**POC for UBS Analytics with Application Insights**

**Application Insights Integration with SPO**

Pre-requisites:

* Azure Subscription that is able to create Application Insights
* Instrumentation Key from the Azure Application Insights resource
* Application ID from the Application Insights resource for Power BI integration
* SPFx familiarization (JavaScript, TypeScript, React)
* VS Code
* Node JS v10x

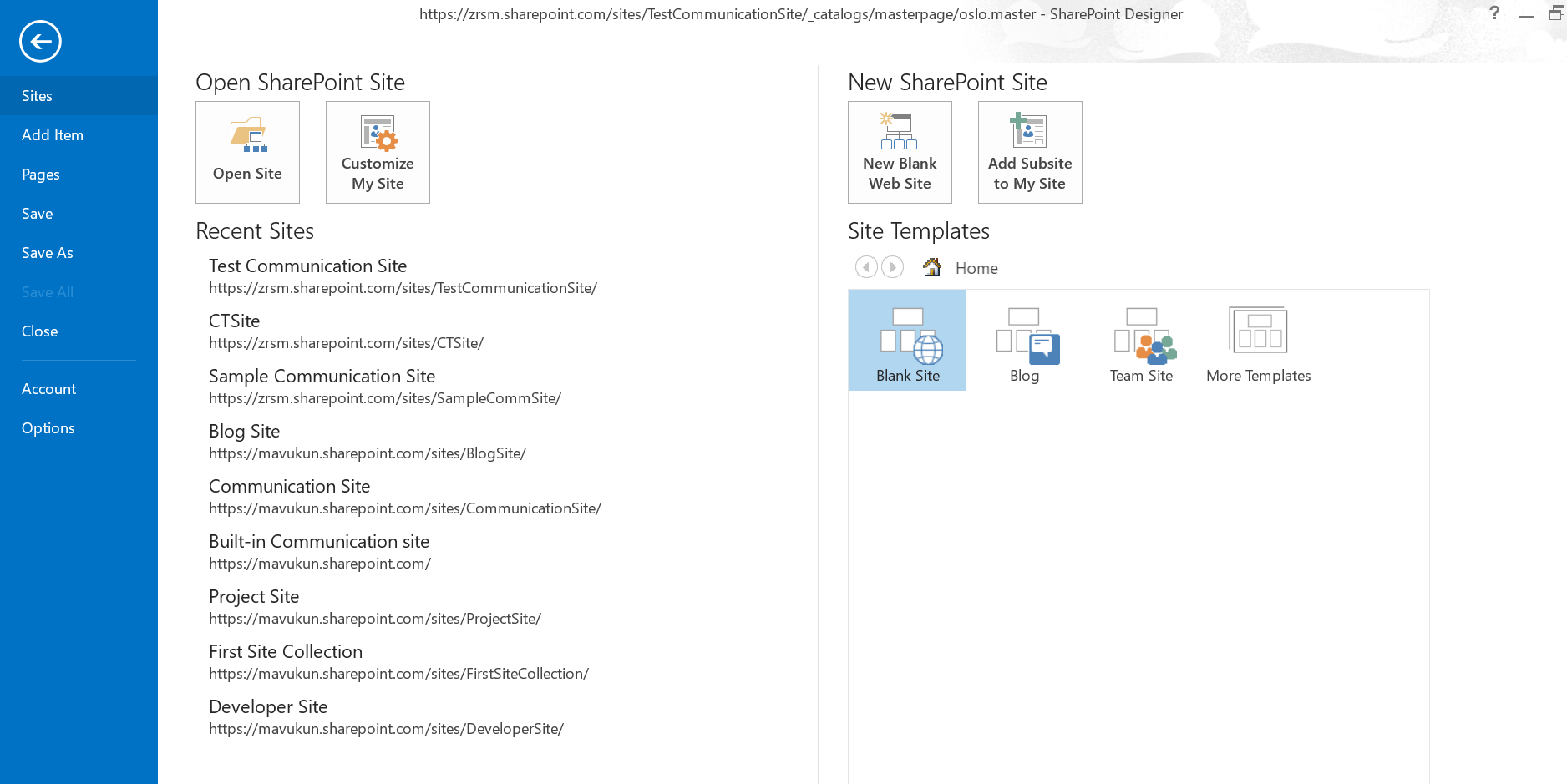
Note:

* Application Insight is a feature of Azure Monitor. For each site collection they should have their own Application Insight. (Costing is still a question but based on Azure’s policy, more usage means more cost.)
* Instrumentation Key – unique ID required for Application Insights to be enabled to the desired web application
* Application ID – unique ID required to access Application Insights API. (Will be used by Power BI)

**Enabling Application Insights to a SharePoint Site Collection**

One method is following the steps below using this site as reference: <https://docs.microsoft.com/en-us/azure/azure-monitor/app/website-monitoring>

1. Insert a desired site using its URL into SharePoint designer and open it.



1. Go to the ms docs site and copy this script:

<script type="text/javascript">

!function(T,l,y){var S=T.location,k="script",D="instrumentationKey",C="ingestionendpoint",I="disableExceptionTracking",E="ai.device.",b="toLowerCase",w="crossOrigin",N="POST",e="appInsightsSDK",t=y.name||"appInsights";(y.name||T[e])&&(T[e]=t);var n=T[t]||function(d){var g=!1,f=!1,m={initialize:!0,queue:[],sv:"5",version:2,config:d};function v(e,t){var n={},a="Browser";return n[E+"id"]=a[b](),n[E+"type"]=a,n["ai.operation.name"]=S&&S.pathname||"\_unknown\_",n["ai.internal.sdkVersion"]="javascript:snippet\_"+(m.sv||m.version),{time:function(){var e=new Date;function t(e){var t=""+e;return 1===t.length&&(t="0"+t),t}return e.getUTCFullYear()+"-"+t(1+e.getUTCMonth())+"-"+t(e.getUTCDate())+"T"+t(e.getUTCHours())+":"+t(e.getUTCMinutes())+":"+t(e.getUTCSeconds())+"."+((e.getUTCMilliseconds()/1e3).toFixed(3)+"").slice(2,5)+"Z"}(),iKey:e,name:"Microsoft.ApplicationInsights."+e.replace(/-/g,"")+"."+t,sampleRate:100,tags:n,data:{baseData:{ver:2}}}}var h=d.url||y.src;if(h){function a(e){var t,n,a,i,r,o,s,c,u,p,l;g=!0,m.queue=[],f||(f=!0,t=h,s=function(){var e={},t=d.connectionString;if(t)for(var n=t.split(";"),a=0;a<n.length;a++){var i=n[a].split("=");2===i.length&&(e[i[0][b]()]=i[1])}if(!e[C]){var r=e.endpointsuffix,o=r?e.location:null;e[C]="https://"+(o?o+".":"")+"dc."+(r||"services.visualstudio.com")}return e}(),c=s[D]||d[D]||"",u=s[C],p=u?u+"/v2/track":d.endpointUrl,(l=[]).push((n="SDK LOAD Failure: Failed to load Application Insights SDK script (See stack for details)",a=t,i=p,(o=(r=v(c,"Exception")).data).baseType="ExceptionData",o.baseData.exceptions=[{typeName:"SDKLoadFailed",message:n.replace(/\./g,"-"),hasFullStack:!1,stack:n+"\nSnippet failed to load ["+a+"] -- Telemetry is disabled\nHelp Link: https://go.microsoft.com/fwlink/?linkid=2128109\nHost: "+(S&&S.pathname||"\_unknown\_")+"\nEndpoint: "+i,parsedStack:[]}],r)),l.push(function(e,t,n,a){var i=v(c,"Message"),r=i.data;r.baseType="MessageData";var o=r.baseData;return o.message='AI (Internal): 99 message:"'+("SDK LOAD Failure: Failed to load Application Insights SDK script (See stack for details) ("+n+")").replace(/\"/g,"")+'"',o.properties={endpoint:a},i}(0,0,t,p)),function(e,t){if(JSON){var n=T.fetch;if(n&&!y.useXhr)n(t,{method:N,body:JSON.stringify(e),mode:"cors"});else if(XMLHttpRequest){var a=new XMLHttpRequest;a.open(N,t),a.setRequestHeader("Content-type","application/json"),a.send(JSON.stringify(e))}}}(l,p))}function i(e,t){f||setTimeout(function(){!t&&m.core||a()},500)}var e=function(){var n=l.createElement(k);n.src=h;var e=y[w];return!e&&""!==e||"undefined"==n[w]||(n[w]=e),n.onload=i,n.onerror=a,n.onreadystatechange=function(e,t){"loaded"!==n.readyState&&"complete"!==n.readyState||i(0,t)},n}();y.ld<0?l.getElementsByTagName("head")[0].appendChild(e):setTimeout(function(){l.getElementsByTagName(k)[0].parentNode.appendChild(e)},y.ld||0)}try{m.cookie=l.cookie}catch(p){}function t(e){for(;e.length;)!function(t){m[t]=function(){var e=arguments;g||m.queue.push(function(){m[t].apply(m,e)})}}(e.pop())}var n="track",r="TrackPage",o="TrackEvent";t([n+"Event",n+"PageView",n+"Exception",n+"Trace",n+"DependencyData",n+"Metric",n+"PageViewPerformance","start"+r,"stop"+r,"start"+o,"stop"+o,"addTelemetryInitializer","setAuthenticatedUserContext","clearAuthenticatedUserContext","flush"]),m.SeverityLevel={Verbose:0,Information:1,Warning:2,Error:3,Critical:4};var s=(d.extensionConfig||{}).ApplicationInsightsAnalytics||{};if(!0!==d[I]&&!0!==s[I]){var c="onerror";t(["\_"+c]);var u=T[c];T[c]=function(e,t,n,a,i){var r=u&&u(e,t,n,a,i);return!0!==r&&m["\_"+c]({message:e,url:t,lineNumber:n,columnNumber:a,error:i}),r},d.autoExceptionInstrumented=!0}return m}(y.cfg);function a(){y.onInit&&y.onInit(n)}(T[t]=n).queue&&0===n.queue.length?(n.queue.push(a),n.trackPageView({})):a()}(window,document,{

src: "https://js.monitor.azure.com/scripts/b/ai.2.min.js", // The SDK URL Source

// name: "appInsights", // Global SDK Instance name defaults to "appInsights" when not supplied

