ONC SITE DROPDOWN MENU COMPONENT

Design Document

Abstract

Design document for the page header component used in ONC SPO sites for all the modern style pages, to be consistent with the current site.

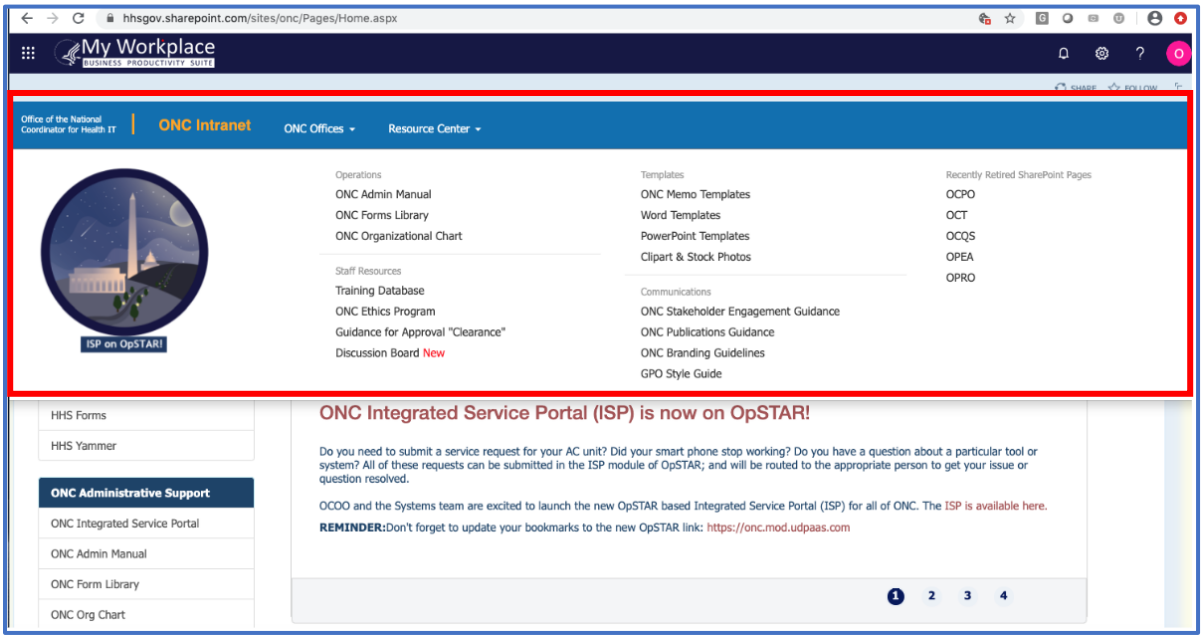
ONC Site Administrator

farrah.darbouze@hhs.gov

TABLE OF CONTENTS

Goal

In ONC SPO site, we have a dropdown menu on top of all the pages (as highlighted below):



This mega dropdown menu is implemented inside the site master page.

Recently when we are going through our site redesign, we want to use the more appealing modern style pages made available by Microsoft. However, master page or layout pages are currently no available in the modern style sites.

Our goal is to implement a new header dropdown menu component that utilizes the new SharePoint Framework to display the current ONC dropdown menu on all the modern style pages.

