaryType="nt:unstructured"

sling:resourceType="granite/ui/components/coral/foundation/form/textarea" fieldLabel="Journalist Bio"

name="./bio"/>

</items>

</content>

</jcr:root>

**JournalistModel.java**

package com.myTraining.core.models;

import org.apache.sling.api.resource.Resource; import javax.inject.Inject;

import javax.inject.Named;

import com.adobe.cq.export.json.ExporterConstants; import org.apache.sling.models.annotations.Default;

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

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

@Model(adaptables = Resource.class,

defaultInjectionStrategy = DefaultInjectionStrategy.*OPTIONAL*) @Exporter(name = ExporterConstants.*SLING\_MODEL\_EXPORTER\_NAME*,

extensions = ExporterConstants.*SLING\_MODEL\_EXTENSION*, selector = "journalist") // Added selector

public class JournalistModel {

@Inject @Named("name")

@Default(values = "Unknown Journalist") private String name;

@Inject @Named("image")

@Default(values = "/content/dam/myTraining/us/en-us/professional.png") private String image;

@Inject @Named("bio")

@Default(values = "No bio available.")

private String bio;

public String getName() { return name;

}

public String getImage() { return image;

}

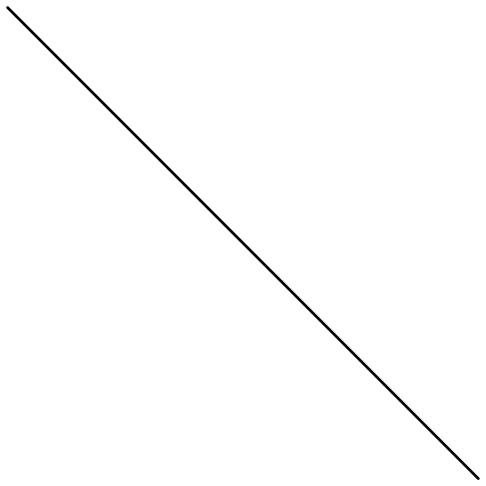
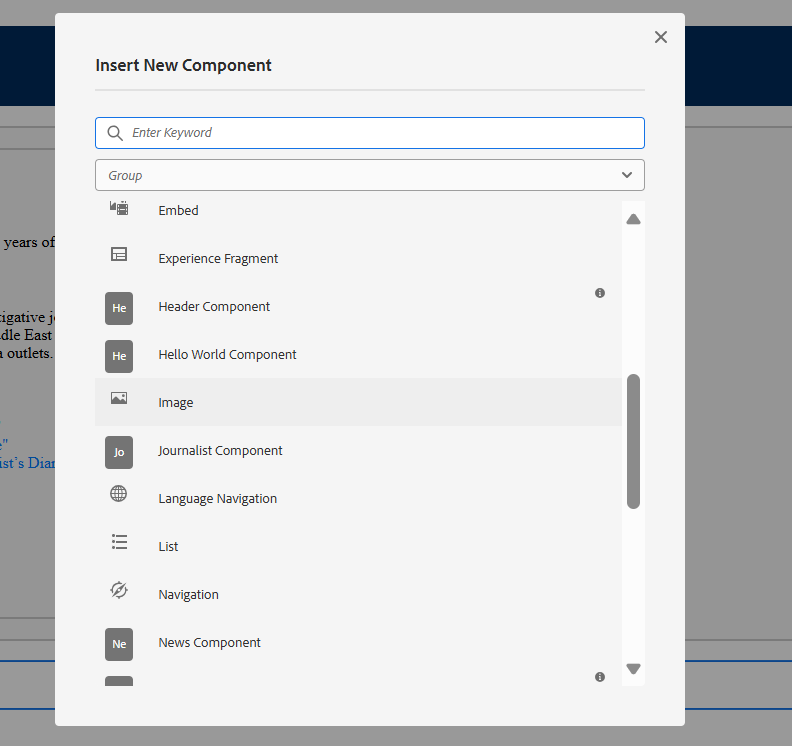
public String getBio() { return bio;