// ld: 0, // Defines the load delay (in ms) before attempting to load the sdk. -1 = block page load and add to head. (default) = 0ms load after timeout,

// useXhr: 1, // Use XHR instead of fetch to report failures (if available),

crossOrigin: "anonymous", // When supplied this will add the provided value as the cross origin attribute on the script tag

// onInit: null, // Once the application insights instance has loaded and initialized this callback function will be called with 1 argument -- the sdk instance (DO NOT ADD anything to the sdk.queue -- As they won't get called)

cfg: { // Application Insights Configuration

connectionString:"InstrumentationKey=YOUR\_INSTRUMENTATION\_KEY\_GOES\_HERE;"

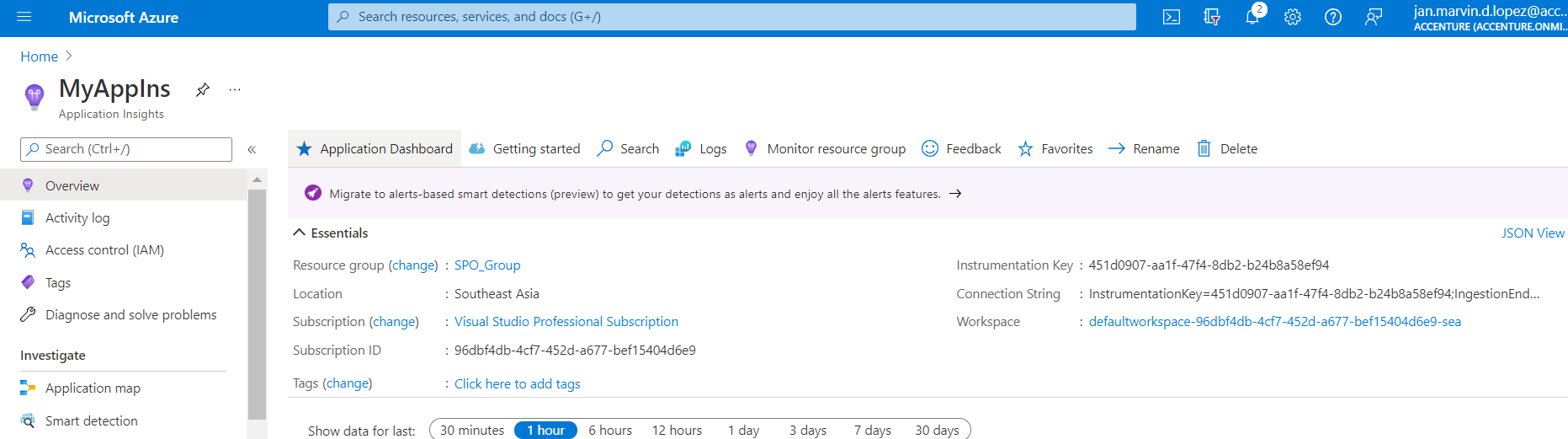
/\* ...Other Configuration Options... \*/

}});