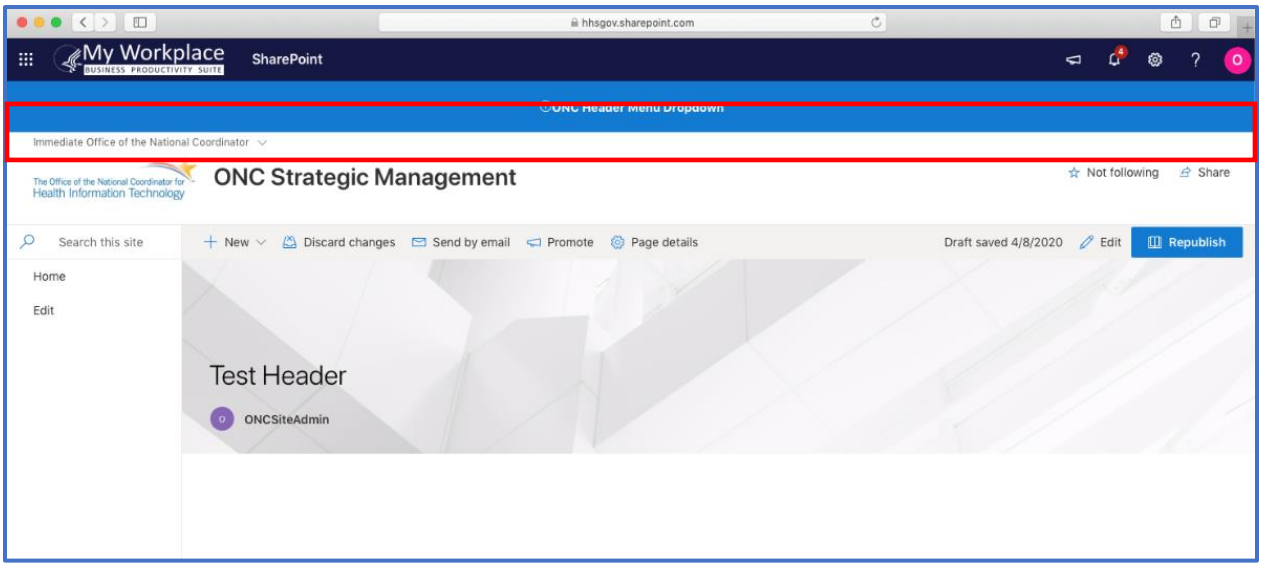
Design

SharePoint Framework Extensions allows user to use the familiar SharePoint Framework tools and libraries for client-side development to extend the SharePoint the modern pages and document libraries. Specifically, SharePoint Framework Extensions type of Application Customizers, allows user to customize the header (PlaceholderName .Top) and footer (PlaceholderName .Bottom) of all modern pages in the site with the extensions application applied to.

In this design, we choose to use Application Customizers load:

1. custom scripts: handle dropdown menu behaviors;
2. third-party JavaScript libraries: include jQuery and bootstrap;
3. styles: include bootstrap, fontawesome, and inline custom CSS;
4. and, custom HTML elements: display the dropdown menu contents.

And render the content in the placeholder “PlaceholderName .Top”:



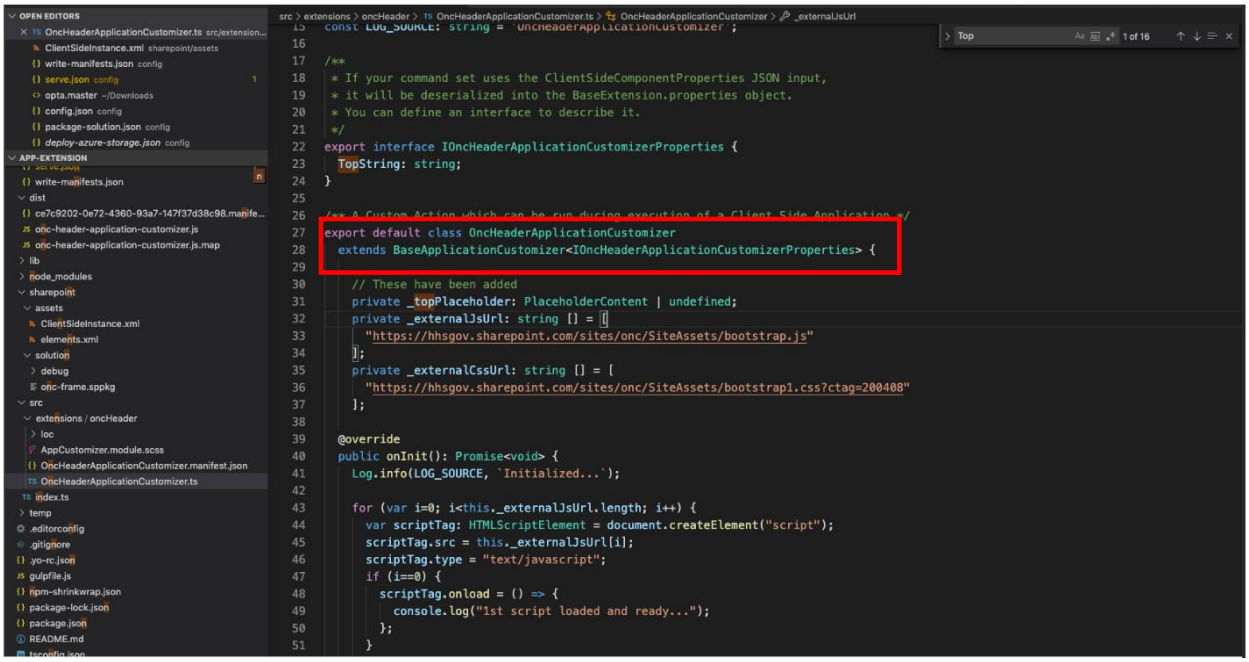
Lastly, Application Customizer extensions are supported with Site, Web, and List scopes. And we are planning to use it on the site level by associating the application customizer to the UserCustomAction collection in Site object level.

