Microsoft Cloud for Retail

In a Day

Lab 02: Microsoft Clarity

Step-by-Step Lab

60 minutes

March 2022

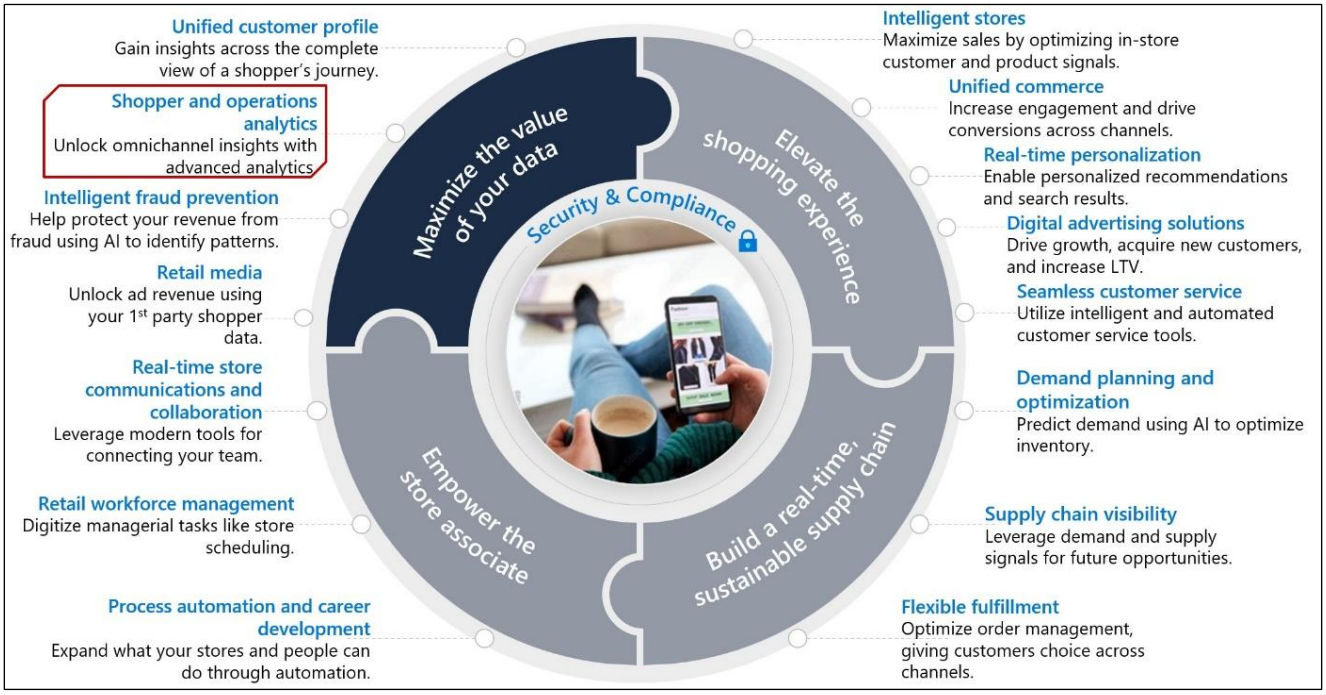
Contents

Overview

Shopper and operations analytics priority helps you unlock omnichannel insights with advanced analytics. With shopper and operations analytics, you can predict customer and operational needs, monitor and understand online engagement, and unify data integration, warehousing, and analytics.

You can **monitor and understand engagement**:

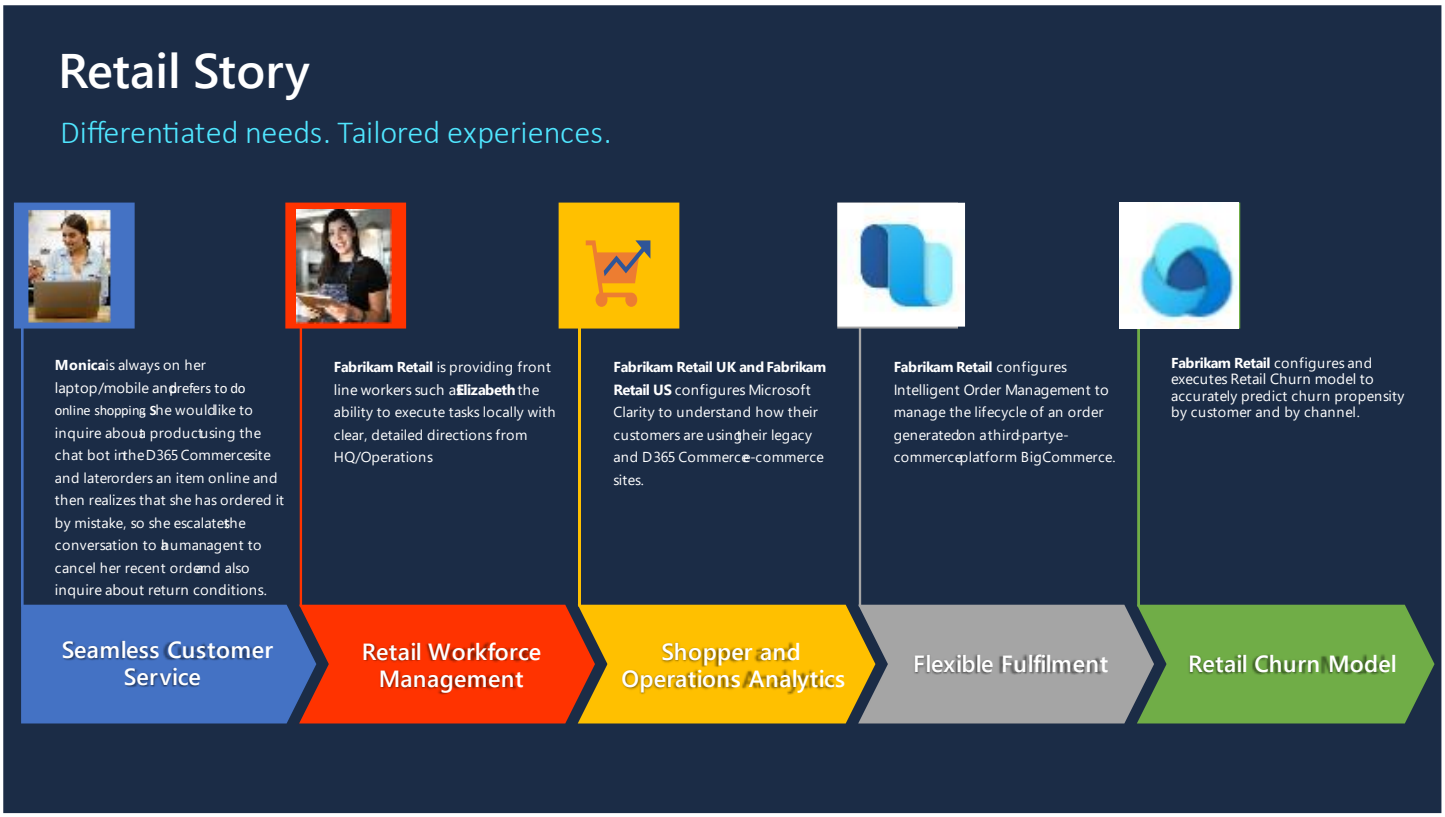
- Utilize the heatmap feature to see where your site generates the most clicks, what people are ignoring, and how far they're scrolling.
- Watch how customers are using your site with anonymized, hi-definition recordings and discover user frustrations by documenting rage clicks, dead clicks, and quick backs with behavior-focused insights allowing you to come up with solutions more efficiently.
- Understand customer engagement using built-in web and mobile analytics and return customer activity recognition, allowing you to create custom reports and views based on real-time customer behavior data.



Microsoft Clarity focuses on the **Maximize the value of your data, Shopper and operations analytics** priority scenario of Microsoft Cloud for Retail, by providing insight on your websites' usage patterns either as an e-commerce web shop built on Dynamics 365 Commerce, or a custom e-commerce website, or just a website for providing information about your physical shop.

Learning Objectives

Retail Story



In this lab, you will assume to be a system administrator of Fabrikam UK and Fabrikam US websites. Fabrikam US has recently migrated to Dynamics 365 E-Commerce but Fabrikam UK is still using an ASP.NET website *(for the sake of Azure deployment simplicity we provided a very simple multipage no image containing web application to represent Fabrikam UK website)*. You want to track, monitor and analyze the user behavior on both websites to provide a better user experience. For that purpose, you will be integrating both the websites with Microsoft Clarity.

In this module, you will learn how to do the following:

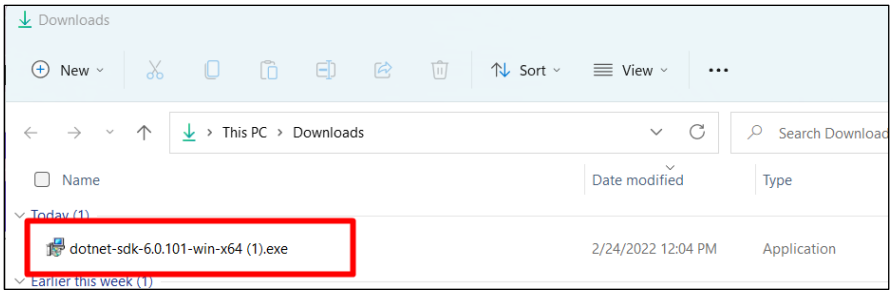
- How to create a Microsoft Clarity project
- How to integrate Microsoft Clarity with a custom website
- How to integrate Microsoft Clarity with Dynamics 365 E-Commerce Web Page
- How to interact with Microsoft Clarity dashboards, heatmaps and recordings

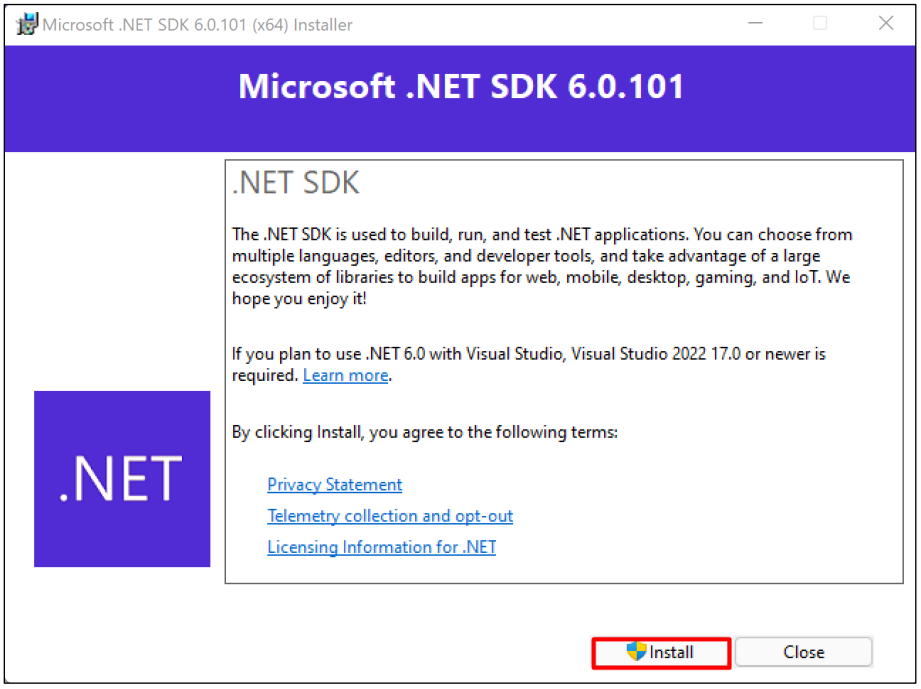
Bonus Learning Opportunity: (the

Since part of this lab needs an up and running sample website per attendee to do the integration, you will also learn how to deploy an ASP.NET web site to Azure App Service with Visual Studio Code.