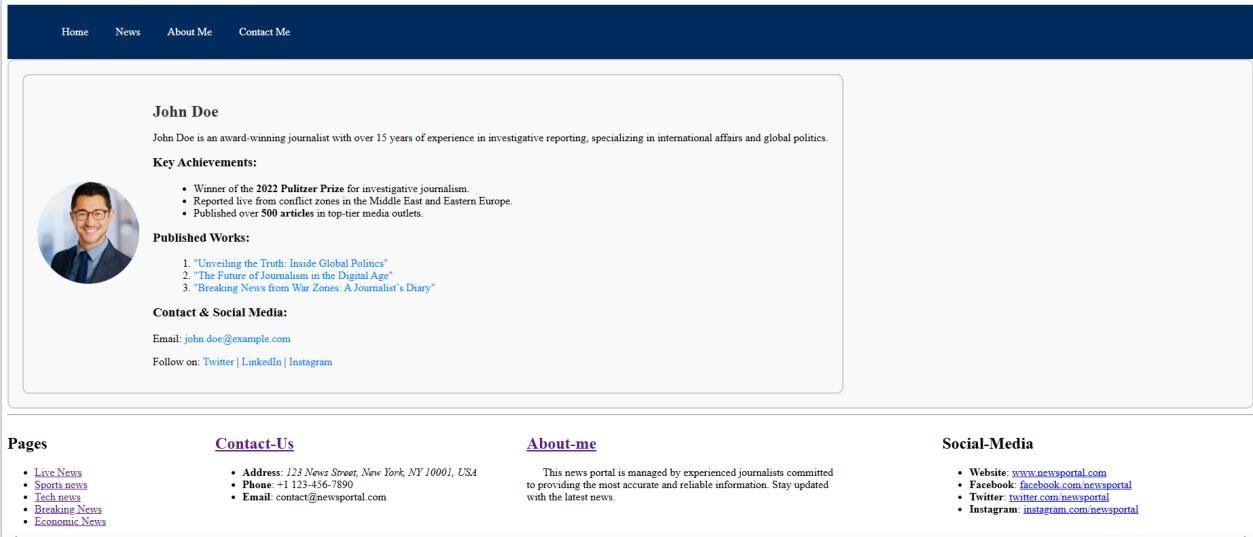
}

}

1. Adding the Journalist Component to the About Me Page
   1. Open the About Me Page in the AEM editor.
   2. Drag and drop the Journalist Component onto the page.
   3. Configure the component by filling in:
      * Image: Upload a journalist’s profile image.
      * Name: Enter the journalist’s name.
      * Title: Provide a short designation (e.g., *Investigative Journalist*).
      * Biography: Enter a short bio or achievements.
   4. Click Save & Close.

# Screenshots



****

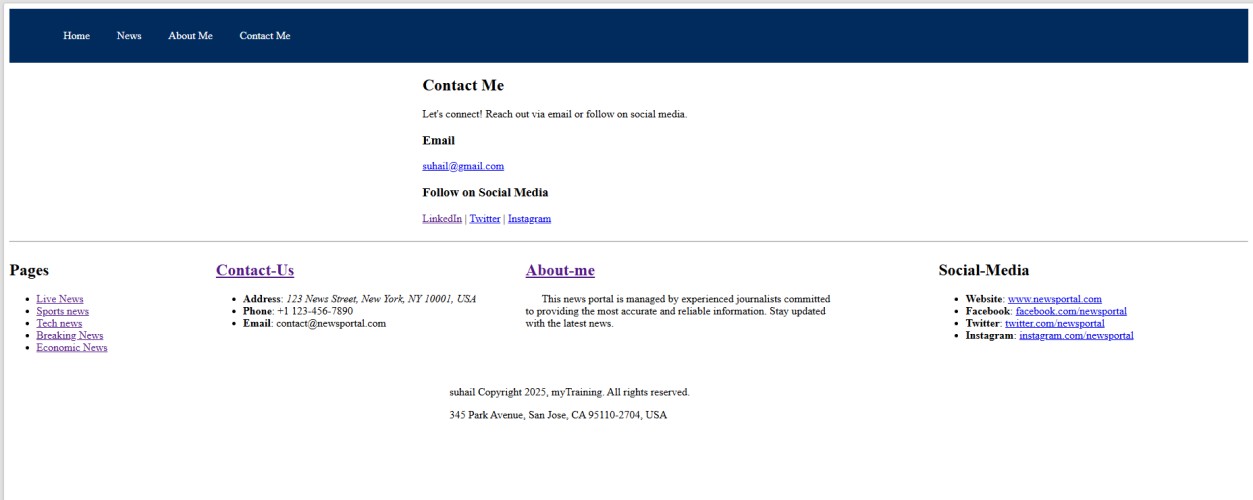
**Contact Me Page**

The Contact Me Page should:

1. Be created under /content/contact-me.
2. Use Text Component or Teaser Component for:
   * Journalist’s Contact Information (Email, Phone)
   * Social Media Links (LinkedIn, Twitter, etc.)
   * Optional Call-to-Action (CTA) for Engagement
3. Creating the Contact Me Page
   1. Navigate to AEM > Sites.
   2. Click on Create > Page.
   3. Select Content Page (OOTB Template).
   4. Provide a title (Contact Me).
   5. Click Create & Open.
4. Adding Contact Details Using Text Component
   1. Open the Contact Me Page in the AEM editor.
   2. Drag and drop a Text Component onto the page.
   3. Enter the contact details:
      * Email: [suhail@example.com](mailto:suhail@example.com)
      * Phone: +1-234-567-890
      * Location: New York, USA
   4. Click Save & Close.
5. Adding Social Media Links Using Teaser Component
   1. Drag and drop a Teaser Component onto the page.
   2. Configure the component with:

* Title: *Follow Me on Social Media*
* Image: Add a social media banner or icon
* CTA Button: *Connect* (with a link to social media profiles)
* Links:
  + LinkedIn: <https://linkedin.com/in/journalist>
  + Twitter: <https://twitter.com/journalist>
  + Instagram: <https://instagram.com/journalist>
  1. Click Save & Close.

**Screenshot**

****