REFERNCE: https://docs.microsoft.com/en-us/sharepoint/dev/spfx/extensions/get-started/using-page-placeholder-with-extensions

Implementation Plan

The implementation plan is:

1. Extract and assemble all necessary code, library, css and HTMLs from the current site for the dropdown menu so that we can have an independent component ready to be injected to the new site;
2. Create the SharePoint Framework Extension project for Application Customizer;
3. Extends the BaseApplicationCustomizer class to handle the Header placeholder rendering, and injects all the artifacts prepared in step #1;



4. Test the application;
5. Deploy the application.

NOTE: Per the Microsoft document online, to deploy the client-side application needs to be deployed in an application catalog site which we do not have privilege to operate on. That's where we need support from the tenant administrator.

Tools and Software Packages

As SharePoint recommended, for our development tools, we are using:

1. NodeJS for application coding;
2. Yeoman for project/framework generating
3. Visual Code for IDE; and,
4. Gulp for build

To reduce development efforts to the minimum, we want to reuse as much code from the current dropdown menu as possible. Some of the third-party libraries/styles used there include:

1. jQuery,
2. bootstrap, and
3. fontawesome.

Build, Test and Deployment

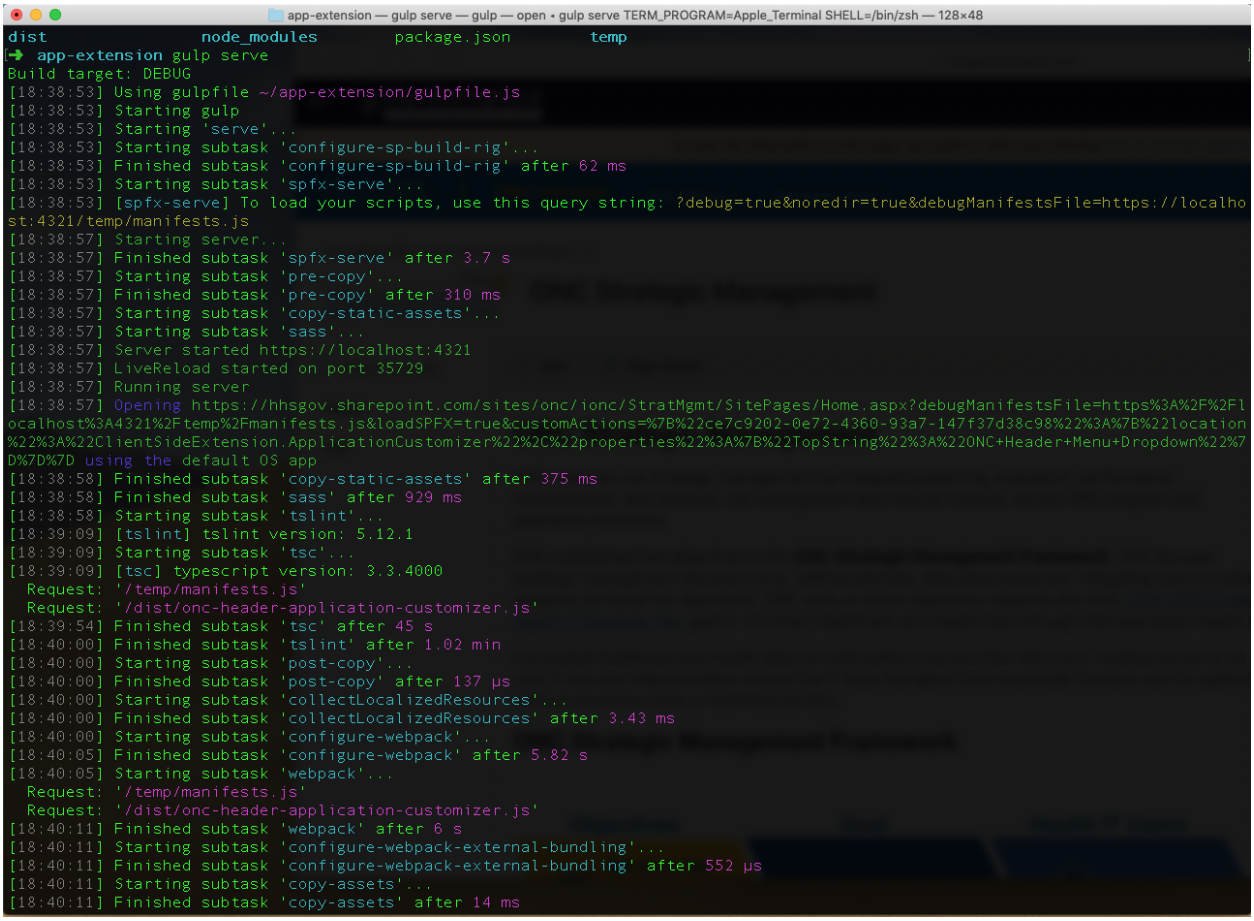
Once the SharePoint project is setup with Yeoman generator, we should be able to do:

1. prepare the source and development tools
   please refer to https://docs.microsoft.com/en-us/sharepoint/dev/spfx/set-up-your-development-environment to get the required tools and software package set up.
   Then: try to clone the repo, install the NodeJS library, gulp and then test build:

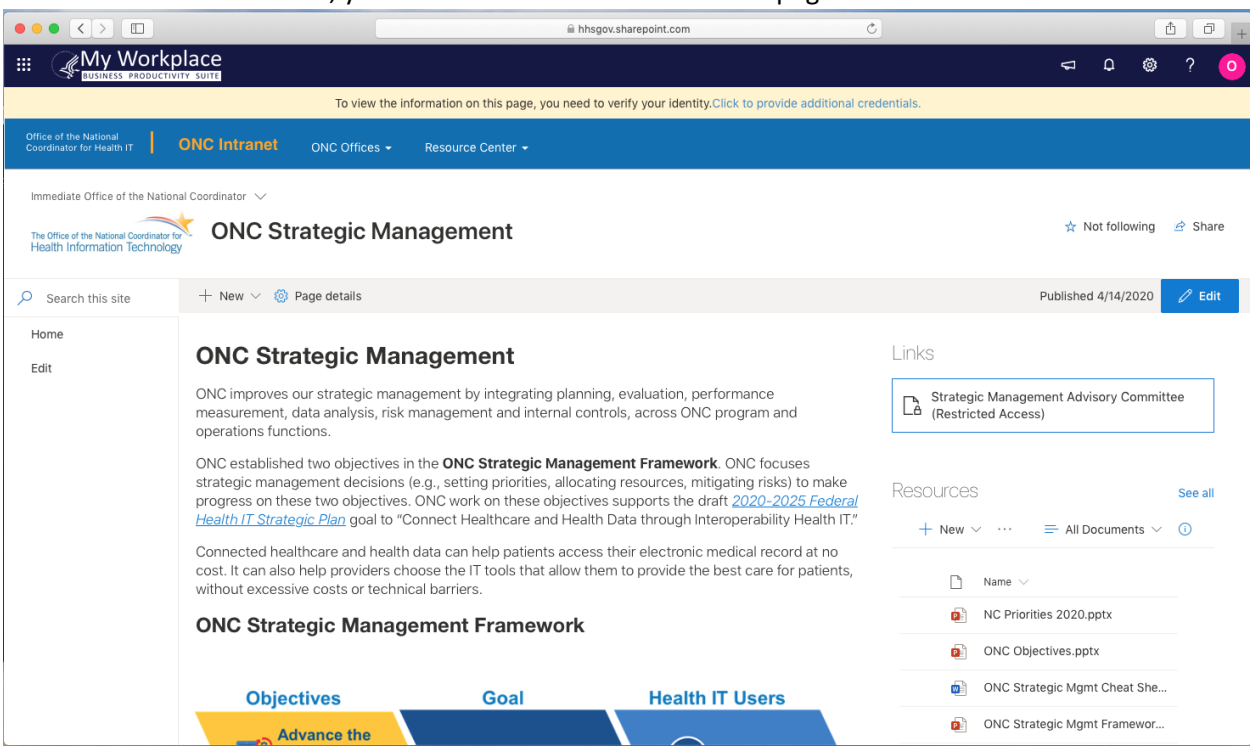
   ```
   git clone the repo
   npm i
   npm i -g gulp
   gulp
   ```

