Phased Headless Rollouts Of PWA

Kit (SIG instance)

November 28, 2023

VERSION 1.0

**Document Information**

|  |  |
| --- | --- |
| **Title** | Phased Headless Rollouts of PWA Kit (SIG instance) |
| **Original Author(s)** | *Rashid, Faizen (Cognizant)* |

**Contents**

[**INTRODUCTION** 4](#_Toc152175711)

[**STEPS TO PERFORM** 4](#_Toc152175712)

[**Important Note** 9](#_Toc152175713)

## **INTRODUCTION**

In this document we are going to discuss Phased rollouts of headless technologies for SIG instances.   
We are going to implement a Hybrid storefront site in which the PDP and the PLP pages will be based upon PWA kit and other pages will be based on SFRA.

## **STEPS TO PERFORM**

* The first step in creating a phased headless rollout is to set up your PWA Kit application to use SLAS (if you haven’t already). Follow the instructions in  [Set Up API Access](https://developer.salesforce.com/docs/commerce/pwa-kit-managed-runtime/guide/setting-up-api-access.html) guide.

Remember to follow the [extra step](https://developer.salesforce.com/docs/commerce/commerce-api/guide/authorization-for-shopper-apis.html#extra-steps-for-phased-rollouts-of-pwa-kit) for phased roll out even if you have SLAS configured earlier.

* Install the [Plugin SLAS cartridge](https://github.com/SalesforceCommerceCloud/plugin_slas). Complete installation instructions are provided in the README for the cartridge.
* Install the [Redirect Cartridge](https://github.com/SalesforceCommerceCloud/plugin_redirect) by following the instructions provided in the README for the cartridge **but** while updating the custom preference only mark the below 2 preference as Yes :
  + Toggle Product Redirect (pwa\_kit\_redirect\_product)
  + Toggle Category Redirect (pwa\_kit\_redirect\_category)

And update the below routes in the default routes under Routers preference

"Product": "/product/[productid]",

"Category": "/category/[categoryid]",

"Search": "/search?q=[searchterms]",

* Replace the code under the app/routes.jsx file in your PWA kit project with the below code:

/\*

 \* Copyright (c) 2021, salesforce.com, inc.

 \* All rights reserved.

 \* SPDX-License-Identifier: BSD-3-Clause

 \* For full license text, see the LICENSE file in the repo root or https://opensource.org/licenses/BSD-3-Clause

 \*/

/\* istanbul ignore file \*/

// NOTE!

// This file is being ignored in the test coverage report for now. It reports `0%` functions

// tested, which brings down the overall coverage and blocks CI. There are tests still, but

// we don't want it to count toward coverage until we figure out how to cover the `functions`

// metric for this file in its test.

import React, {useEffect} from 'react'

import loadable from '@loadable/component'

import {getConfig} from '@salesforce/pwa-kit-runtime/utils/ssr-config'

import {withRouter} from 'react-router-dom'

// non-extensible

import useMultiSite from './hooks/use-multi-site';

const DEFAULT\_SITE\_ID = 'RefArch'

// Components

import {Skeleton} from '@salesforce/retail-react-app/app/components/shared/ui'

import {configureRoutes} from '@salesforce/retail-react-app/app/utils/routes-utils'

const fallback = <Skeleton height="75vh" width="100%" />

// Pages

const Home = loadable(() => import('./pages/home'), {fallback})

const Login = loadable(() => import('./pages/login'), {fallback})

const ContentSearch = loadable(() => import('./pages/content-search'), {fallback})

const ContentDetails = loadable(() => import('./pages/content-details'), {fallback})

const Registration = loadable(() => import('./pages/registration'), {

    fallback

})

const ResetPassword = loadable(() => import('./pages/reset-password'), {fallback})

const Account = loadable(() => import('./pages/account'), {fallback})

const Cart = loadable(() => import('./pages/cart'), {fallback})

const Checkout = loadable(() => import('./pages/checkout'), {

    fallback

})

