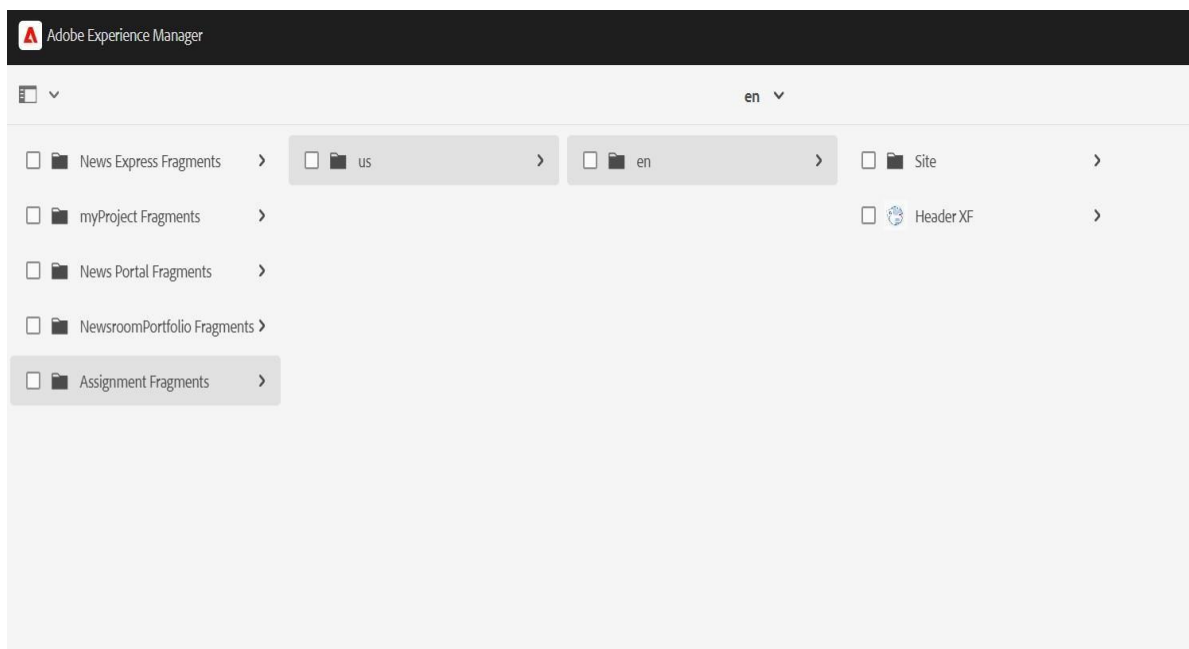1. 1. Create 5 news article pages under /content/us/en/news

Create Header Experience fragment for header and use these page as menu (name should be news) and apart from hat create contact us page and about me page (you can use teaser or image, text, title components to provide some details about journalist on about me page and on contact us page it should be some contact related details it can have their mobile number office address or email address).

1. **News Menu** – Links to the 5 news article pages you created.

2. **Contact Us Page** – With office address, email, or mobile number.

3. **About Me Page** – With journalist details using Teaser, Image, Text, and Title components.

Create footer XF and it could have 4 sections:

1. News Menu Section

2. About Me Section

3. Contact Us Section

4. Social Media Section

Implementation Steps

Step 3.1: Create the

Footer XF

1. Go to:

   o AEM Navigation Panel → Experience Fragments.

2. Click: Create → Experience Fragment.

3. Enter the Title: "Footer XF"

4. Select a Template: oChoose a Footer Template or a Blank Template.

5. Click Create → Open.

Step 3.2: Add Sections to Footer XF

1. **News Menu Section:**

   o Add a List Component.

   o Configure it to display 4 news articles from /content/us/en/news.

2. **About Me Section:**

   o Add a Text Component.

   o Enter a short bio about the journalist.

3. **Contact Us Section**: o**Add a Text Component**. oEnter details like: Office

Address: 123 News Street, City

Email: journalist@example.com

Phone: +1 234 567 890

4. **Social Media Section**:

   o Add a List Component.

   o Configure it with links to social media accounts (e.g., LinkedIn, Twitter)

4. 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" in Logs

Overview

We will create an OSGi Service in AEM that:

1. Returns the string "Hello World".

2. Calls this service from the News Component's Sling Model.

3. Prints the "Hello World" message in the AEM logs.

Implementation Steps

Step 4.1: Create the OSGi Service

1. Navigate to your AEM project in IntelliJ IDEA.

2. Go to the core module → assignment/core/src/main/java/com/assignment/services.

3. Create an Interface HelloWorldService.java:

File Path:

assignment/core/src/main/java/com/assignment/services/HelloWorldS

ervice.j ava package com.assignment.services; public interface HelloWorldService

{

   String getMessage();

}

Step 4.2: Implement the OSGi Service

 File Path:

assignment/core/src/main/java/com/assignment/services/impl/Hello

WorldSer viceImpl.java package com.assignment.services.impl;

import com.assignment.services.HelloWorldService;

import

org.osgi.service.component.annotations.Componen

t; import org.slf4j.Logger; import

org.slf4j.LoggerFactory;

@Component(service = HelloWorldService.class, immediate =

true) public class HelloWorldServiceImpl implements

HelloWorldService {

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

   @Override

public String

getMessage() {

     String message = "Hello World";

```
    LOG.info("Custom Service Output: {}",
message);      return message;
  }
}
```

Explanation:

@Component → Registers this as an OSGi service.

implements HelloWorldService → Implements the interface.

LOG.info → Prints "Hello World" in logs.

getMessage() → Returns "Hello World".

Step 4.3: Call the Service from the News Component Sling Model

1. **Go to**:
   assignment/core/src/main/java/com/assignment/models/NewsModel
   .ja va

2. **Modify the NewsModel to Use the Service**:

```
package com.assignment.models;


import com.assignment.services.HelloWorldService; import

com.adobe.cq.sightly.WCMUsePojo; 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.OSGiServic

e; import org.slf4j.Logger; import org.slf4j.LoggerFactory;
```

```java
import javax.inject.Inject;


@Model(adaptables = WCMUsePojo.class,
defaultInjectionStrategy =
DefaultInjectionStrategy.OPTIONAL) public class
NewsModel {


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


    @OSGiService     private
HelloWorldService helloWorldService;


    public String getHelloMessage() {
        String message = helloWorldService.getMessage();
        LOG.info("NewsModel: Hello World Message = {}", message);
        return message;
    }
}
```

Explanation:
@OSGiService → Injects the HelloWorldService.
getHelloMessage() → Calls the service and logs the
output.

Step 4.4: Deploy and Test

1. **Build & Deploy the Project**:

    mvn clean install -PautoInstallPackage

2. **Open AEM Logs**: ○ In **AEM Dev Console**, run:

        tail -f logs/error.log

- Look for:

    Custom Service Output: Hello World

    NewsModel: Hello World Message = Hello World