2. build and test with command:

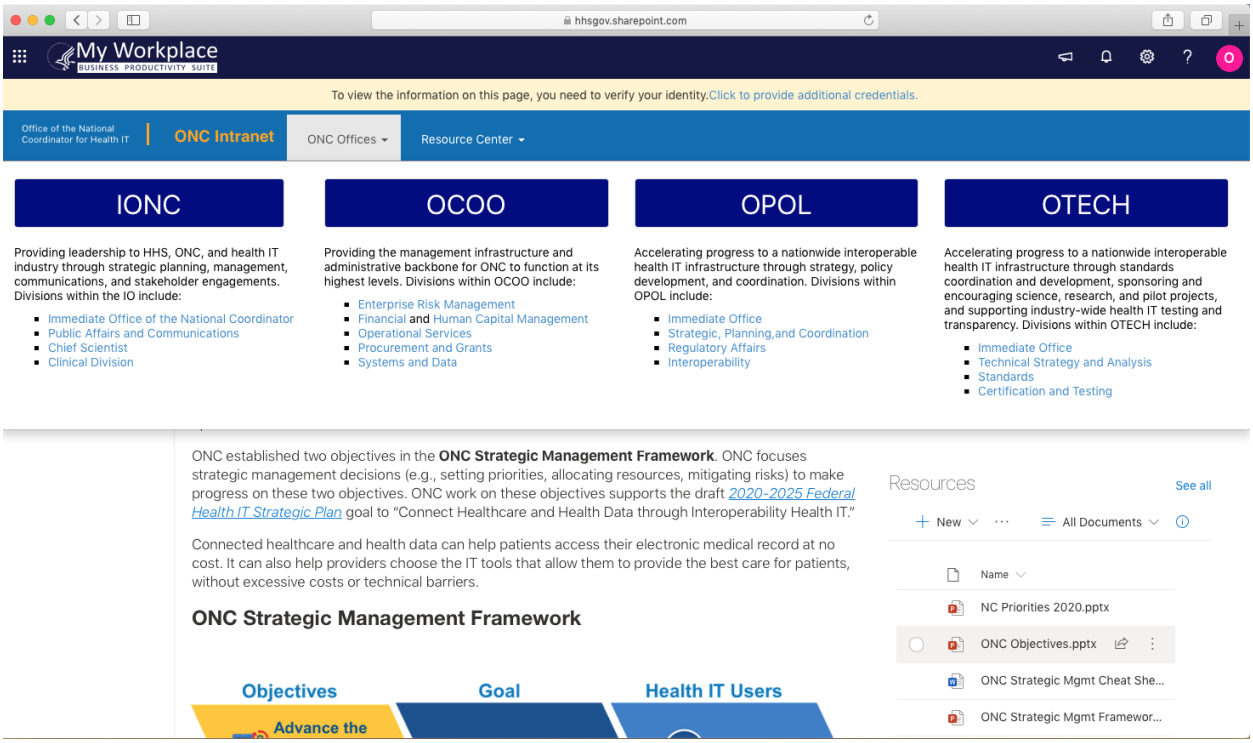
   gulp serve



Once the browser is loaded, you can see the menu load with the page:



Click on the dropdown, you can check things working –



3. to build a deployable package:

   gulp bundle --ship
   gulp package-solution –ship

4. Generated directories after build:

   lib/ - intermediate-stage common js build artifacts*
   dist/ - the bundled script, along with other resources*
   deploy/ - all resources which should be uploaded to a CDN.*

   For SharePoint deployment, you can use SharePoint/solution/onc-frame.sppkg.