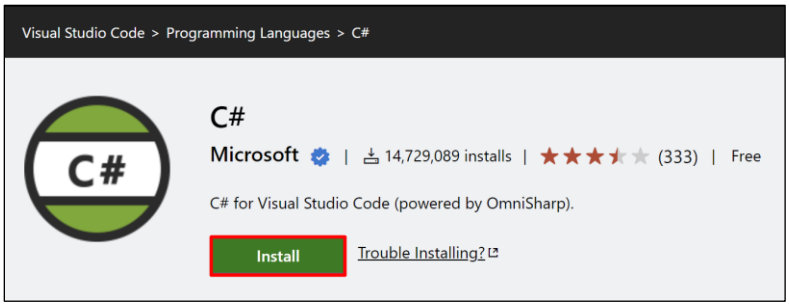
Prerequisites*

- Azure Subscription, and a resource group (will be provided by the instructor)
- Install the latest version of
  - Visual Studio Code
  - .NET 6.0 SDK (The link directly downloads the setup. Double click the Exe file, and click Install.)

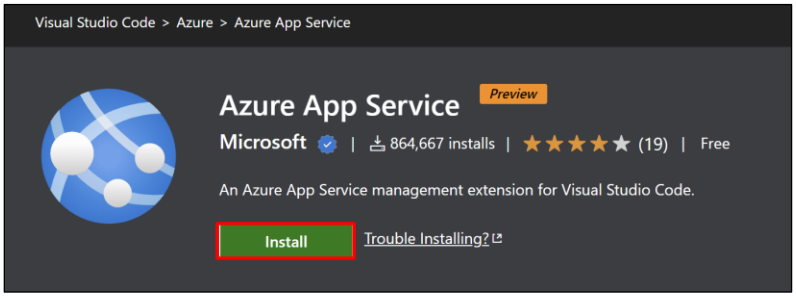




- Install below Visual Studio Code Extensions (you can click on green Install buttons to install each, this is required to be done **after** installing the **VSCode**):
  - C# - Visual Studio Marketplace



  - Azure App Service - Visual Studio Marketplace



- A local copy of sample Fabrikam Application, (*the URL for the up-to-date MC4R-Clarity.zip containing the code for the web site will be shared by your instructor.*)

***Note:** If you do not have the prerequisites done, you can still explore Clarity with Exercise 4, by using the publicly available Microsoft Clarity Demo Project.

Microsoft Clarity

Clarity is a behavioral analysis tool that helps you understand user interaction with your website. By using Clarity's analysis tools, you can monitor how your website is used on different platforms like PC, Mobile, Tablet and enhance your website for your clients and your business. Through Dashboard insights, Session Recordings and Heatmaps, Clarity helps you identify usability requirements of your web site by allowing you to study user behavior.

Exercise 1: Initial Deployment of Sample Custom Website

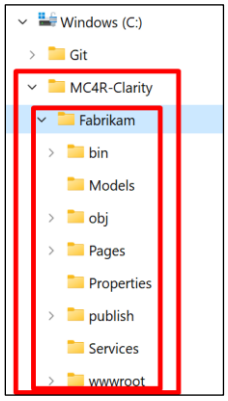
In this exercise, you will deploy a simple multipage ASP.NET website to a free tier Azure App Service on Microsoft Azure, using Visual Studio Code (VS Code) and its extensions.

Please make sure that all prerequisite installations are completed before you start the exercise.

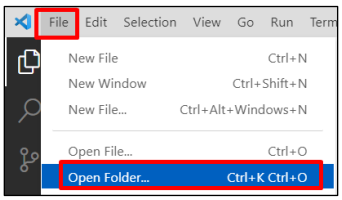
Task 1: Open the provided web site code with VS Code and inspect

In this task, we are going to look open the provided Website code and inspect its CSS and HTML code. We will find where the **<HEAD></HEAD>** and the **<HEADER></HEADER>** blocks of the HTML code are stored.

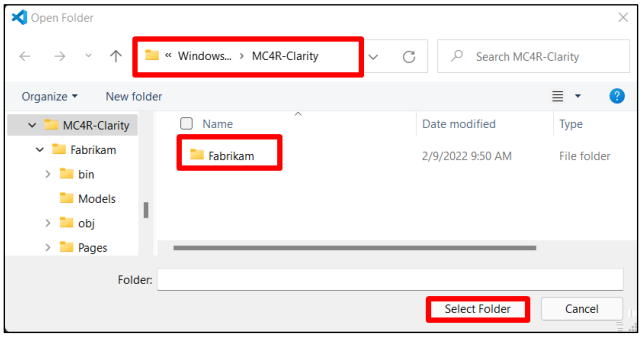
1. Download the provided website code from the location provided in prerequisites.
2. Create a directory called **MC4R-Clarity** under your **C:** drive and extract the project zip file to that path. You should see below folder structure when you expand the folder in your File Explorer's Navigation Pane tree view.



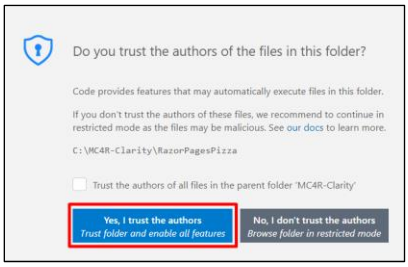
3. Open the **Fabrikam** directory under your **C:\MC4R-Clarity** directory with VS Code which you pre-installed already the required extensions and .NET SDK as listed in the prerequisites. To accomplish this task:
    a. Launch **VS Code,**
    b. Click on **File** menu,
    c. Click on **Open Folder…** menu item,



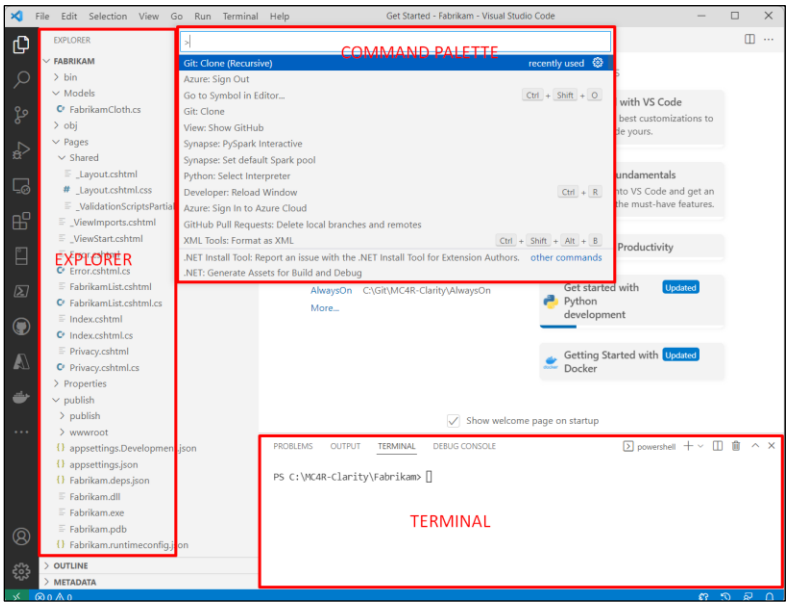
    d. Select **C:\MC4R-Clarity\Fabrikam** from the file browser and click **Select Folder.**



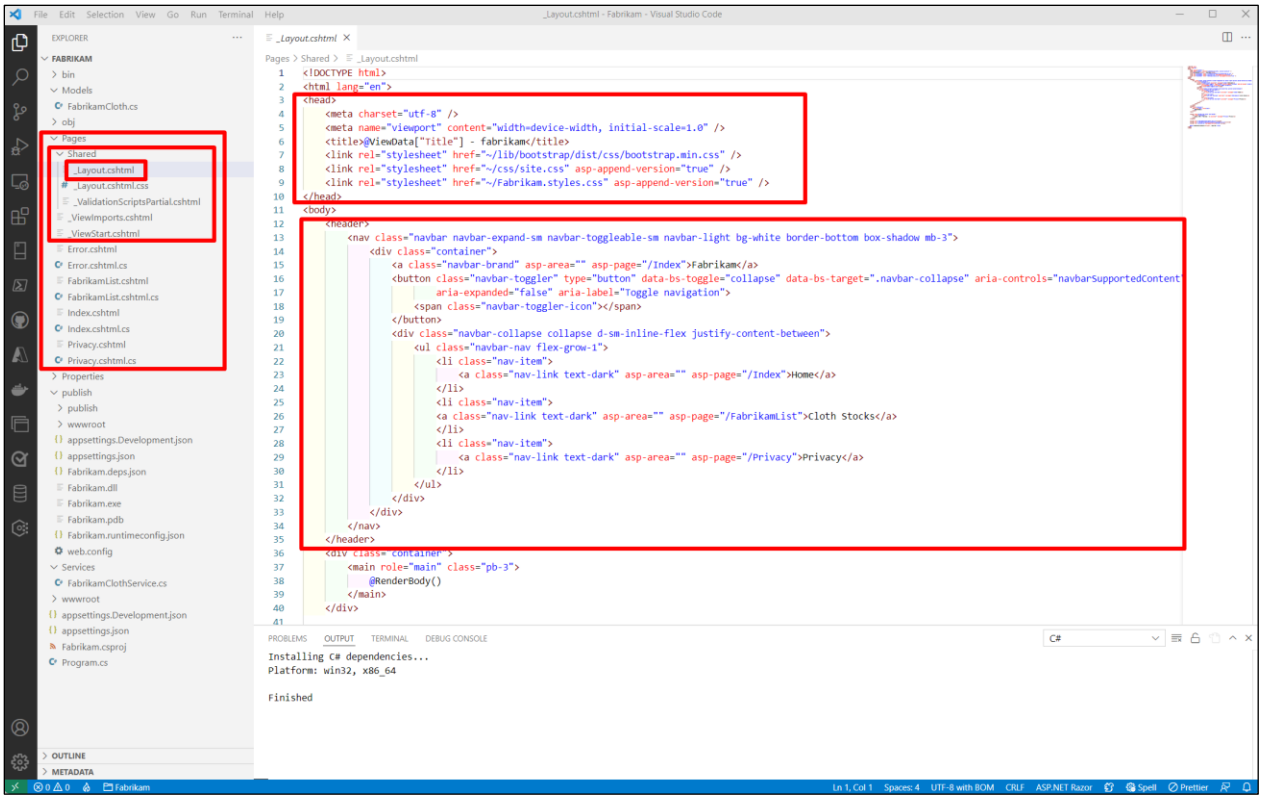
e. Click on the **Yes, I trust the authors** button on the popped-up trust notification.



f. Your VS Code screen should look similar to below snapshot:



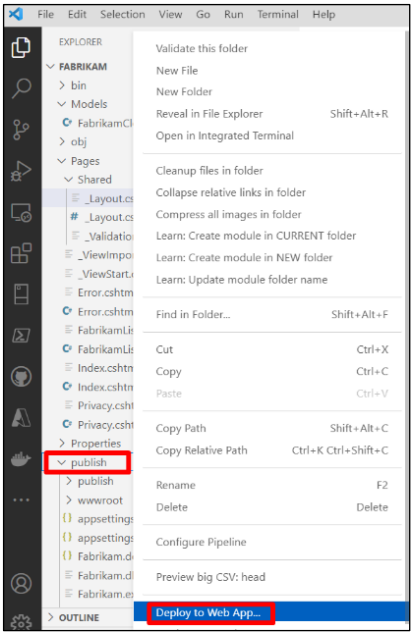
4. From the **EXPLORER** pane of VS Code expand the **>Pages** sub directory and **>Shared** sub directory under it respectively and click on **_Layout.cshtml**. This is where you should find the **<HEAD></HEAD>** section. This **<HEAD></HEAD>** section will be where we are changing to integrate any web site with Microsoft Clarity. ***<HEAD>*** *Section is the Document Metadata (Header) element of HTML Web development language, contains machine-readable information (metadata) about the document, like its title, scripts, and style sheets. It shouldn't be confused with the* ***<HEADER>*** *section which is a sub section of* ***<BODY>*** *section.*



Task 2: Deploy the web application as it is to Azure App Service.

In this task, you will deploy the sample application to Azure App Service using VS Code Azure App Service Extension. To do the deployment follow the steps below:

1. From the **EXPLORER** Blade, find the **>publish** folder, right click on **>publish** and click to the **Deploy to Web App...** menu item.

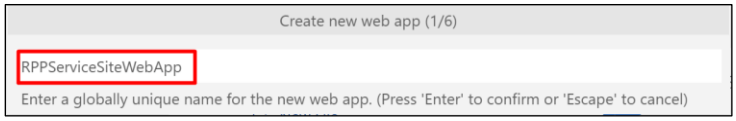


2. This will open the **Command Pallet** on the top mid-section of the VS Code and ask you to **sign in to your Azure Account**.
   Note: If you are in instructor led training. Sign in with the credentials provided by your instructor in the training information document.