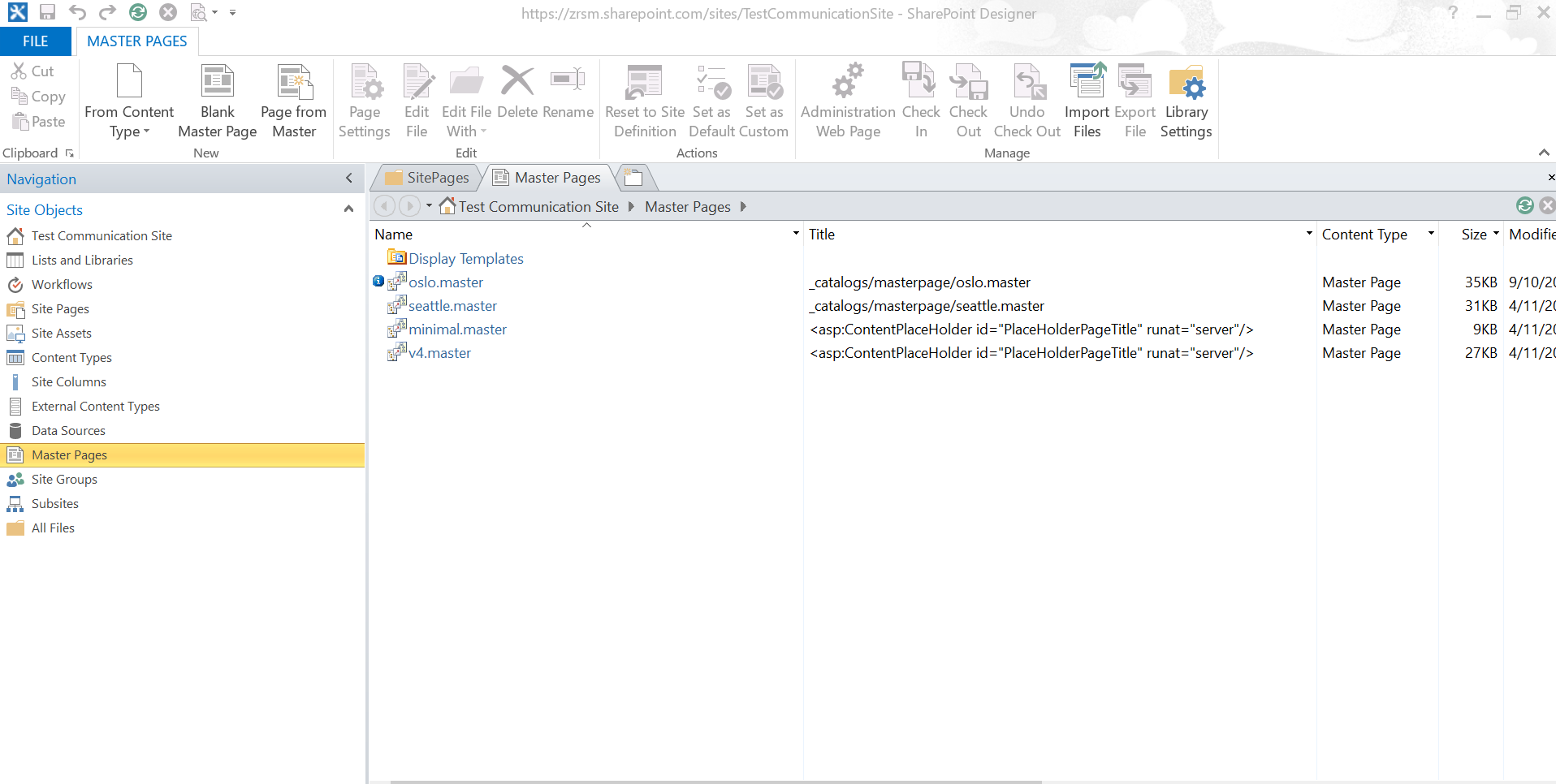
</script>

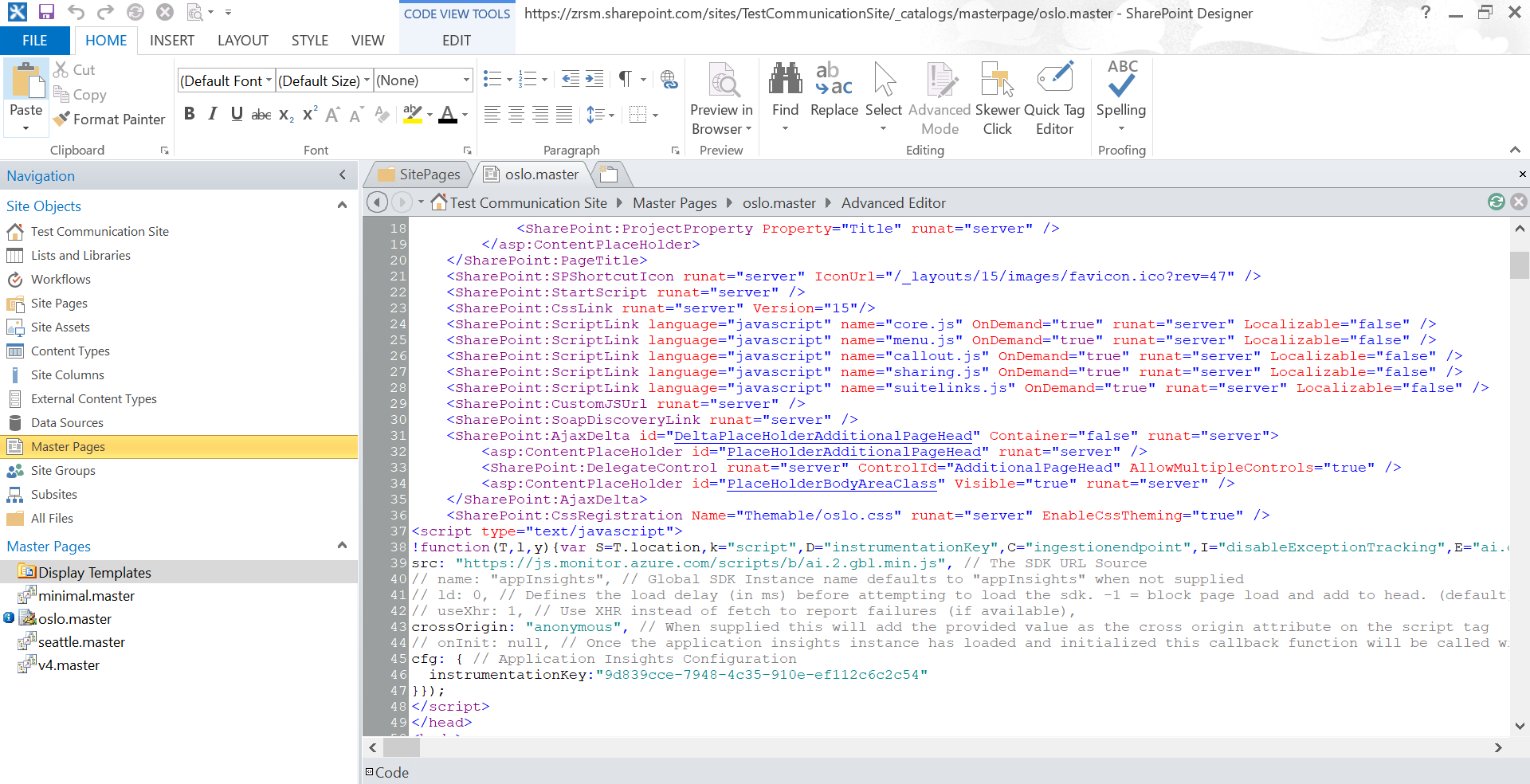
1. The script should be injected within the master pages of a SharePoint site. Also you need to identify what master page is currently being used by the site.
2. Using SharePoint designer, we can edit a site collection’s master page. The script should be injected within the <HEAD> tag. The Instrumentation key value should be replaced with the Instrumentation key value of the created Application Insights resource.

Instrumentation key is found here (‘MyAppIns’ is an application insights resource name):



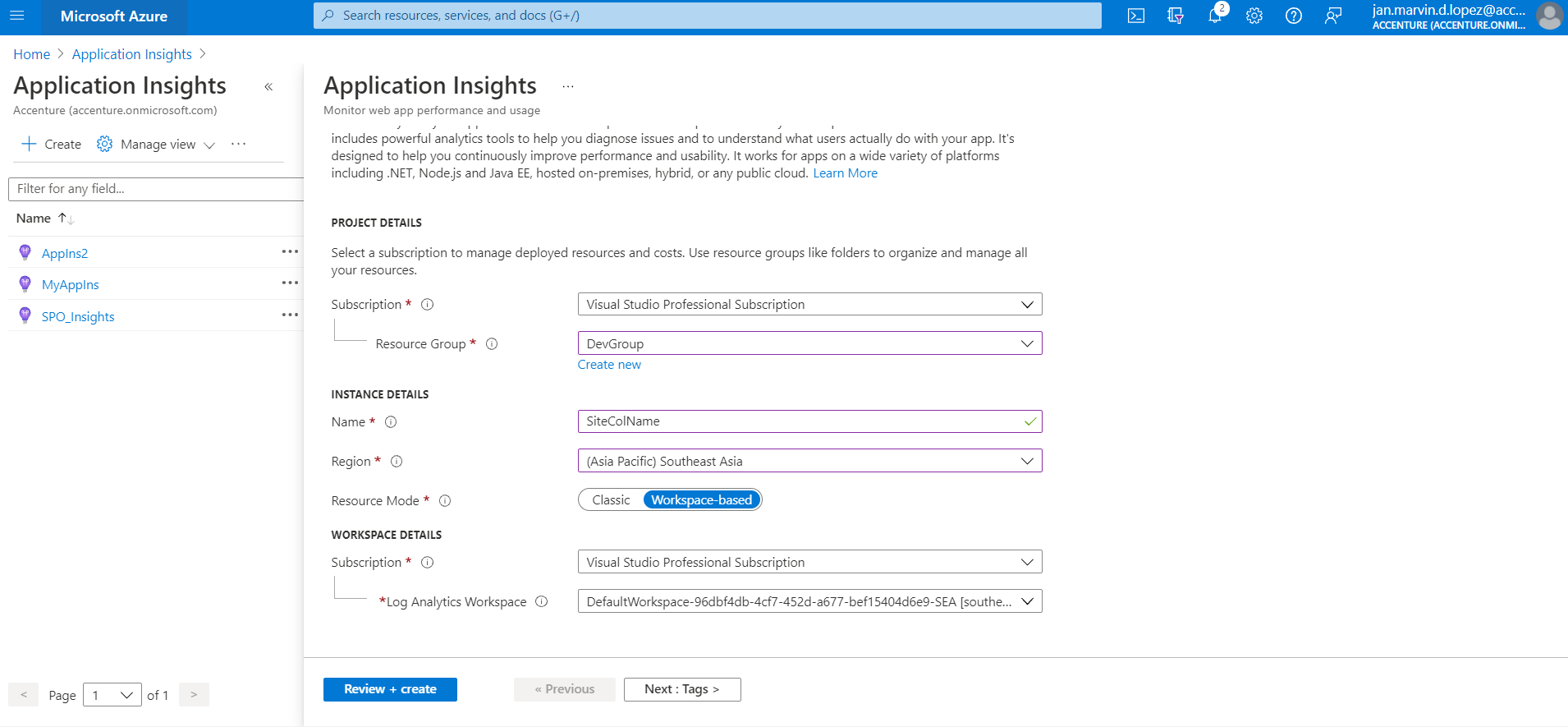
SharePoint Designer master pages are found here:

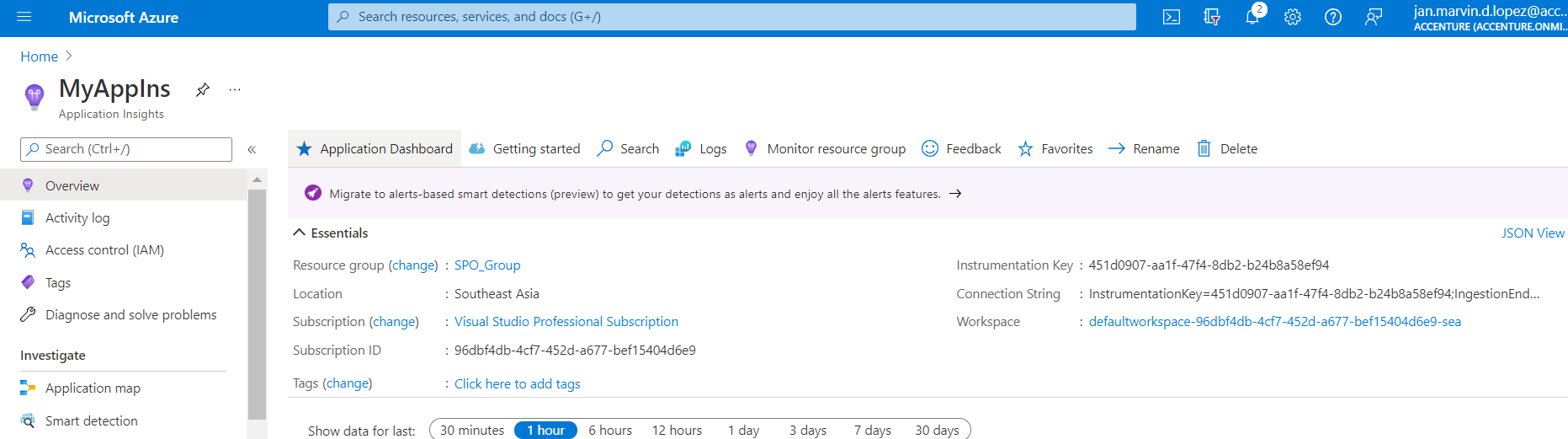




The other method is using SPFx:

This grants more customizations and gives access to the SharePoint Framework which provides full support for client-side SharePoint development.