5. to deploy the package:

   Previously, all add-ins and SharePoint Framework solutions had to be managed centrally in the tenant app catalog. While tenant administrators could delegate the access to other people in the organization, a deployed package was visible on all site collections. SharePoint offered no supported way of deploying add-ins and SharePoint Framework solutions only to specific sites.

   With the introduction of site collection app catalogs, tenant administrators can enable app catalog on the specific sites. Once enabled, site collection administrators can deploy SharePoint add-ins and SharePoint Framework solutions that will be available only in that particular site collection.

   The following schema illustrates using site collection app catalogs:

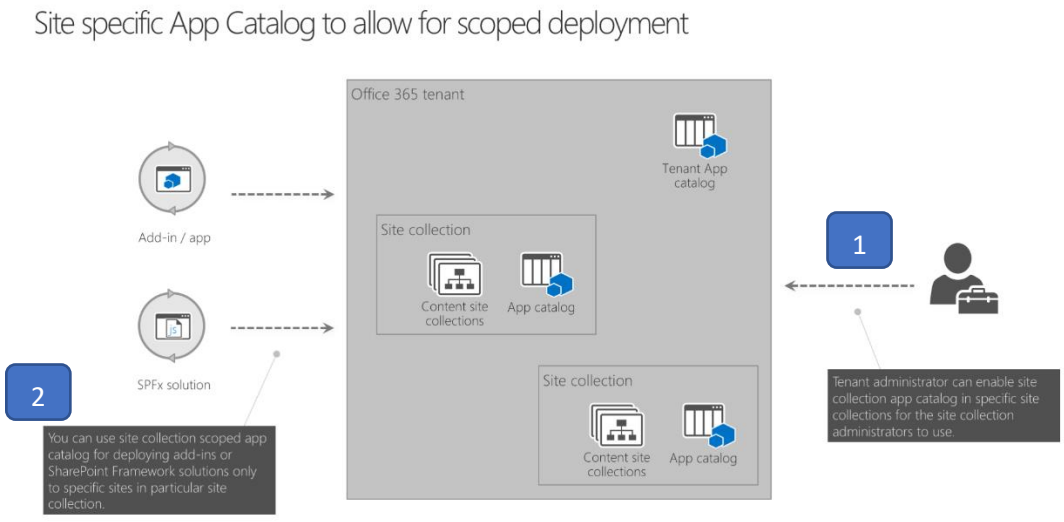


Diagram illustrating the concept of site collection app catalog

a. Tenant administrator enables the site collection app catalogs for our site collection
b. ONC developer upload deployable client-side solution package (SharePoint/solution/onc-frame.sppkg) to the site collection app catalog library.
c. Add the customer solution to required ONC office sites and verify the solution.