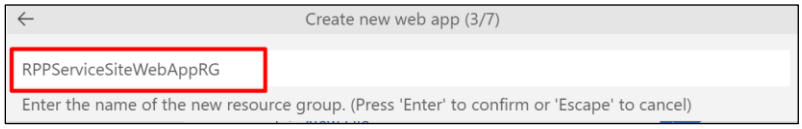
3. After signing in, from the **Command Pallet** you will be asked to select a **Subscription** from a list of subscriptions, select the subscription that your instructor guided you to.
   Note: If you are in instructor led training. Use the Azure subscription provided by your instructor in the training information document.

4. Next you will be asked to **Select Web App**. Click on **Create new Web App... Advanced**. *If you chose **Create new Web App...** you will not be able to specify which Azure region you want to create the service.*

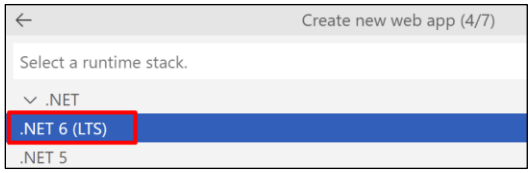
5. Enter a unique name for your web app(For Example: Your UserName). This name will be part of the URL that will be used to access your website, therefore it has to be unique. *The name can only contain letters, numbers or hyphens.*



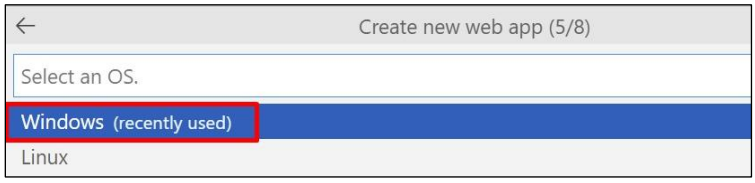
6. Select your choice of **Azure Resource Group** which will contain the App Service.
   Note: If you are in instructor led training. Use the Azure Resource Group provided by your instructor in the training information document.



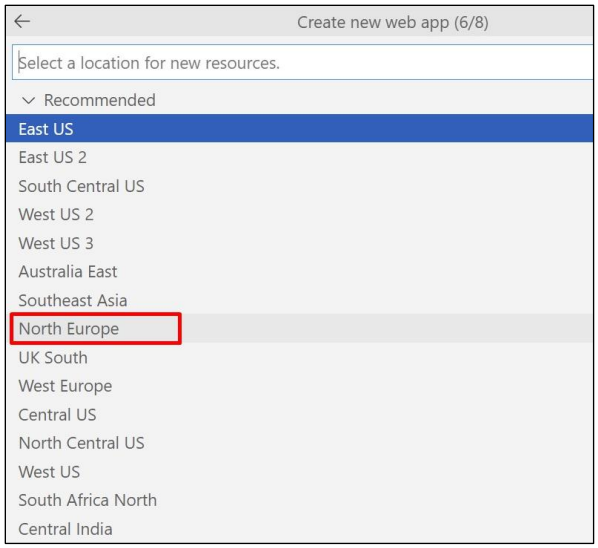
7. Select the run time stack, since our application is a .NET 6 application as stated in the prerequisites we will pick **.NET 6(LTS)**.



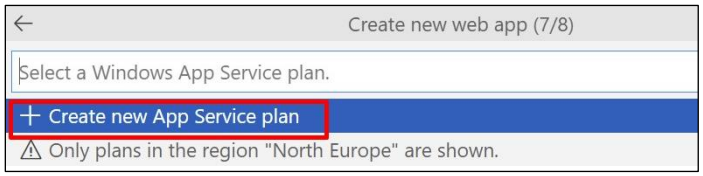
8. Pick your choice of **OS (Operating System)**, for our application we will be picking **Windows**.



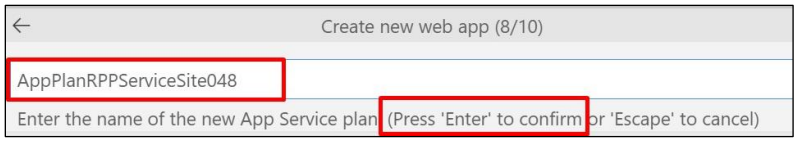
9. Select an **Azure Region** which will host the **Azure App Service**. Pick a region that is geographically closest to you to have the maximum network performance.



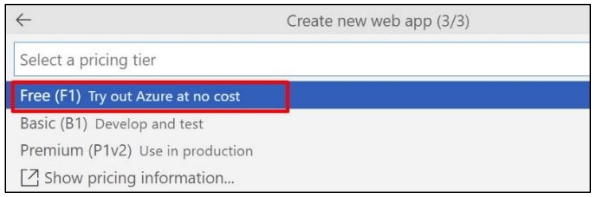
10. Click on **+Create new App Service** plan to create the app service plan for your app hosting.



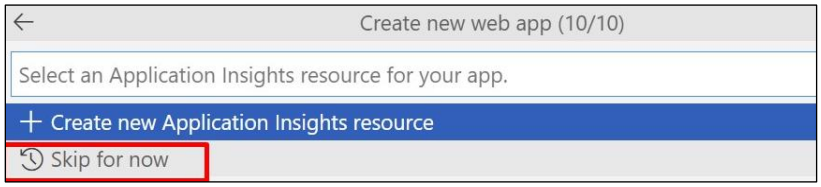
11. Pick a name for your Azure App Service plan type it and press enter.



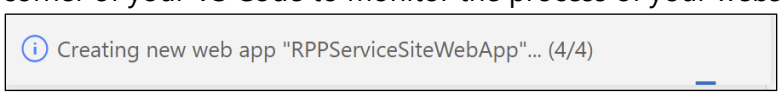
12. For this lab you do not need a very powerful hosting, so you can pick the **Free(F1)**



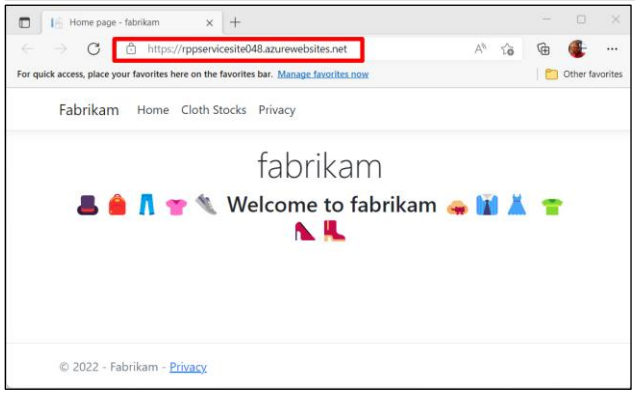
13. You will **skip the Application Insights creation** by clicking **Skip for now**



14. Once the setup is complete, you should see a notification similar to the one below at the bottom right corner of your VS Code to monitor the process of your website's deployment.



15. When the deployment is completed, you can click on **Browse Website** to access your deployment. Don't forget to **save the URL** information of the website.



16. If you see another pop-up window like below asking whether you will **deploy** this project **always** to the **same web app**, click **Yes**, this will make it easier to deploy when we do the necessary changes for the Clarity integration on the web app.

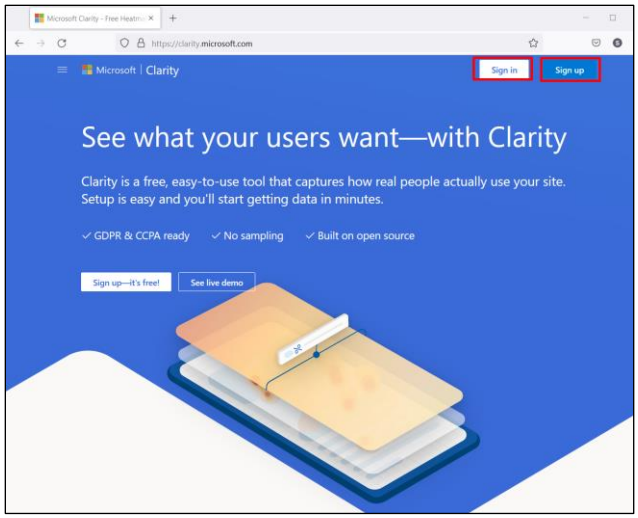
Congratulations! You have deployed your web site. Don't forget to **save the URL information of the website**.

Exercise 2: Configure Microsoft Clarity

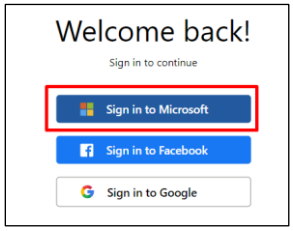
In this exercise, you will first create your Clarity project and configure the Clarity project and your website to complete the integration.

Task 1: Sign into Microsoft Clarity and create your project.

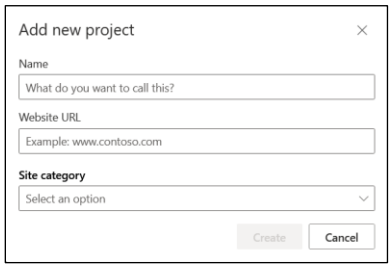
1. Go to https://clarity.microsoft.com/ and if you have a Microsoft Account either work or personal, Click to **Sign in** or **Sign up**. If you do not have any you can click **Microsoft Account** and create one for free. For the lab purposes you can use your work account since it is a free service.



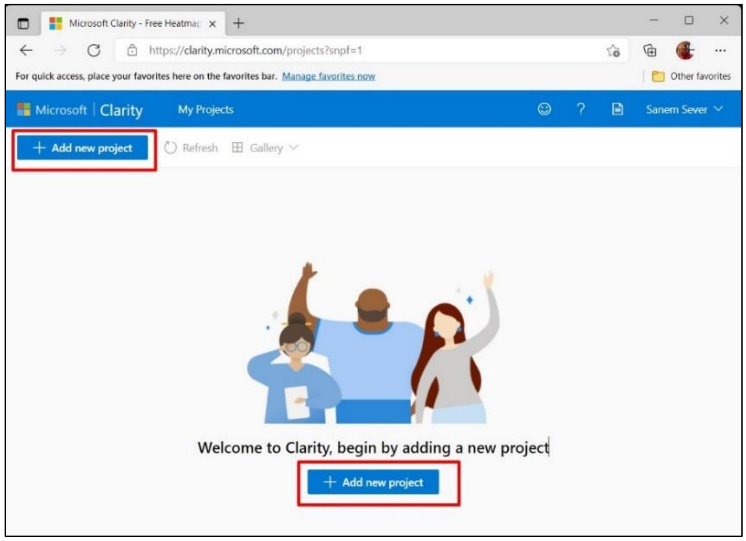
2. Click to **Sign into Microsoft**



3. You will be welcomed with below pop up to **Add new project**. You can either use this one

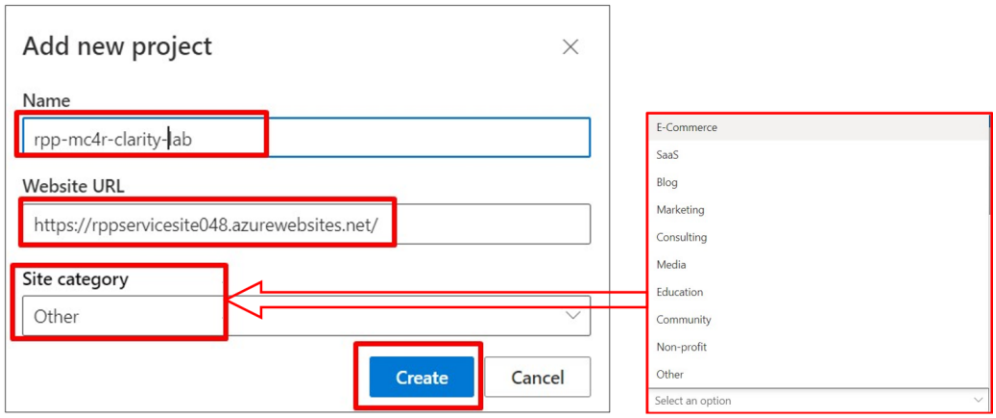


Or click **Cancel** or **X** and use any **+Add new project** buttons marked below:

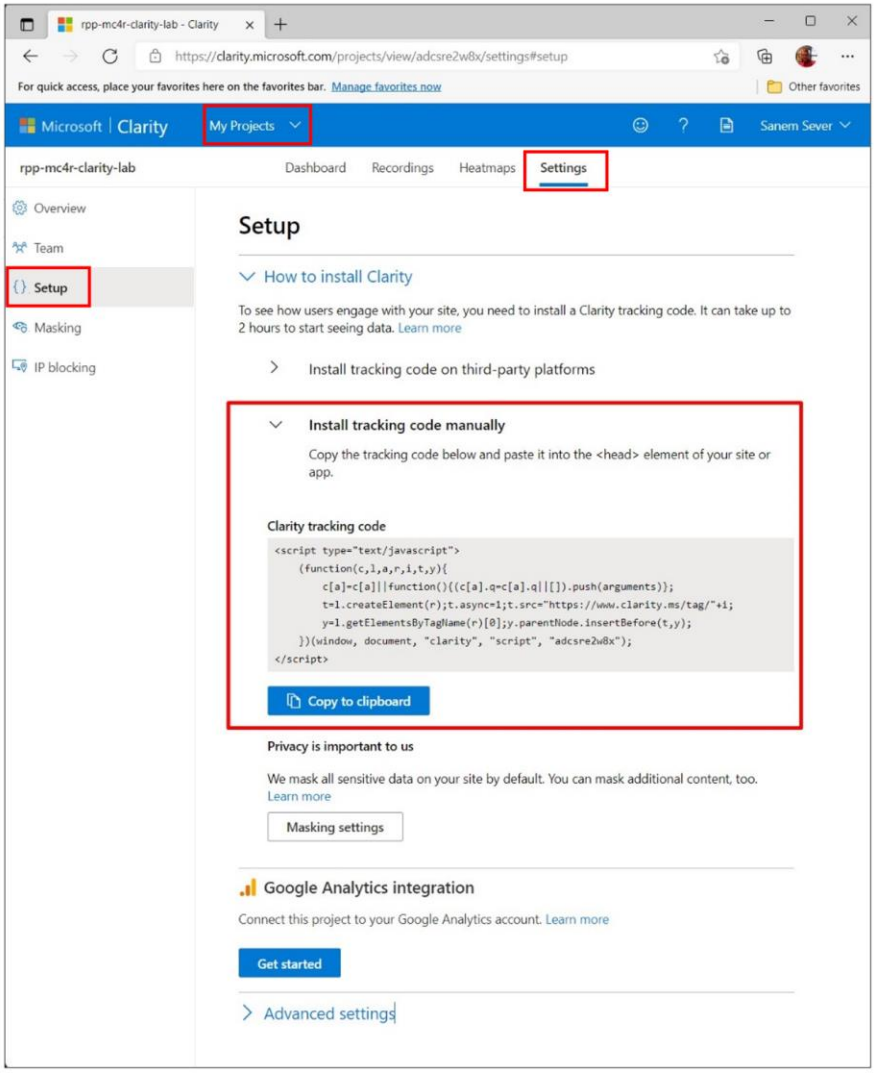


The same pop-up window to **Add new project** will come from all options above.

4. Start filling in the form on the pop-up window:
    a. Pick a name for your Clarity project and type it to the **Name**
    b. Paste the **website URL** you already saved on Exercise 1, Task 2, Item 15.
    c. Pick a website **category**. You can pick other for this sample application*, but there are various other categories that you can pick from like****, E-Commerce, SaaS, Blog, Marketing, Consulting, Media, Education, Community, Non-profit, Other***.
    d. Click to **Create.**



5. After you click on **Create** you will be forwarded to the **Setup** section**.** Copy the **Clarity tracking code** provided to you. We will be using the code shared in the **Install tracking code manually** section to integrate clarity into our custom website.

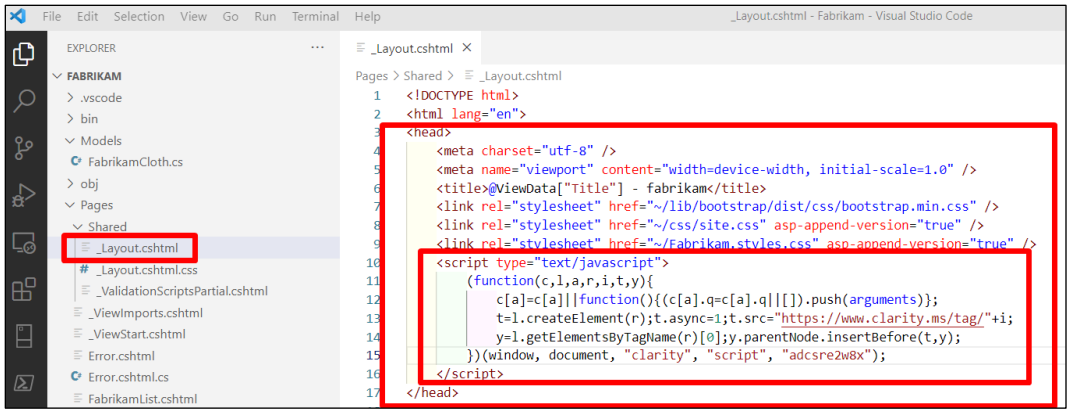


*If you forget to copy the tracking code that is perfectly fine. You can go to **My Projects**, pick the project that you want to integrate with your website click on **Settings** then from there click on **Setup**. In the **How to install Clarity** expendable section you will find the **tracking code** in **the Install tracking code manually** section.*

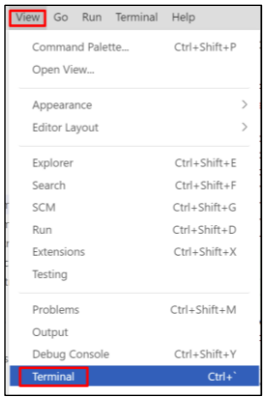
***!** Read the **warning** under the Install tracking code manually section, it states that the **data flow from your application to Clarity Portal** can take **up to 2 hours**. Therefore after finalizing Exercise 2 we will continue with other labs and come back to Exercise 3 later. The timing and order will be shared by your instructor.*

Task 2: Embed/Install the Clarity Tracking Code to the custom website and redeploy the website

1. Go back to VS Code and expand the **>Pages** sub directory and **>Shared** sub directory under it respectively and click on **_Layout.cshtml**. Locate the **<HEAD></HEAD>** section. This is where you will integrate the tracking code. Paste the tracking code as the last item in the **<HEAD></HEAD>** section as shown with the blue box on the below snapshot and save with **CTRL+S.**



2. Check locally whether the web application still works properly using **Terminal** section of VS Code. *If you cannot see the Terminal window you can go to **View Menu** and click on the **Terminal Menu item** to enable it.*



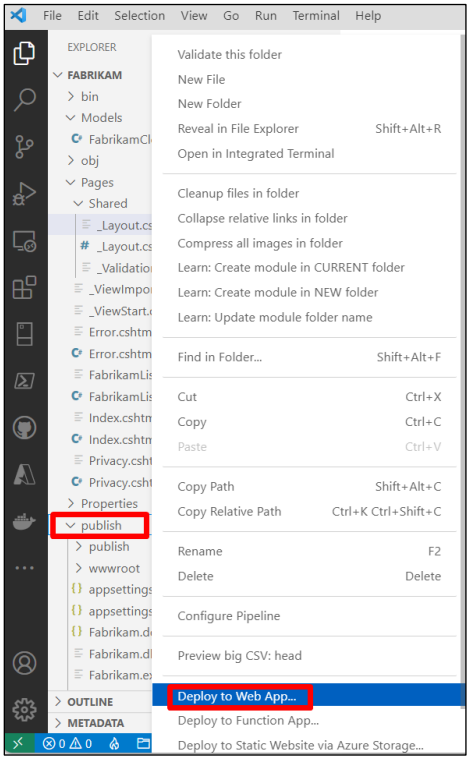
   a. Type **`dotnet run`** in the terminal screen and press enter to see the website building without an error after our change. *The result will list local host URLs. You can click one and reach your website from local installation. You can press Ctrl+C to shut the local instance down*

```
PS C:\MC4R-Clarity\Fabrikam> dotnet run
Building...
info: Microsoft.Hosting.Lifetime[14]
      Now listening on: https://localhost:7203
info: Microsoft.Hosting.Lifetime[14]
      Now listening on: http://localhost:5009
info: Microsoft.Hosting.Lifetime[0]
      Application started. Press Ctrl+C to shut down.
info: Microsoft.Hosting.Lifetime[0]
      Hosting environment: Development
info: Microsoft.Hosting.Lifetime[0]
      Content root path: C:\MC4R-Clarity\Fabrikam\
```

   b. To build and create the Clarity integrated website executables type **`dotnet publish -c Release -o ./publish`** and press enter

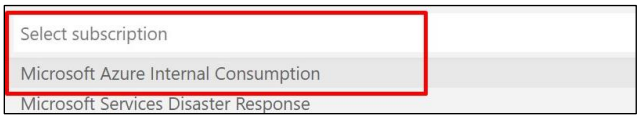
```
PS C:\MC4R-Clarity\Fabrikam> dotnet publish -c Release -o ./publish
Microsoft (R) Build Engine version 17.0.0+c9eb9dd64 for .NET
Copyright (C) Microsoft Corporation. All rights reserved.

  Determining projects to restore...
  All projects are up-to-date for restore.
  Fabrikam -> C:\MC4R-Clarity\Fabrikam\bin\Release\net6.0\Fabrikam.dll
  Fabrikam -> C:\MC4R-Clarity\Fabrikam\publish\
PS C:\MC4R-Clarity\Fabrikam>
```

c. To redeploy new version of the website, again go to **>publish** in the **EXPLORER** blade and right click, find **Deploy to Web App…** menu item and click.

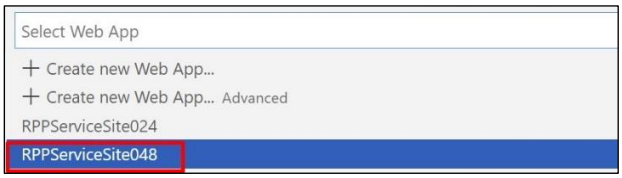


d. If you did say **Yes** when VSCode asked "Always deploy same webapp" below window will pop-up. Click **Deploy**, if you do not see this pop-up continue from item **e.**

e. If you did not sign out from the Azure Account that you used previously VS Code will ask you to select your **subscription** again, pick the subscription that you used before.



f. Next VS Code will ask you to select the Web App to redeploy. Choose the one you previously deployed to.



g. Now browse to your website both from your mobile devices and your computer multiple times, click on the URLS, play with the form on the Cloth Stocks page and add & delete some cloths. This will create the usage logs and recordings which Clarity will sniff to build its Dashboards, Heatmaps and Recordings. You can ask your friends to play with your website as well.

! *As you have read in Exercise 2, Task 1, Task Item 5 note; the logs to sync with clarity can take from 30 minutes to 2 hours. Therefore, after completing this exercise, you will start the remaining labs, and after completing them you will come back to this lab and continue with Exercise 3.*

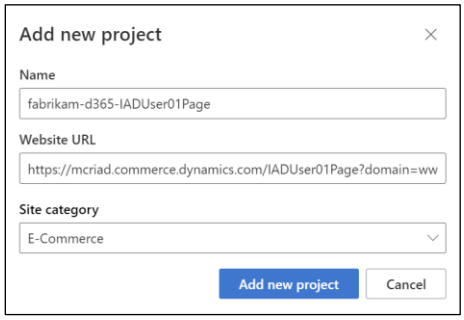
Congratulations! You have integrated your custom website with Microsoft Clarity.

Exercise 3: Integrate Microsoft Clarity in Dynamics 365 E-Commerce

In this exercise you will integrate Microsoft Clarity with Dynamics 365 E-Commerce. During the setup process, you need to use the live production domain URL associated with your Commerce site. You should have already created a new web page on Dynamics 365 E-Commerce in Lab 1. You will use here that web page.

Task 1: Create Clarity Project for your Dynamics 365 E-Commerce page.