const CheckoutConfirmation = loadable(() => import('./pages/checkout/confirmation'), {fallback})

const LoginRedirect = loadable(() => import('./pages/login-redirect'), {fallback})

const ProductDetail = loadable(() => import('./pages/product-detail'), {fallback})

const ProductList = loadable(() => import('./pages/product-list'), {

    fallback

})

const Wishlist = loadable(() => import('./pages/account/wishlist'), {

    fallback

})

const PageNotFound = loadable(() => import('./pages/page-not-found'))

export const routes = [

    {

        path: '/',

        component: Home,

        exact: true

    },

    {

        path: '/product/:productId',

        component: ProductDetail

    },

    {

        path: '/search',

        component: ProductList

    },

    {

        path: '/category/:categoryId',

        component: ProductList

    },

    {

        path: '\*',

        component:  withRouter((props) => {

            const {location} = props

            const urlParams = new URLSearchParams(location.search)

            const {site} = useMultiSite()

            const siteId = site && site.id ? site.id : DEFAULT\_SITE\_ID

            useEffect(() => {

                var newURL = new URL(window.location)

                if (!urlParams.has('redirected')) {

                   newURL.searchParams.append('redirected', '1')

                    window.location.href = newURL

                }

            })

            if (urlParams.has('redirected')) {

                return <PageNotFound {...props} />

            }

            return null

        })

    }

]

export default () => {

    const config = getConfig()

    return configureRoutes(routes, config, {

        ignoredRoutes: ['/callback', '\*']

    })

}

* Configure On-Demand Sandbox to use alias by updating your Aliases(Merchant Tools > SEO > Aliases) with the changes below:

{

"\_\_version" : "1",

"settings" : {

"http-host" : "zzxi-001.dx.commercecloud.salesforce.com",

"https-host" : "zzxi-001.dx.commercecloud.salesforce.com"

}

}

* Install the reverse-proxy by following the instructions for setting up reverse-proxy as mentioned in the [README](https://github.com/SalesforceCommerceCloud/composable-hybrid-dev-server) for the repo but with the below two exception:
  + Instead of updating the routes.js file as per the instruction replace it’s entire code with the below:

'use strict';

const prependLocaleToPath = process.env.PREPEND\_LOCALE\_TO\_PATH === 'true';

const prependSiteIdToPath = process.env.PREPEND\_SITEID\_TO\_PATH === 'true';

const routes = [

  { path: '/product/:productId' },

  { path: '/search' },

  { path: '/category/:categoryId' },

  { path: '/\_\_mrt/hmr'},

  { path: '/worker.js'}

];

const PWA\_ROUTES = [

  {

    path: '/callback'

  }

];

// add :locale and :siteId path to routes (RefArch/en-US)

const prependPath = [];

if (prependSiteIdToPath) {

  prependPath.push(':siteId');

}

if (prependLocaleToPath) {

  PWA\_ROUTES.push({

    path: '/:locale',

    exact: true

  });

  prependPath.push(':locale');

}

if (prependPath.length) {

  PWA\_ROUTES.push({

    path: `/${prependPath.join('/')}`,

    exact: true

  });

}

for (let i = 0; i < routes.length; i++) {

  let route = routes[i];

  PWA\_ROUTES.push(route);

  // add :locale and :siteId path to routes

  if (prependPath.length) {

    PWA\_ROUTES.push({ path: `/${prependPath.join('/')}${route.path}` });

  }

}

module.exports = PWA\_ROUTES;

* + In the .env file update the value of PREPEND\_LOCALE\_TO\_PATH and PREPEND\_SITEID\_TO\_PATH as false
* Then open browser <http://localhost:8001>, you should have both the SFRA and PWA sites (for PLP and PDP) running on the same domain.

## **Important Note**

The steps mentioned in this POC documentation are specific to this particular use case, for getting detailed information on Phased Headless Rollouts please refer to the [developer guide](https://developer.salesforce.com/docs/commerce/pwa-kit-managed-runtime/guide/phased-headless-rollouts.html).