Key Code Logic and APIs

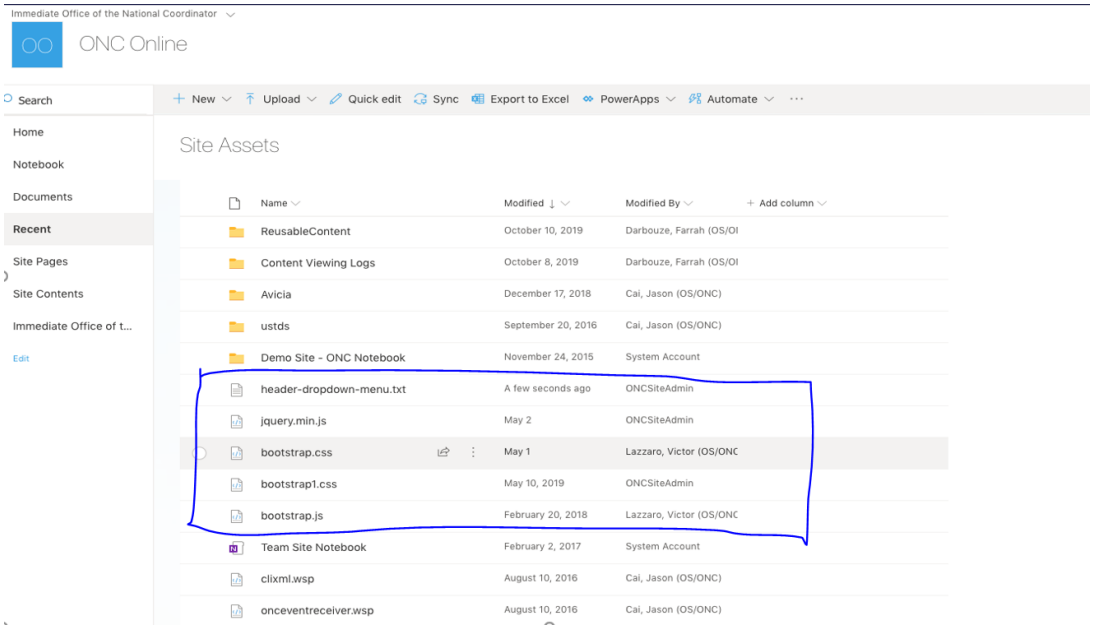
This code does not use any APIs.

This code loads library and contents from the ONC site and renders it to the page to generate the dropdown menu. Contents and libraries loaded from the ONC site are:

The open source libraries such as jquery, bootstrap and fontawesome, they need to be downloaded and upload to the ONC site manually. The dropdown menu contents are used for the ONC staff to change the content of the menu without redeploying the component.

Dependencies	**Purpose**	**File Name**
1. jQuery	Open source JS library for HTML page, used in the current dropdown menu	jquery.min.js
2. bootstrap	Open source JS/CSS library for HTML page, used in the current dropdown menu	bootstrap.js, bootstrap1.css
3. fontawesome	Open source CSS library for HTML page fonts and icons, used in the current dropdown menu	all.css
4. dropdown menu contents.	dynamic Text/HTML contents, extracted from the current dropdown menu	header-dropdown-menu.txt

All these are located on the ONC site in the SharePoint library - https://hhsgov.sharepoint.com/sites/onc/SiteAssets



All the above dependencies are dynamically loaded. The logic of loading these is in the code located at: src/extensions/oncHeader/OncHeaderApplicationCustomizer.ts.

Look into the code and check lines 125-142 shown as below. Those code perform dynamic loading of the library and contents:

```
125             function loadAllObjects() {
126               console.log('jquery loading completed....');
127               loadScript("https://hhsgov.sharepoint.com/sites/onc/SiteAssets/bootstrap.js", function(){console.log("bootstrap.js ready")});
128               locaCss("https://hhsgov.sharepoint.com/sites/onc/SiteAssets/bootstrap1.css", function(){console.log("bootstrap.css ready")});
129               locaCss("https://use.fontawesome.com/releases/v5.5.0/css/all.css", function(){console.log("fontawesome.css ready")});
130               jQuery(document).ready(function(){
131                 jQuery.get('https://hhsgov.sharepoint.com/sites/onc/SiteAssets/header-dropdown-menu.txt', function(tdata) {
132                   $('#oncmenubar').html(tdata);
133                 });
134                 $(".dropdown").hover(
135                     function() { $('.dropdown-menu', this).stop().fadeIn("fast");
136                     },
137                     function() { $('.dropdown-menu', this).stop().fadeOut("fast");
138                 });
139               });
140             }
141             loadScript("https://hhsgov.sharepoint.com/sites/onc/SiteAssets/jquery.min.js", loadAllObjects);
142             console.log('jquery loading started....');
```

After libraries loaded. Line #132 render the dropdown menu dynamically into the top bar placeholder.

Maintenance and Change Control

The contents header dropdown menu component is rather static. We'll have some of the content pulling from our current site. This allows us to reduce the needs of updating the component while maintain control of the displayed information in the meantime. Use other words, once test and deployed, the component should not require much maintenance.

Nevertheless, there can be requirements for major update of the component. In that case, we'll follow the current build, test and deployment procedure to redeploy the component.