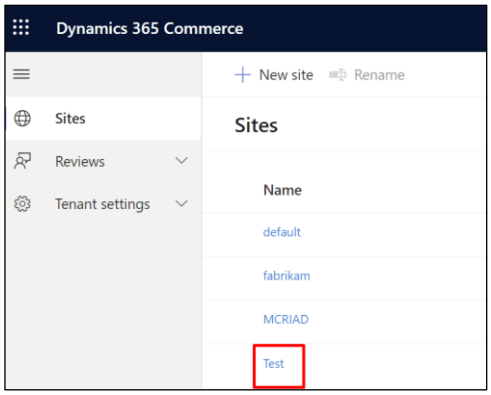
1. Go to https://clarity.microsoft.com/ and sign in.
2. Open a private window since you will be using provided credentials for this lab.
3. Create a new project for tracking your Dynamics 365 E-Commerce page using your pages direct URL.



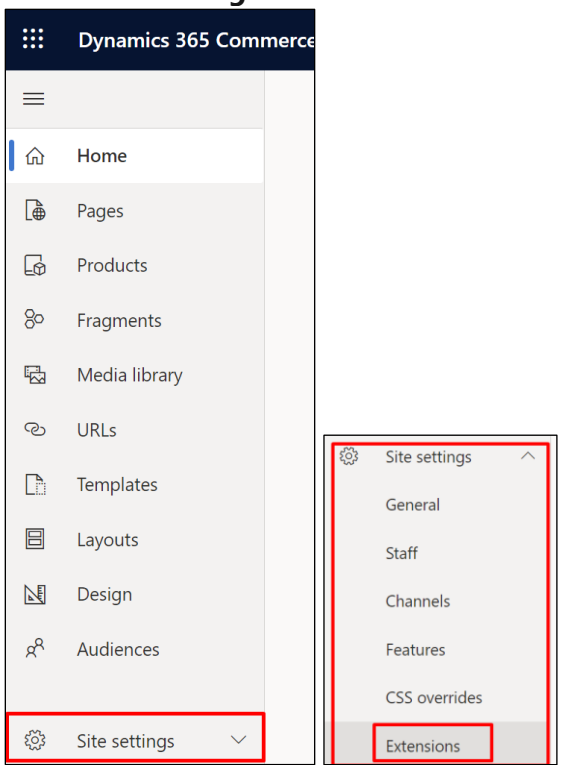
Task 2: Configure Content Security Policy for Clarity

For Clarity to function on a Dynamics 365 E-Commerce site, some CSP directives must be configured to allow Clarity resources to be called. This configuration may be performed initially due to the lab environment setup by your instructor, Follow the steps below to verify that the configuration is complete, if not you can use again the same steps to configure.

1. Open the D365 Commerce site builder URL in an in-private window.
   Note: If you are in instructor led training. Use the Commerce Site Builder URL and credentials provided by your instructor in the training information document.
2. Click on Test project.

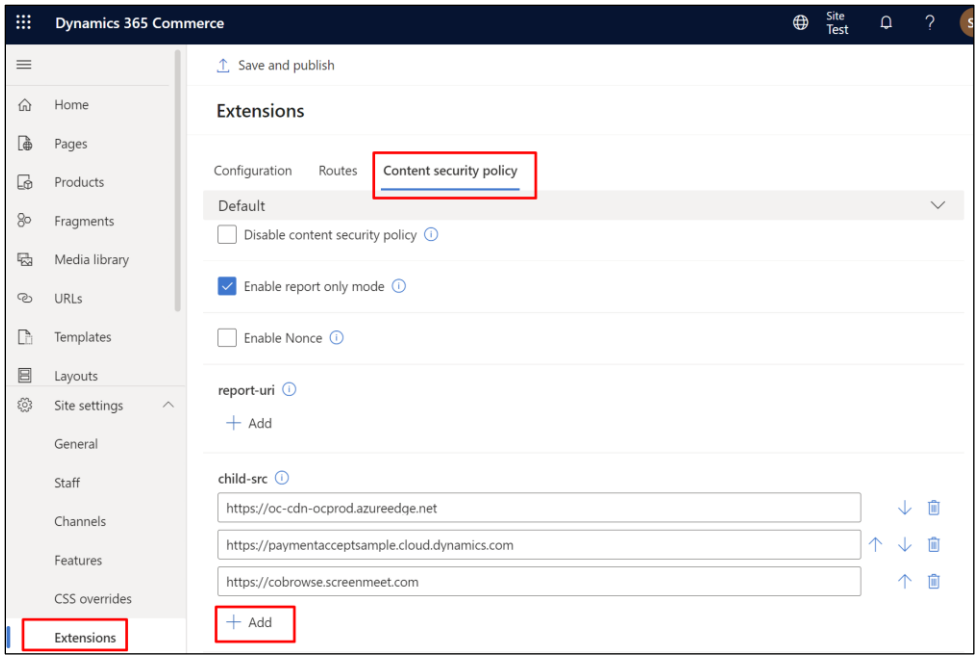


3. Select **Site Settings > Extensions**.

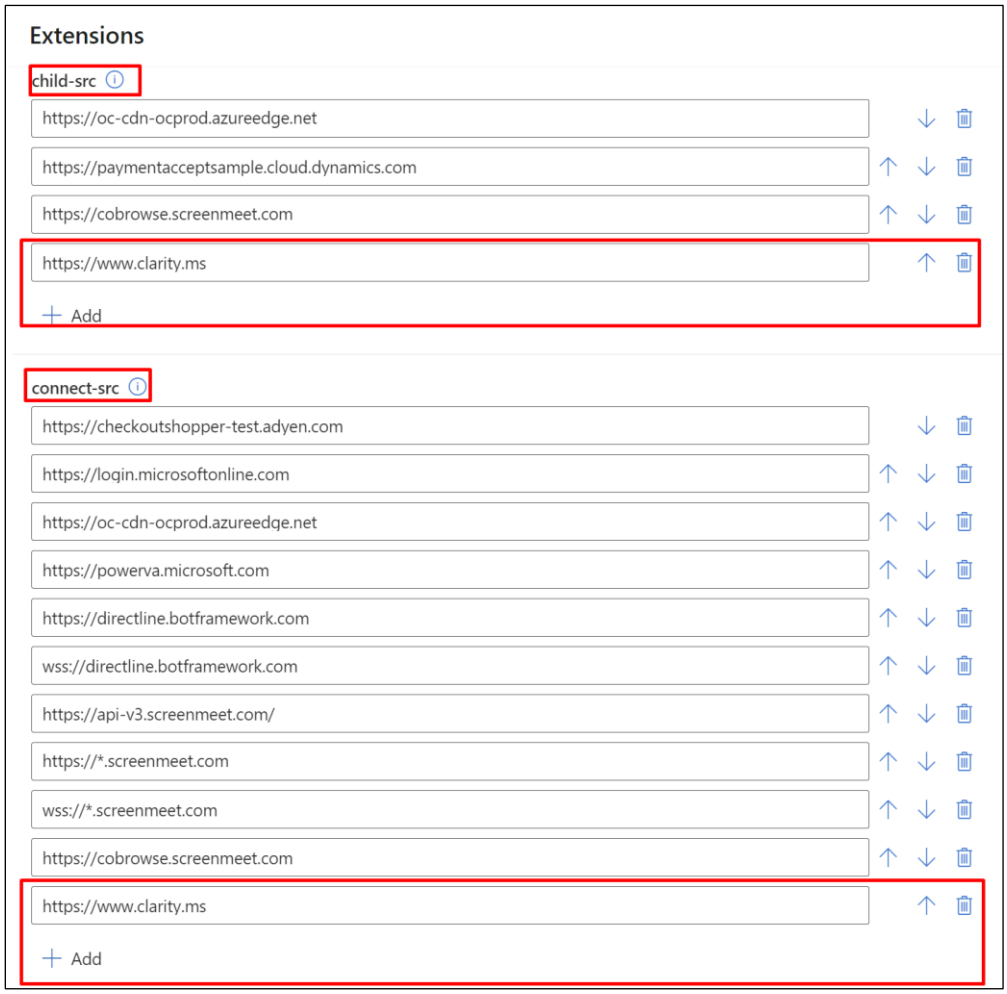


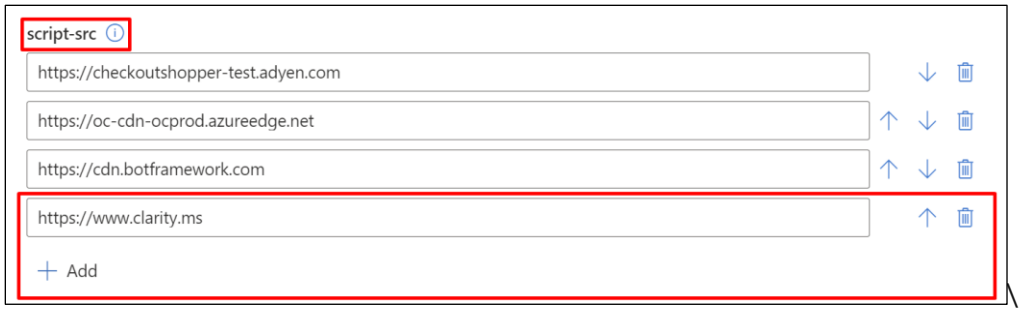
Note: If you are in an instructor led training, you can skip the remaining steps in this task because these are done only once per site and your instructor would have already configured these in your site.

4. Click on the **Content security policy tab**, scroll to **child-src, connect-src** and **script-src** directive sections, check if **https://www.clarity.ms**. Is added in all three sections.
5. If not click **Add** and enter **https://www.clarity.ms** in all three sections, **child-src, connect-src** and **script-src**.



6. The Extensions should look like below:



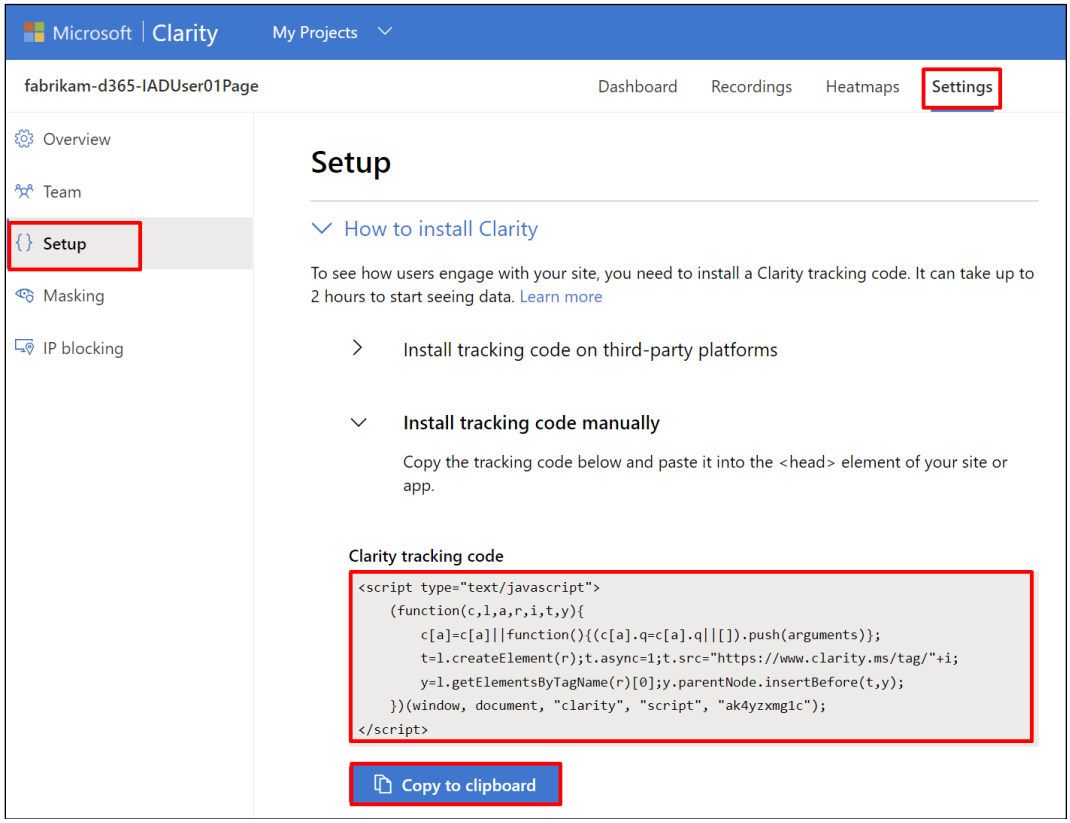


7. Click on **Save and Publish** at the top of the Extensions blade to save your changes.

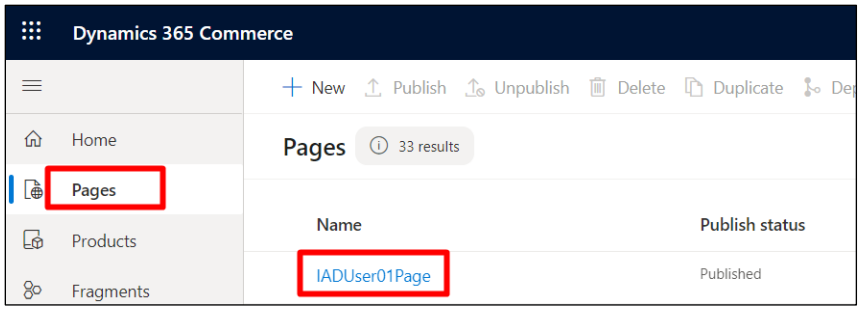
Task 3: Embed Clarity tracking script code into your site page

You can embed Clarity tracking script code into any Commerce site page that you want to track with Clarity.