1. Create an Application Insights resource. Fill out the following and once done click create. The name of this resource should be the same with the web application we want to monitor. 
2. Get Instrumentation Key



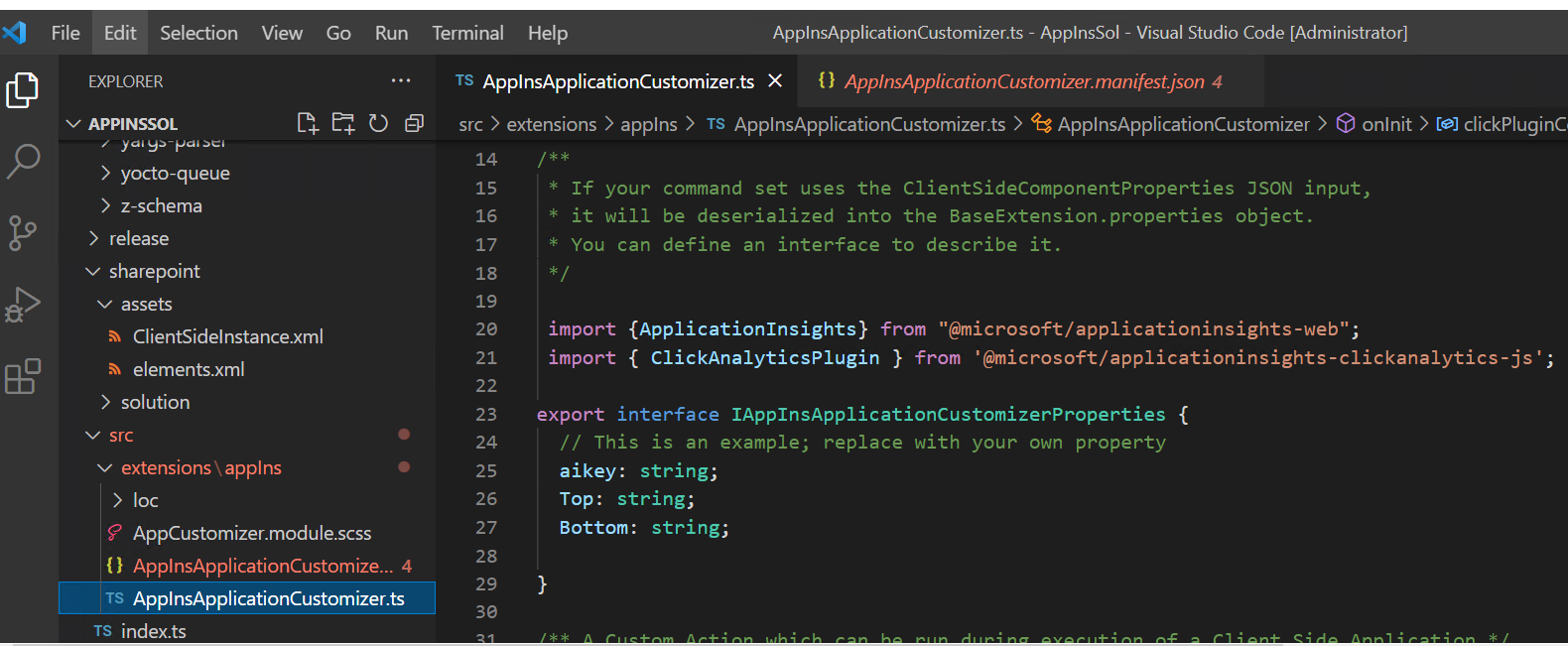
1. Create SPFx application customizer (Extension) use the reference provide to get an in-depth step by step on how to create an SPFx application customizer: <https://docs.microsoft.com/en-us/sharepoint/dev/spfx/extensions/get-started/build-a-hello-world-extension>
2. Install the following SDKs (you can use the VS Code’s terminal to run this command):

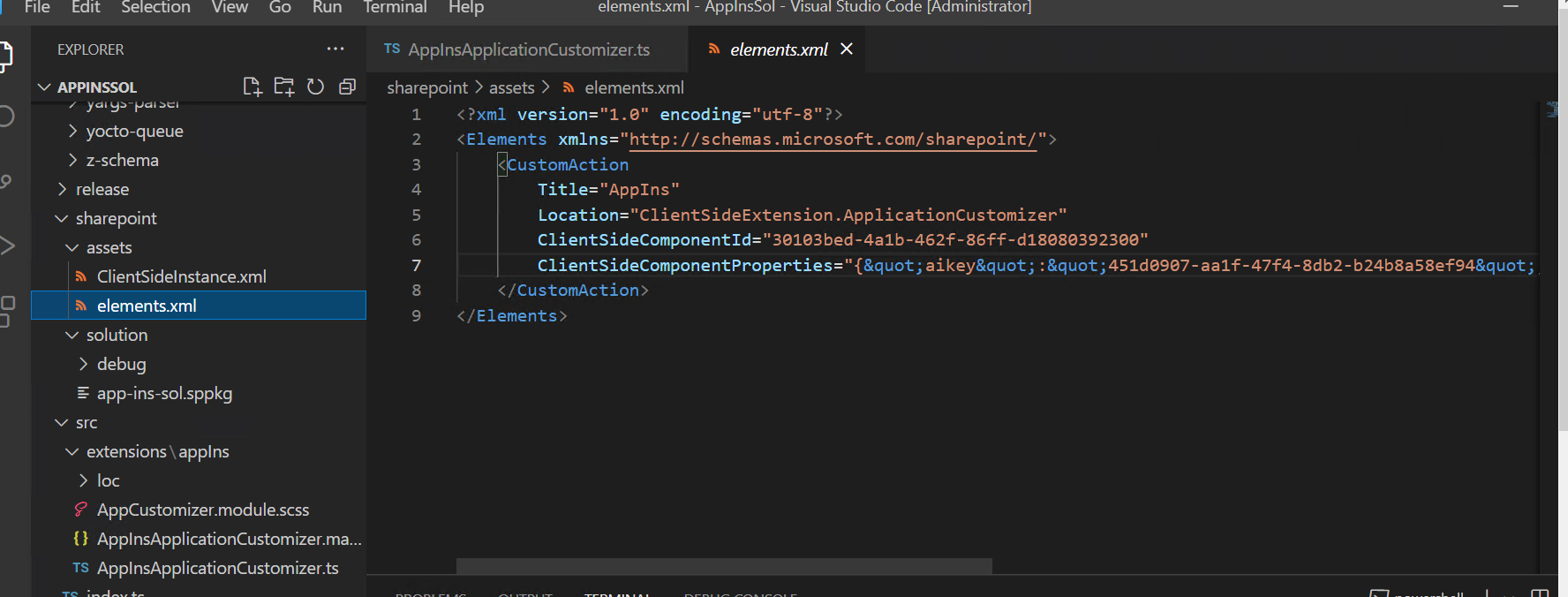
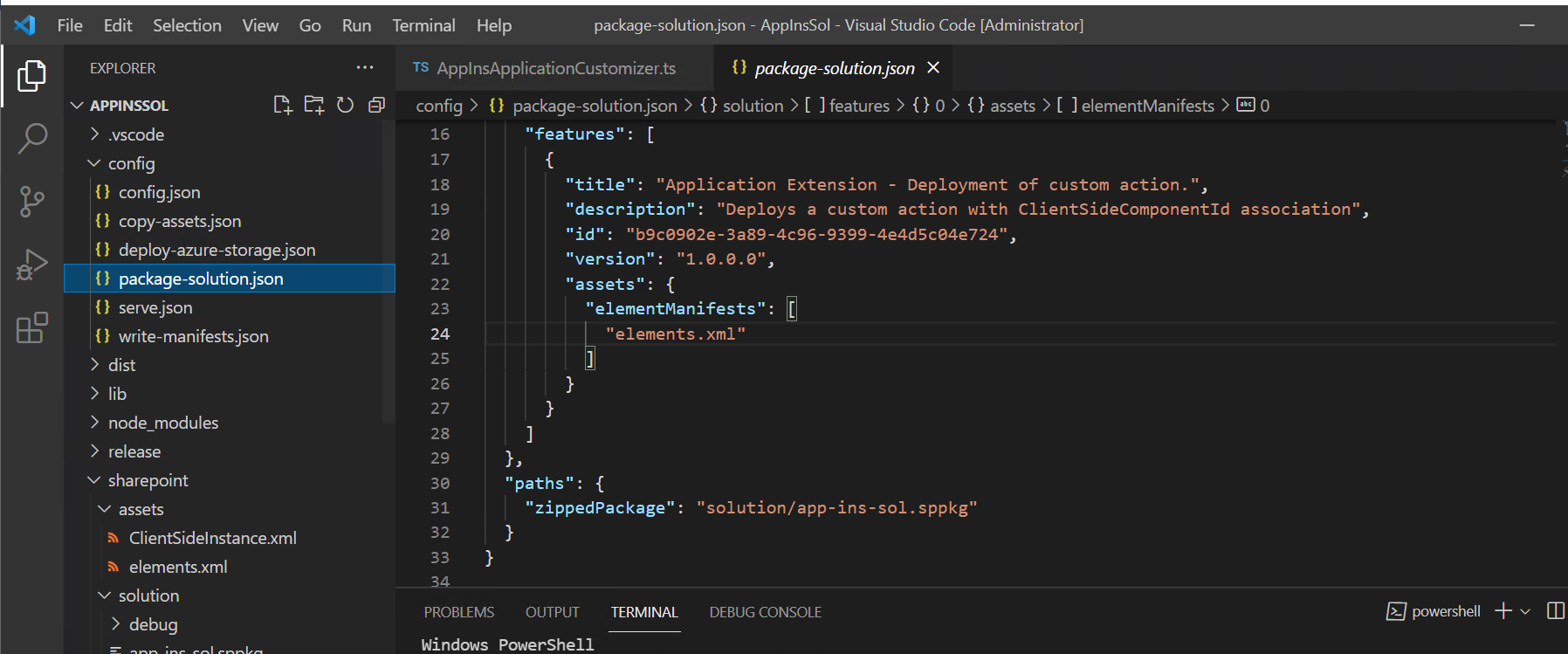
npm install applicationinsights –save : This SDK is Deprecated (not recommended)