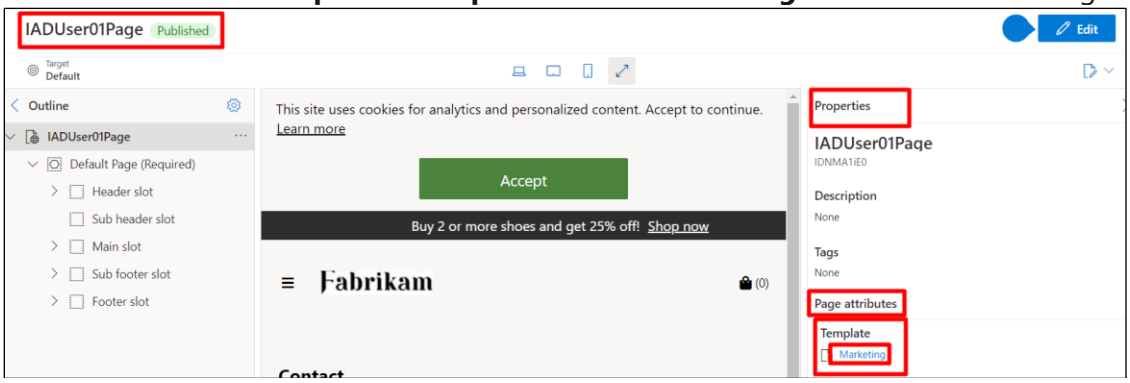
1. Copy the tracking code from **Microsoft Clarity Settings, Setup, How to install Clarity, Clarity tracking code**.



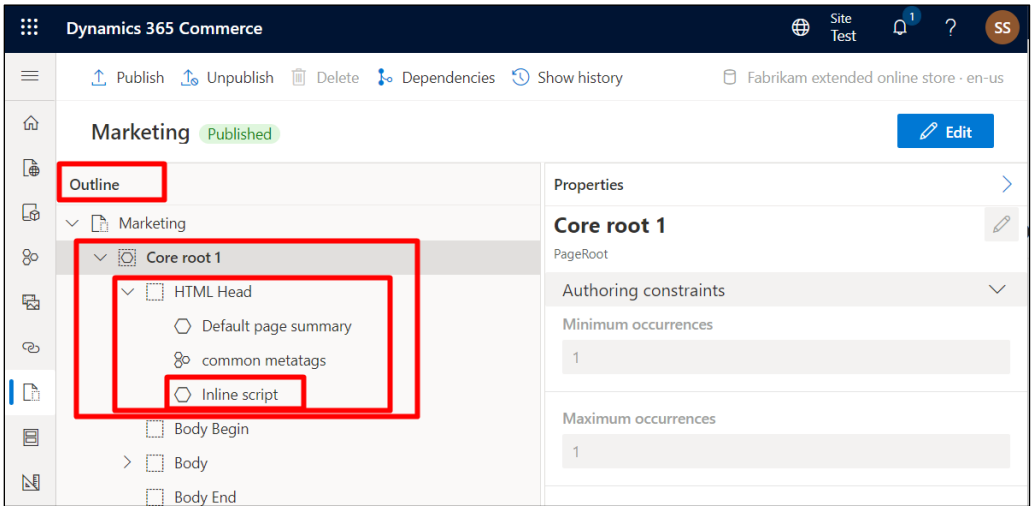
2. In the Commerce site builder, Test project. from **Pages** click on to your page. Your Page will be named IADUserXXPage, where XX refers to your User Number.
Note: If you are in instructor led training. Use the page which was created as part of Lab 01(Seamless Customer Service).



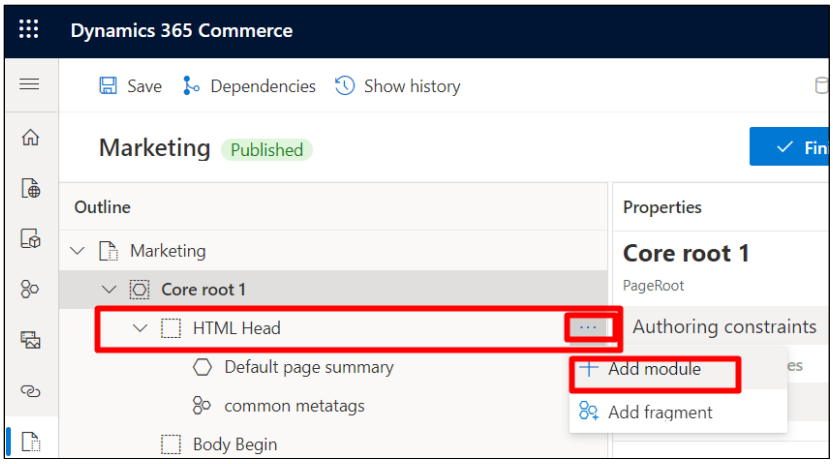
3. For you to be able to add the tracking script on a D365 E-=Commerce page, the **Template** that the **page** is inheriting should support **inline script**. To verify that,
    a. Click on the shown **Template** in **Properties** blade under **Page attributes** of the Page view.



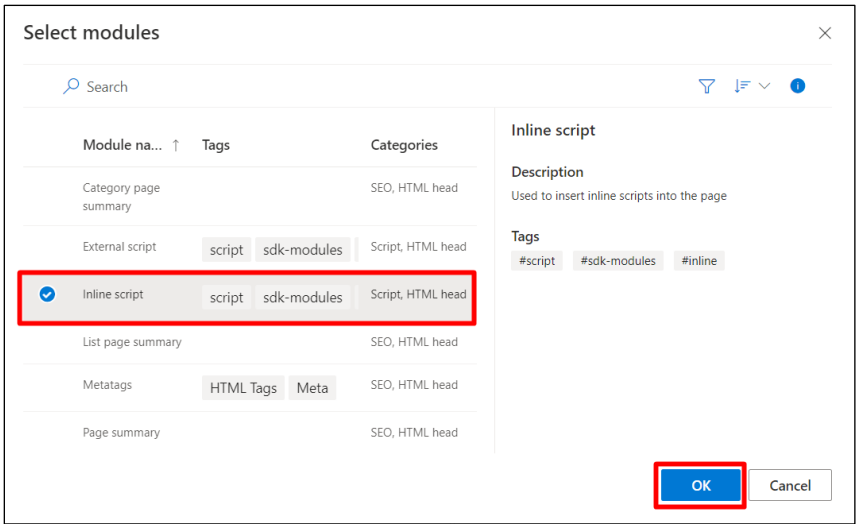
    b. From the **Template** view, go to **Outline** blade, Click on **Core Root 1,**expand **HTML Head** and check if you have **Inline script.**



    c. If you do not have **Inline script,** click on **Edit.**
    d. From **Outline**, **Core Root 1, HTML Head** click on **...** and **+Add module.**



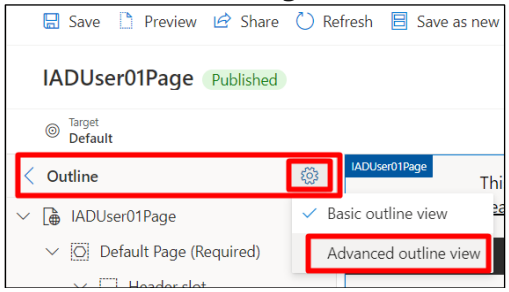
e. From the pop-up window select **Inline script** and click **OK.**



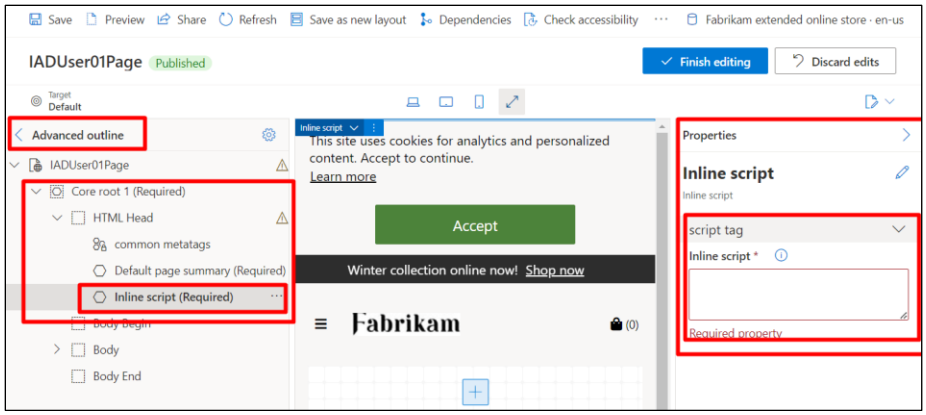
f. Now you should see **Inline script** under **HTML Head**. Click **Finish editing** and then **Publish** your changes.

4. Go back to your Page and click **Edit.**

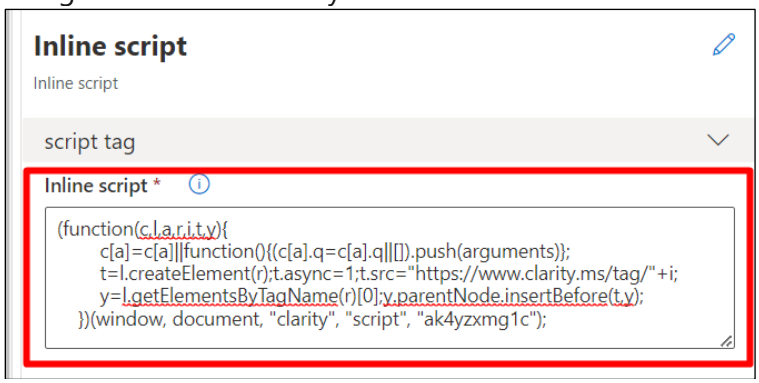
5. Go to **Outline, Settings** and switch to **Advanced outline view.**



6. Click on the **Inline Script** from **Core Root 1, HTML Head** on the advanced view of **Outline** blade. Now you should see the **Inline Script** in **Properties** blade.



7. Paste the tracking script you copied from **Microsoft Clarity** in the **Inline script** text box. Make sure to remove the surrounding **<script type="text/javascript">** **</script>** tags if you copied the script string. IT will be added by the builder.



8. Click **Finish Editing.**
9. Click **Publish** and publish the page.

Congratulations! You have integrated your Dynamics 365 E-Commerce website page with Microsoft Clarity. Now you need to create some traffic on your Dynamics 365 E-Commerce website page like you did with the custom page to be able to observe it Microsoft Clarity.

Exercise 4: Explore your websites usage data with Clarity

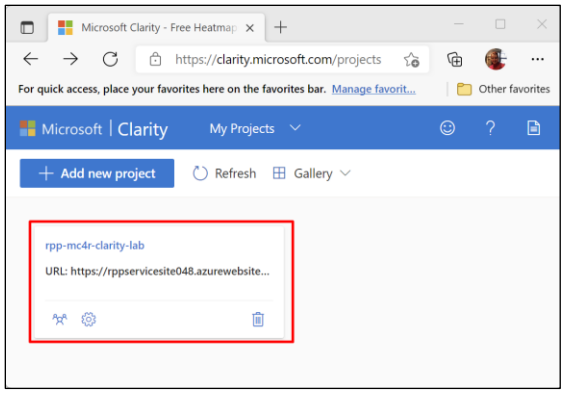
To start this exercise, you should have created some traffic on your website and waited around 30 mins minimum to reflect the data. If you do not have the prerequisites done, you can still follow the steps of Exercise 4 by using the publicly available Microsoft Clarity Demo Project.

In this final exercise, you will wander around the Microsoft Clarity capabilities to monitor and understand the usage patterns of your own application.

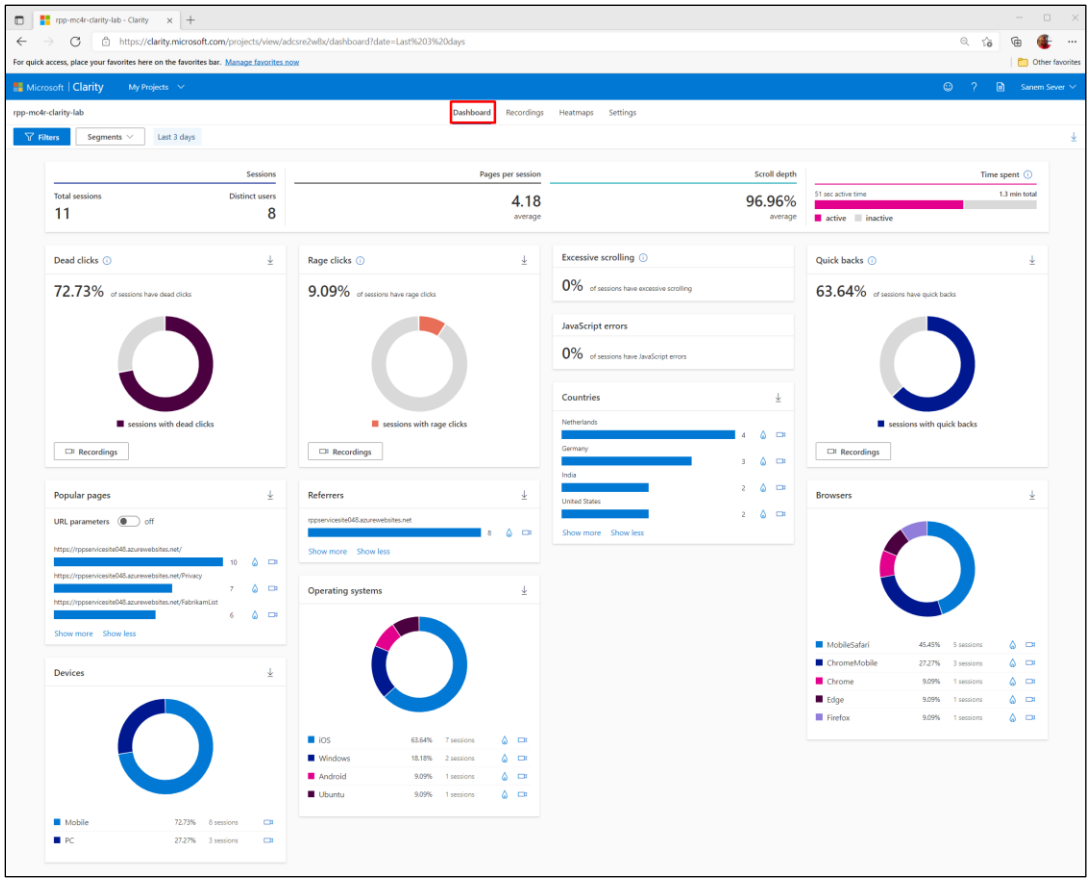
Task 1: Discover Clarity Dashboard capabilities

In this task, you will sign-in to the Microsoft Clarity and discover the main dashboard capabilities.

1. Go to https://clarity.microsoft.com/ and sign in.
2. Click on your project



3. Click on Dashboard view and observe the contents of the view

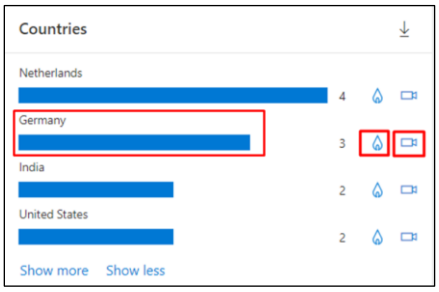


Here you can see:

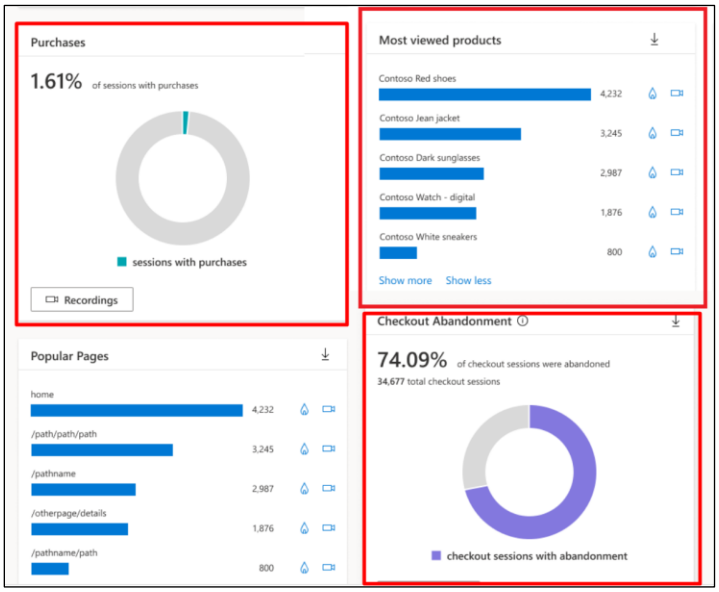
a. Number of sessions, number of distinct users, pages browsed per session
b. Whether the pages are scrolled fully
c. How much time people spend on your page and how much of the time was active
d. Dead Clicks: ratio of sessions where user clicked or tapped with no effect / go nowhere
e. Rage Clicks: ratio of sessions where user rapidly clicked or tapped in the same small area
f. Quick Backs: ratio of sessions when user navigated to a page then quickly returned to the previous one
g. Javascript errors: ratio of sessions which had a Javascript error
h. The ranked popularity of your application's sub pages
i. The main referrers to your site (which sites users are coming to your page from)
j. Device Type distribution

k. Operating Systems distribution
l. Browser distribution
m. Countries distribution

You can interact with these Dashboard view contents(modules) and filter the data directly from them as well. You can also jump to other views like Recordings and/or Heatmaps by clicking camera and fire icons shown on the module. The detailed information on Recordings and Heatmaps that will be learned in future tasks. Clicking directly on the country itself allows you to filter your dashboard data to it as well, so you can see details like Referrer breakdown for sessions in a specific country.



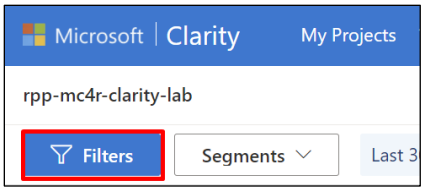
4. For select Clarity projects (see below), you can also see additional e-commerce information:
   a. % Sessions ended with purchase (all Shopify sites)
   b. % Checkout sessions abandoned (all Shopify Pro sites)
   c. Most viewed products (all sites using Product JSON.LD)



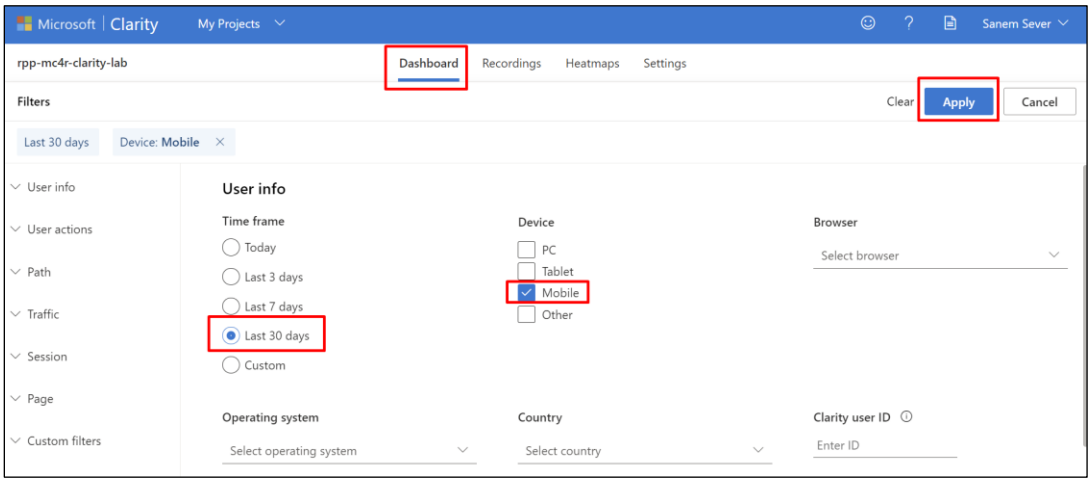
Task 2: Discover Clarity Filtering capabilities and Segments

In this task you will learn how you can use Filters and how you can save Filters by creating Segments. Filters can be used in all Capability views of Clarity, like Dashboards that you just have seen or Recordings and Heatmaps that you will see in subsequent tasks.

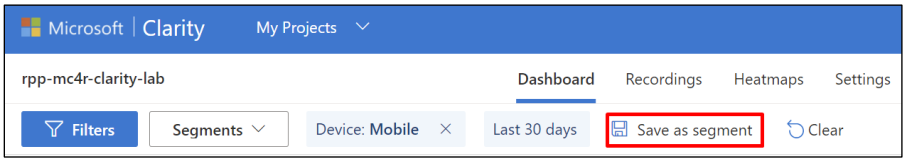
1. **Use filters** to view data coming from specific interest groups and create **segments** to have specific views for specific purposes.
   a. Click to Filters:



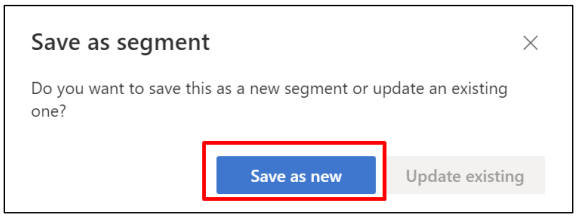
b. Change the time frame and pick **30 days** from **User Info>Time Frame**. From **User Info>Device** pick one of the existing options for example **Mobile** *(If you have more clicks from any other options, you can pick that one as well)* and click **Apply**.



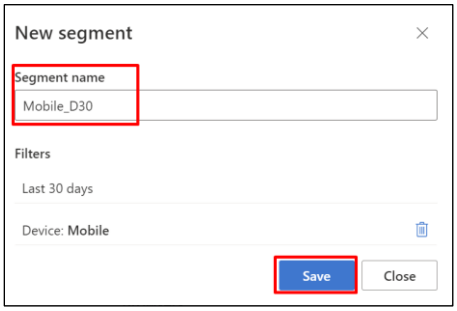
c. Click on **Save as Segment**



d. You will receive a pop-up window asking whether you want to create a new segment or update an existing one. Click on **Save as new.**

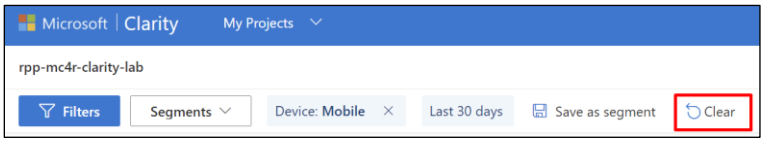


e. Pick a name for your segment and write that in the Segment **Name** text box then click **Save**.