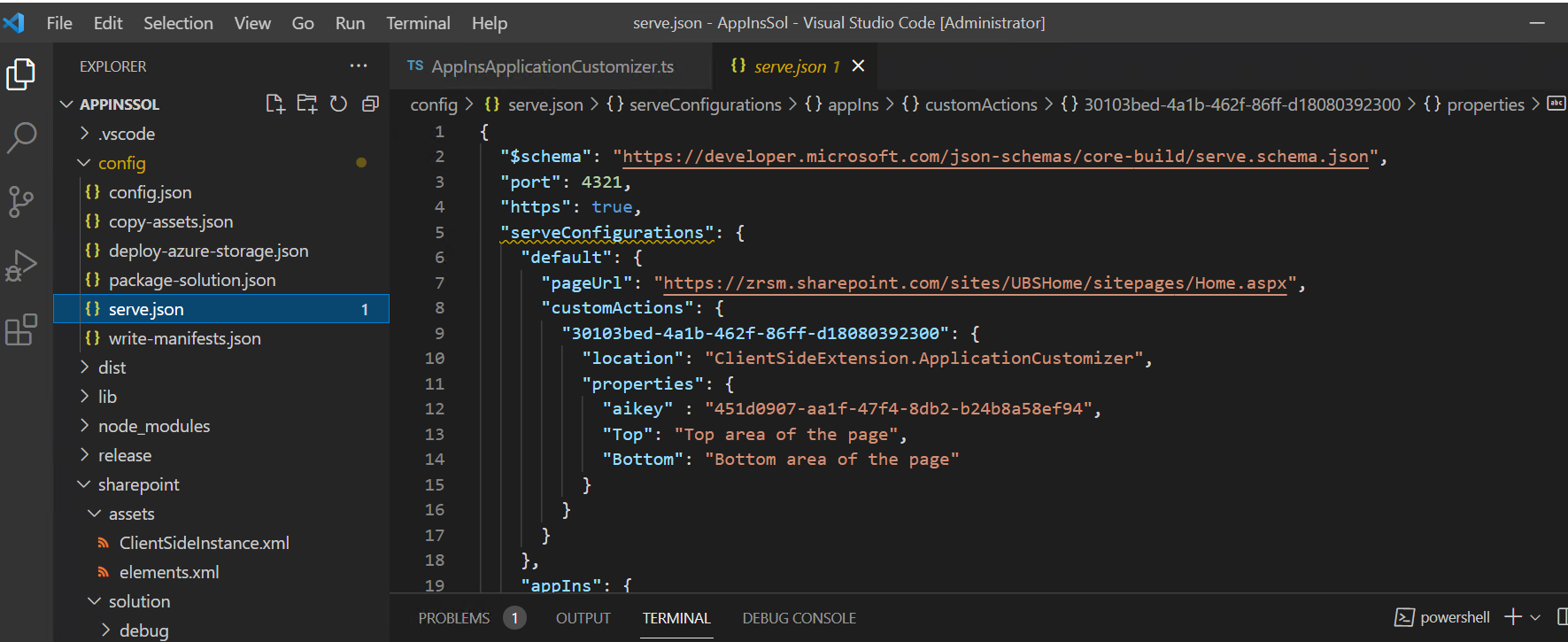
npm install --save @microsoft/applicationinsights-clickanalytics-js @microsoft/applicationinsights-web (This SDK also include the ‘ApplicationInsights’ library)

ApplicationInsights – Library used to enable the application insights monitoring within the site.

ClickAnalyticsPlugin – used to automatically capture click events within a site collection. (Within web pages and site libraries)

The following libraries should be imported on the main script

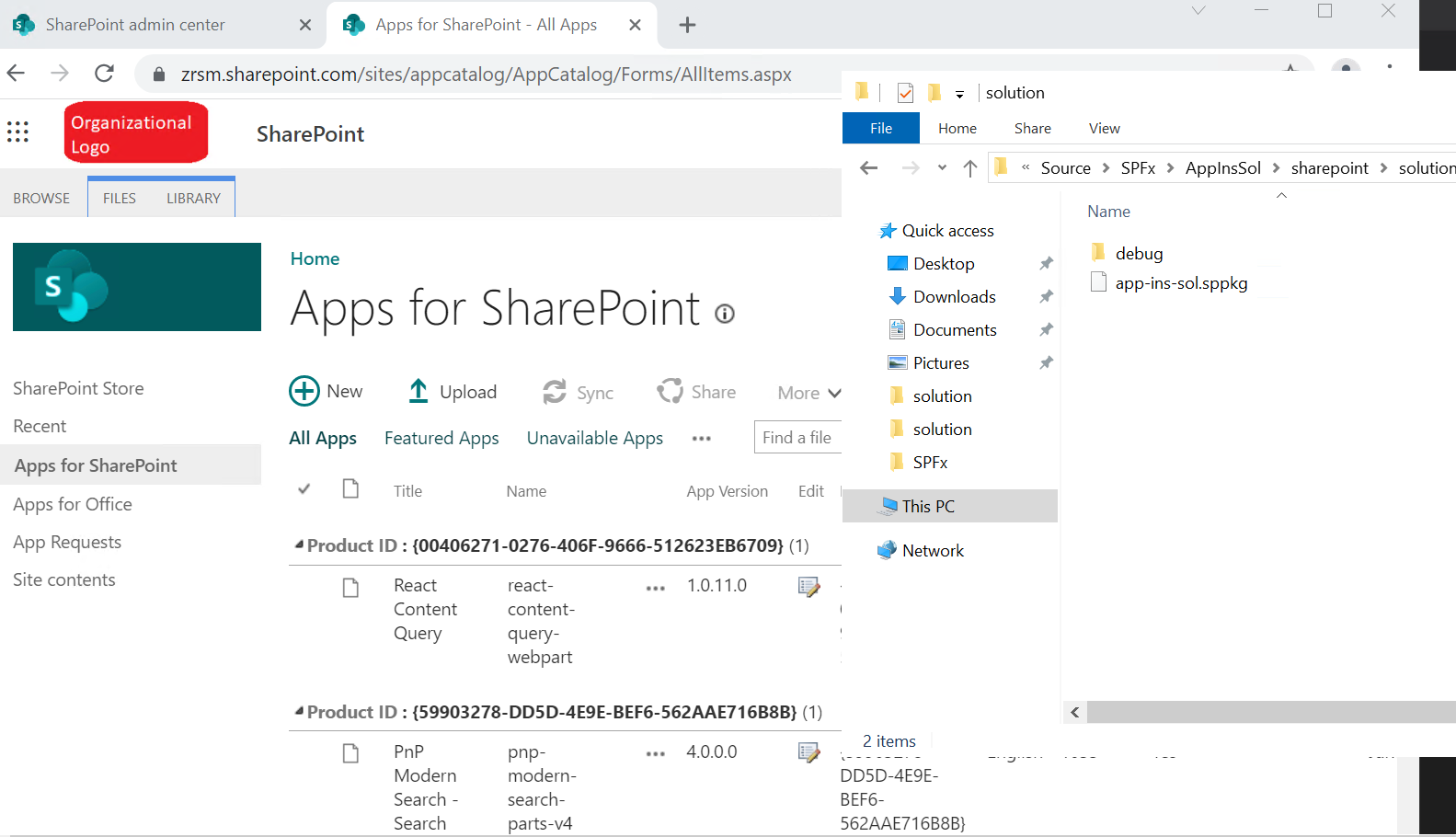
1. Whole code of the main script (Please disregard the place holder part of the solution. That is just to confirm if the solution is working and properly deployed)
2. import { override } from '@microsoft/decorators';
3. import { Log } from '@microsoft/sp-core-library';
4. import {
5. BaseApplicationCustomizer,
6. PlaceholderContent,
7. PlaceholderName
8. } from '@microsoft/sp-application-base';
9. import { Dialog } from '@microsoft/sp-dialog';
10. import styles from './AppCustomizer.module.scss';
11. import \* as strings from 'AppInsApplicationCustomizerStrings';
12. import { escape } from '@microsoft/sp-lodash-subset';
13. const LOG\_SOURCE: string = 'AppInsApplicationCustomizer';
14. /\*\*
15. \* If your command set uses the ClientSideComponentProperties JSON input,
16. \* it will be deserialized into the BaseExtension.properties object.
17. \* You can define an interface to describe it.
18. \*/
19. import {ApplicationInsights} from "@microsoft/applicationinsights-web";
20. import { ClickAnalyticsPlugin } from '@microsoft/applicationinsights-clickanalytics-js';
21. export interface IAppInsApplicationCustomizerProperties {
22. // This is an example; replace with your own property
23. aikey: string;
24. Top: string;
25. Bottom: string;
26. }
27. /\*\* A Custom Action which can be run during execution of a Client Side Application \*/
28. export default class AppInsApplicationCustomizer
29. extends BaseApplicationCustomizer<IAppInsApplicationCustomizerProperties> {
30. private \_topPlaceholder: PlaceholderContent | undefined;
31. private \_bottomPlaceholder: PlaceholderContent | undefined;
32. @override
33. public onInit(): Promise<void> {
34. Log.info(LOG\_SOURCE, `Initialized ${strings.Title}`);
35. const clickPluginInstance = new ClickAnalyticsPlugin();
36. // Click Analytics configuration
37. const clickPluginConfig = {
38. autoCapture: true,
39. dataTags: {
40. useDefaultContentNameOrId: true,
41. parentDataTag: true
42. }
43. };
44. const AppInsights = new ApplicationInsights({config: {
45. instrumentationKey: this.properties.aikey,
46. //instrumentationKey: "451d0907-aa1f-47f4-8db2-b24b8a58ef94",
47. extensions: [clickPluginInstance],
48. extensionConfig: {
49. [clickPluginInstance.identifier]: clickPluginConfig
50. },
51. }});
52. AppInsights.loadAppInsights();
53. AppInsights.trackPageView();
54. AppInsights.trackEvent({name: "analyticsButton"});
55. this.context.placeholderProvider.changedEvent.add(this, this.\_renderPlaceHolders);
56. return Promise.resolve();
57. }
58. private \_renderPlaceHolders(): void {
59. console.log("HelloWorldApplicationCustomizer.\_renderPlaceHolders()");
60. console.log(
61. "Available placeholders: ",
62. this.context.placeholderProvider.placeholderNames
63. .map(name => PlaceholderName[name])
64. .join(", ")
65. );
67. // Handling the top placeholder
68. if (!this.\_topPlaceholder) {
69. this.\_topPlaceholder = this.context.placeholderProvider.tryCreateContent(
70. PlaceholderName.Top,
71. { onDispose: this.\_onDispose }
72. );
74. // The extension should not assume that the expected placeholder is available.
75. if (!this.\_topPlaceholder) {
76. console.error("The expected placeholder (Top) was not found.");
77. return;
78. }
80. if (this.properties) {
81. let topString: string = this.properties.Top;
82. if (!topString) {
83. topString = "(Top property was not defined.)";
84. }
86. if (this.\_topPlaceholder.domElement) {
87. this.\_topPlaceholder.domElement.innerHTML = `
88. <div class="${styles.app}">
89. <div class="${styles.top}">
90. <i class="ms-Icon ms-Icon--Info" aria-hidden="true"></i> ${escape(
91. topString
92. )}
93. </div>
94. </div>`;
95. }
96. }
97. }
99. // Handling the bottom placeholder
100. if (!this.\_bottomPlaceholder) {
101. this.\_bottomPlaceholder = this.context.placeholderProvider.tryCreateContent(
102. PlaceholderName.Bottom,
103. { onDispose: this.\_onDispose }
104. );
106. // The extension should not assume that the expected placeholder is available.
107. if (!this.\_bottomPlaceholder) {
108. console.error("The expected placeholder (Bottom) was not found.");
109. return;
110. }
112. if (this.properties) {
113. let bottomString: string = this.properties.Bottom;
114. if (!bottomString) {
115. bottomString = "(Bottom property was not defined.)";
116. }
118. if (this.\_bottomPlaceholder.domElement) {
119. this.\_bottomPlaceholder.domElement.innerHTML = `
120. <div class="${styles.app}">
121. <div class="${styles.bottom}">
122. <i class="ms-Icon ms-Icon--Info" aria-hidden="true"></i> ${escape(
123. bottomString
124. )}
125. </div>
126. </div>`;
127. }
128. }
129. }
130. }
131. private \_onDispose(): void {
132. console.log('[HelloWorldApplicationCustomizer.\_onDispose] Disposed custom top and bottom placeholders.');
133. }
134. }
135. We need to supply the instrumentation key (aikey) of the application insights here on the elements.xml before deploying the solution. Format should be “{&quot ;<property name of the instrumentation key>&quot;:&quot;<VALUE OF THE INSTRUMENTATION KEY TO BE FOUND IN APPLICATION INSIGHTS RESOURCE>&quot;,<OTHER PROPERTIES ADD HERE, SAME FORMAT AS THE FIRST ONE, ALSO IT SHOULD BE SEPARATED BY COMMA>}
136. The elements.xml should be included in the “elementManifests” so that it will be recognized by the package-solution gulp command to be included in packaging. You will notice that the ClientSideInstance is not included because I only deployed this solution to a single site collection. ClientSideInstance will be used if the solution will be deployed to multiple site collections at once. If that’s the case, populate the ClientSideInstance the same format with elements.xml
137. For Testing purposes you can populate the serve.json with the instrumentation key



1. Once you are done with these steps, please run the following command on the vs code terminal: gulp build, gulp bundle --ship, gulp package-solution--ship

(--ship is to minify the package solution and make it final. If the --ship is not present, running the gulp serve command would still be required to run this solution within the site collection)

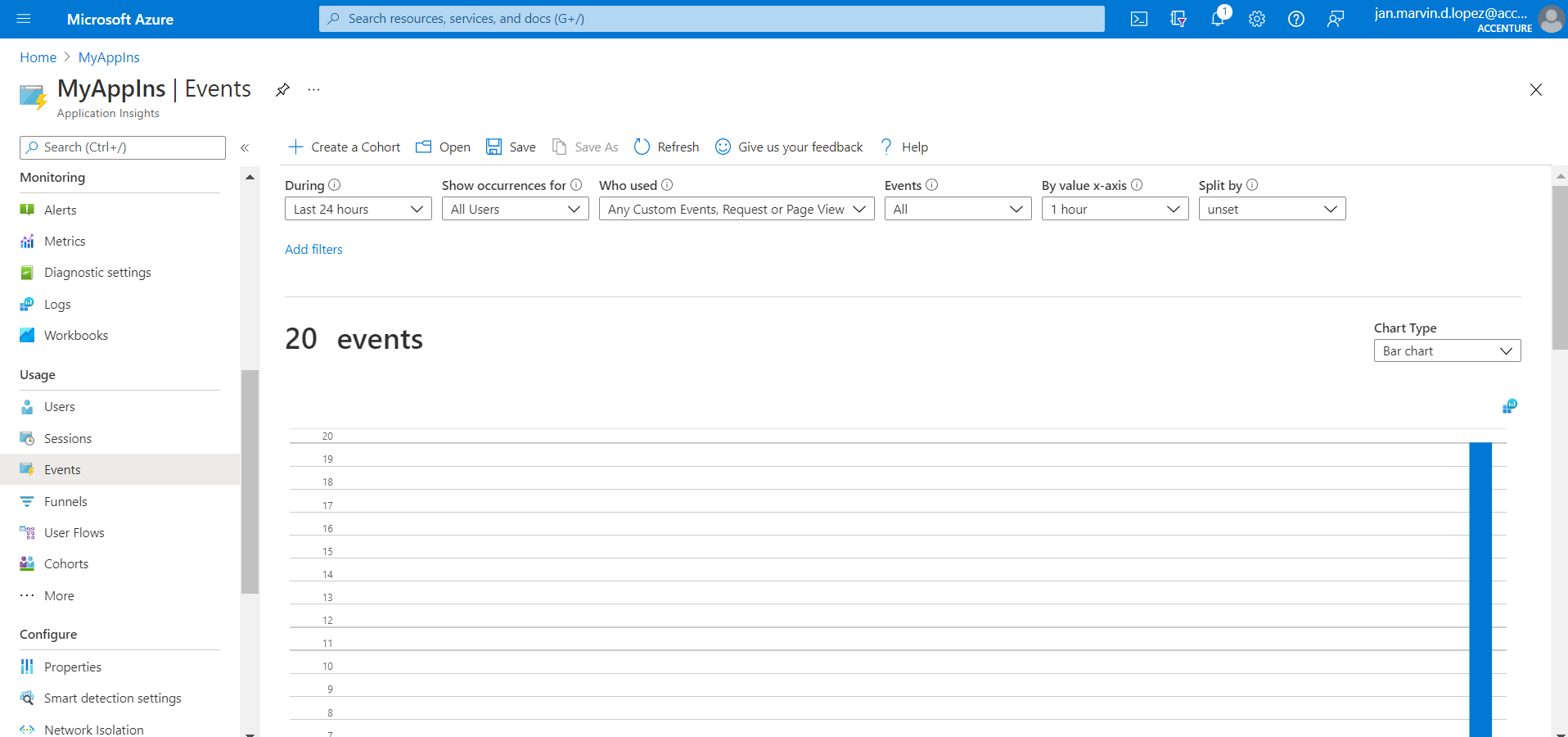
1. Go to the solution folder and grab the .sppkg file (this file is to be deployed in the app catalog of the SharePoint online tenant). You can simply right click the solution folder within the vs code and hit reveal in folder to show the folder location of the .sppkg file.
2. Go to tenant sharepoint admin site and locate the app catalog(it can be found in “More Features”). Deploy the sppkg solution by uploading it.



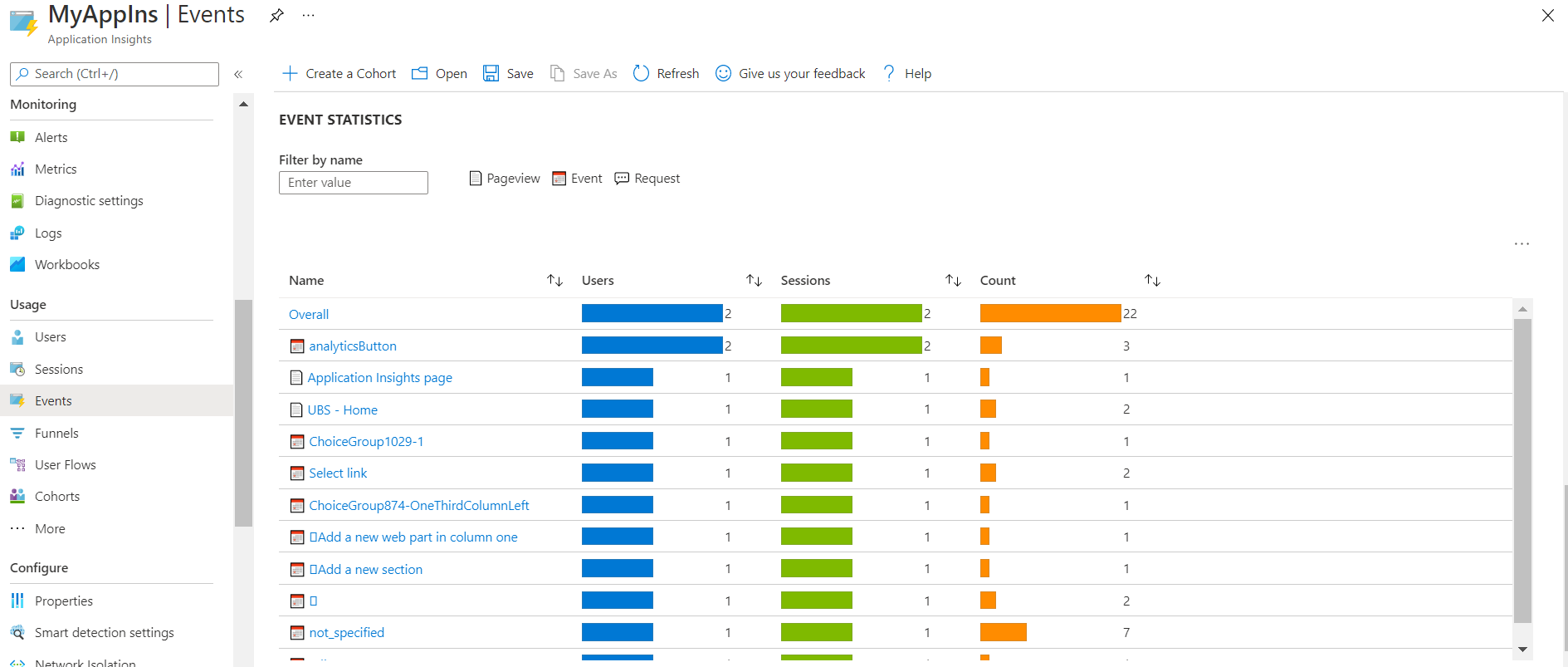
1. Once that is enabled, the Application Insights will start on monitoring the site collection to which the SPFx solution is deployed and enabled.

Checking if the Application Insights is enabled and working:

We can verify it by checking the Application Insights resource we used. By going to the events tab in usage, we can check the number of events we triggered during a certain time period(in this case “Last 24 hours”).

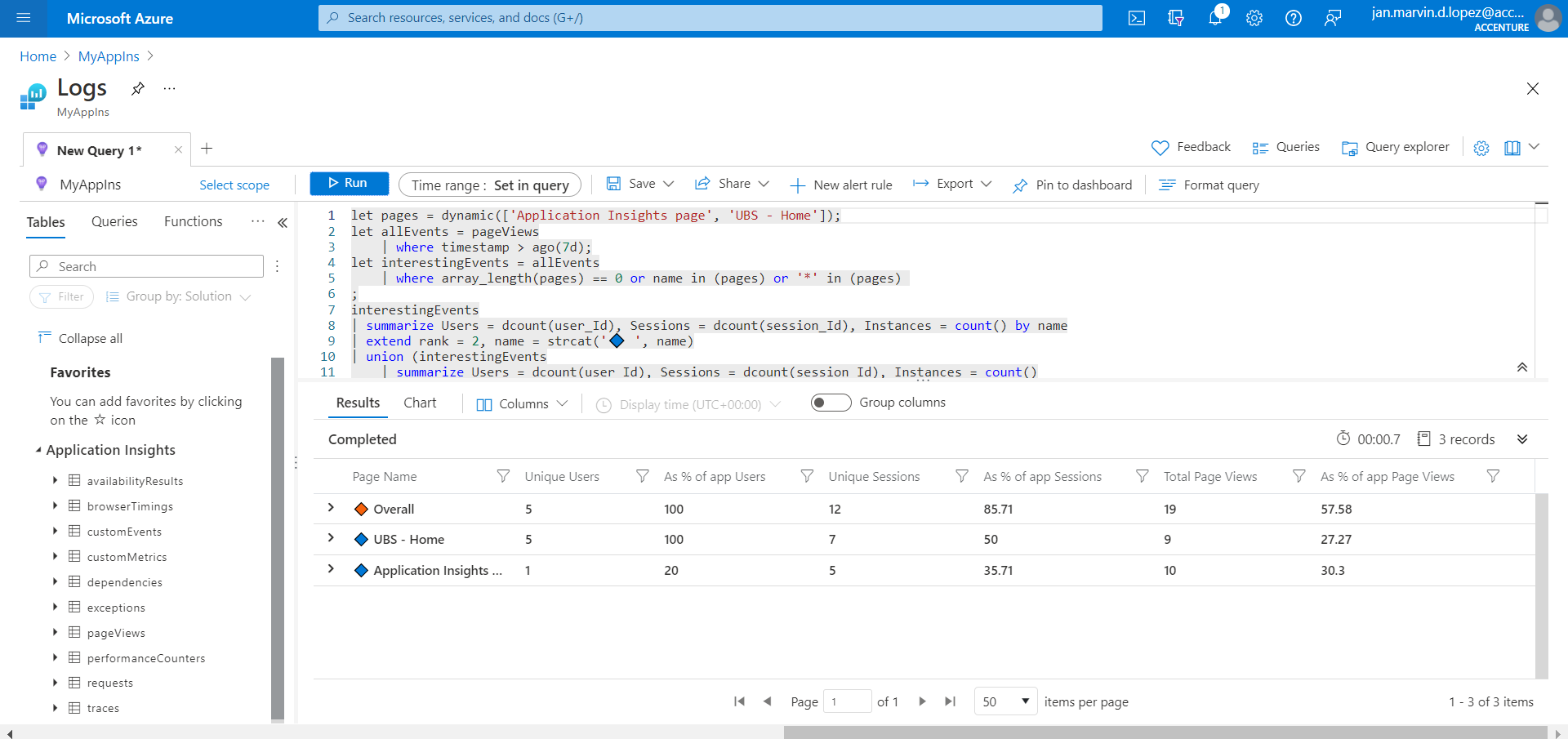


If we click View More Insights, a list events will appear.



**Note:** The Events triggered prior to the deployment of the SPFx solution will not be recorded here, only the events triggers right after it was deployed.

To effectively manipulate how the variables, elements, events are shown we can leverage the Logs feature of the Application Insights. Ref: <https://docs.microsoft.com/en-us/azure/azure-monitor/logs/log-query-overview>

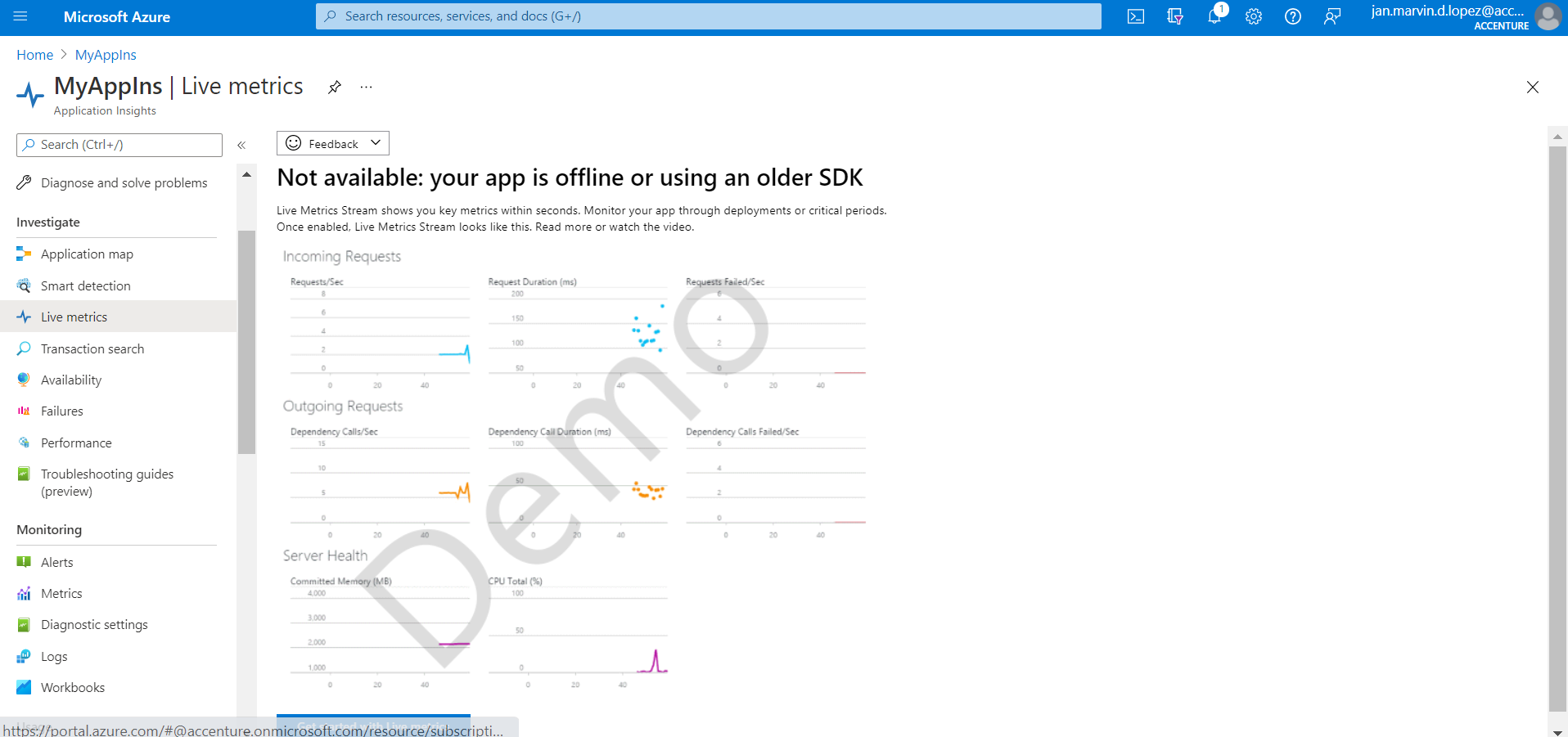
This is just a sample query where it shows analytics of the selected pages (Application Insights page, UBS - Home)

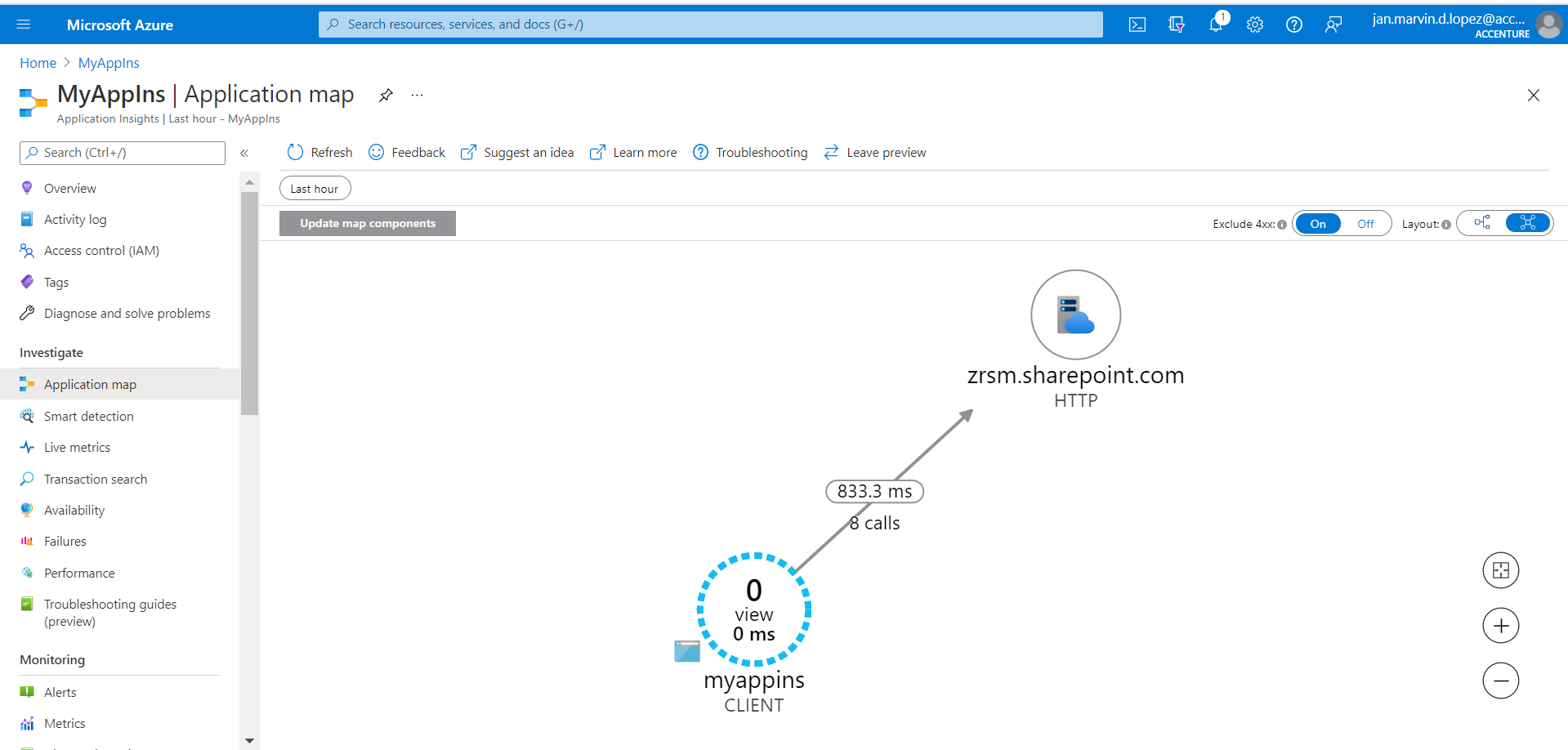
Things to consider:

* Costing? One Application Insight Resource per Site Collection (Azure’s policy which is costing is based on usage)
* This solution should be implemented at the start of production. Otherwise it won’t be able to capture or monitor previous events, variables, usage.
* Other feasibility of the pre-requirements provided by Floriana is still not confirmed. Should there be another POC? (Things like tracking users, video analytics)

Other Features of Application Insights:

Live Metrics monitoring - Monitor your live, in-production web application by using Live Metrics Stream (also known as QuickPulse) from [Application Insights](https://docs.microsoft.com/en-us/azure/azure-monitor/app/app-insights-overview). Ref: https://docs.microsoft.com/en-us/azure/azure-monitor/app/live-stream



Application Map - Each node on the map represents an application component or its dependencies; and has health KPI and alerts status. Ref: https://docs.microsoft.com/en-us/azure/azure-monitor/app/app-map?tabs=net

Reference:

Microsoft references

<https://docs.microsoft.com/en-us/azure/azure-monitor/app/nodejs>

<https://docs.microsoft.com/en-us/azure/azure-monitor/app/javascript-click-analytics-plugin>

deploy globally

https://sharepoint.handsontek.net/2019/02/23/deploy-application-insights-globally-on-modern-sharepoint/

https://github.com/joaoferreira/Application-Insights-Modern-SharePoint

https://sharepoint.handsontek.net/2019/02/19/how-to-add-application-insights-to-sharepoint-without-modifying-the-master-page/

https://www.c-sharpcorner.com/article/create-a-custom-master-page-in-html-for-Sharepoint-online/

https://blog.velingeorgiev.com/analytics-application-insights-sharepoint-classic-sites

https://docs.microsoft.com/en-us/azure/azure-monitor/app/website-monitoring

https://docs.microsoft.com/en-us/azure/azure-monitor/app/sharepoint

SP ribbon Can only be seen in classic view

https://answers.microsoft.com/en-us/msoffice/forum/all/how-do-i-get-the-ribbon-to-show-in-sharepoint/ad86fbf9-bfc0-48e3-8d36-78ea56a6c045

click analytics

https://docs.microsoft.com/en-us/azure/azure-monitor/app/javascript-click-analytics-plugin

<https://www.ejazhussain.com/2018/09/09/sharepoint-online-integration-with-azure-app-insights/>

Polish creation spfx trackpage

\*\*https://www.linkedin.com/pulse/how-use-spfx-application-customizer-do-telemetry-tracking-katoch

<https://www.sharepointnutsandbolts.com/2017/09/App-Insights-for-SPFx-and-provisioning.html>

Change elements.xml property using powershell

<https://blog.velingeorgiev.com/track-page-hits-sharepoint-framework-and-app-insights>

\*\*app ins video analytics

https://docs.microsoft.com/en-us/samples/azure-samples/media-services-javascript-azure-media-player-application-insights-plugin/media-services-javascript-azure-media-player-application-insights-plugin/