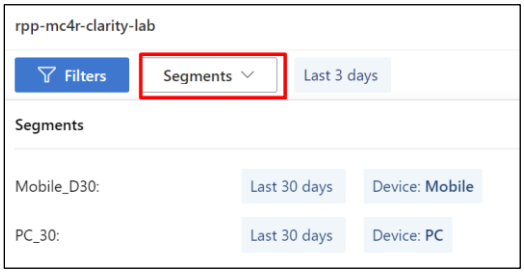


f. Do the same for **PC** Device option.

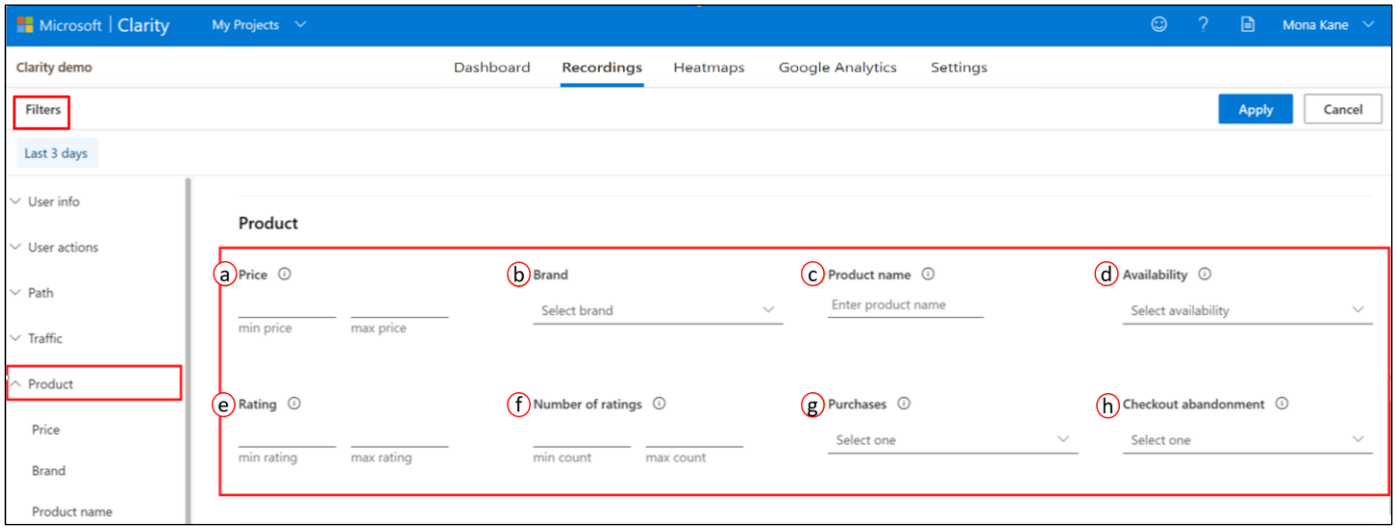
g. Click on **Clear** to clean the remaining filters from the dashboard and return to default view.



h. Now you can see the and change the segments from **Segments Drop Down list**



2. If you integrate with sites that use Product JSON-LD or are hosted on Shopify, you can view additional Product filters:



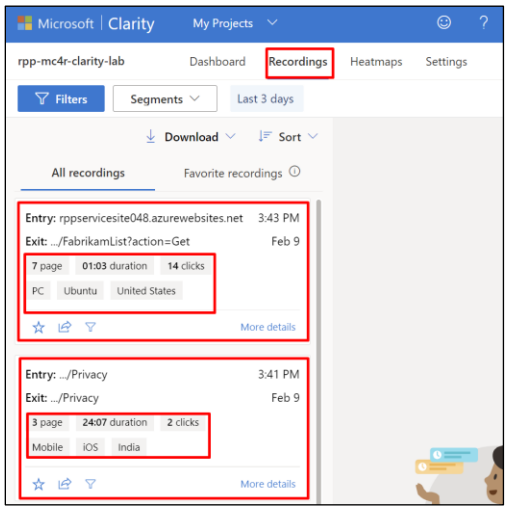
a. **Price**: Select data for the price value of the product viewed based on the currency used on your site. Choose to view data based on the minimum and maximum values you input.
b. **Brand**: Select data that includes the product brand. This filter helps you to view more from the specific brand.
c. **Product name**: Select data based on a product name.
d. **Availability**: Select data that includes whether a viewed product was in stock or not. The dropdown list includes In stock, None, Out of stock.
e. **Rating**: Select data with average user rating. Enter the star min and max rating from 1 to 5.
f. **Number of ratings**: Select data with the total number of users rated. Enter a numeric input in min and max count.
g. **Purchases**: Select the sessions where site visitors did or didn't purchase a product. Choose to view sessions based on "Yes" or "No".
h. **Checkout abandonment**: Select the sessions where the user abandoned the checkout process at a specific step. Choose to view sessions based on the step in the checkout process that was abandoned.

***Note**: **Filters a-f** are available for site instrumenting with Product JSON-LD. **Filter g** is available for Clarity projects of Shopify sites. **Filter h** is available for only Shopify plus sites.*

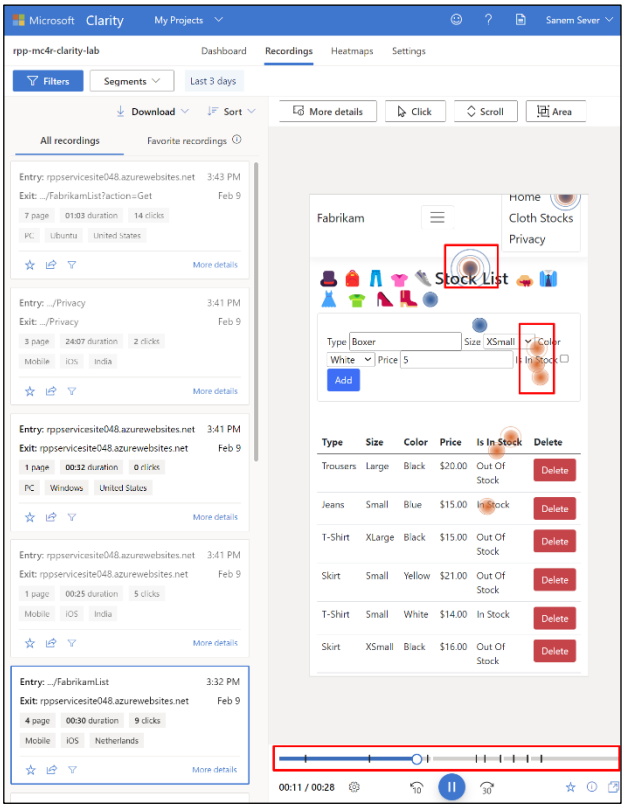
Task 3: Discover Clarity Recording capabilities

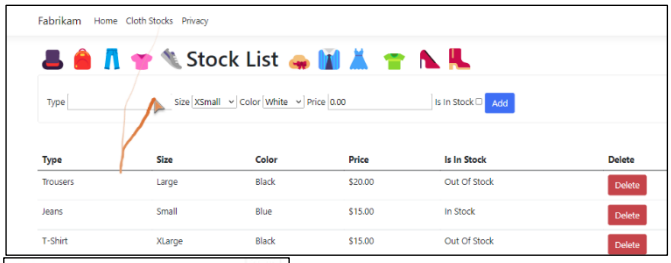
Another important capability of Clarity is showing you recordings of user sessions.

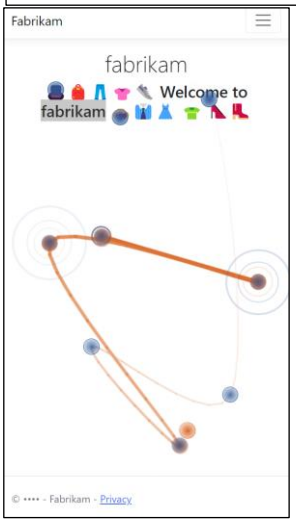
1. Go to https://clarity.microsoft.com/ and sign in if required.
2. Click on your project
3. Click on Recordings view and observe the contents in the view. *The recordings are tagged with information like the Originating **Country**, **OS**, **Device**, Num of **Clicks**, Num of **visited Pages** and Session **Duration**. You can also filter them manually or by creating Segments.*



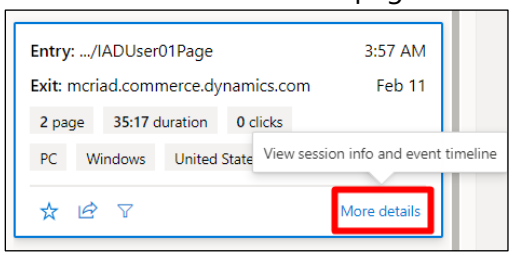
4. Select the Mobile Segment and **view one of the recordings**. You can watch **clicks and scrolls** happening, along with **mouse movement and hover activity** on the page. Also, you can go directly to **click moments which are marked on the play bar**. This is a very useful capability to understand how users perceive your web page and whether they struggle with some menus buttons or operations.

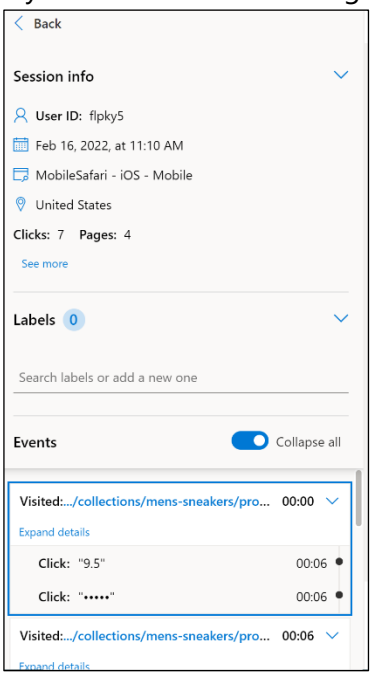






5. You can also click “More Details” to view more details about the recording. It will show a timeline view of all the user clicks on the page and allows you to label the recordings, in case you want to find it later.



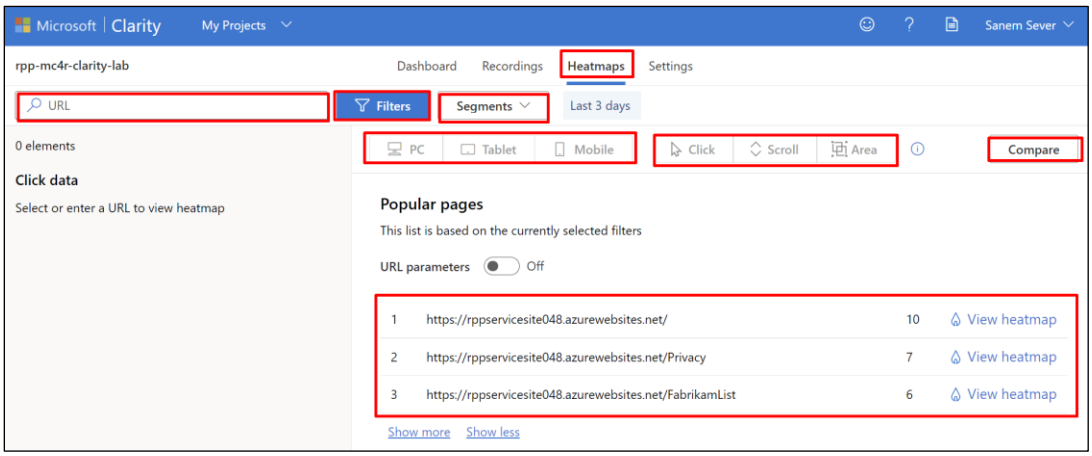


Note: Clarity provides a **Masking** option, which enables you to mask out certain elements / data on the page that you do not want to collect from them. Please see Task 4 for more info.

Task 4: Discover Clarity Heatmap capabilities

Clarity Heatmaps capability lets you observe **where** in your website **users engage the most or the least:** where they **click, tap, or scroll**. You can analyze this information by **device type** like **PC**, **Mobile**, **Tablet** and you can look at the details by **Segment filtering** and/or **custom filters**.

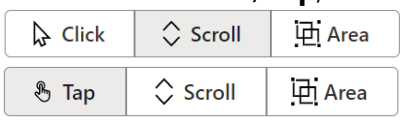
You can even **compare** two different **heatmaps**, for example to understand differences in user behavior after a change in the website UI.

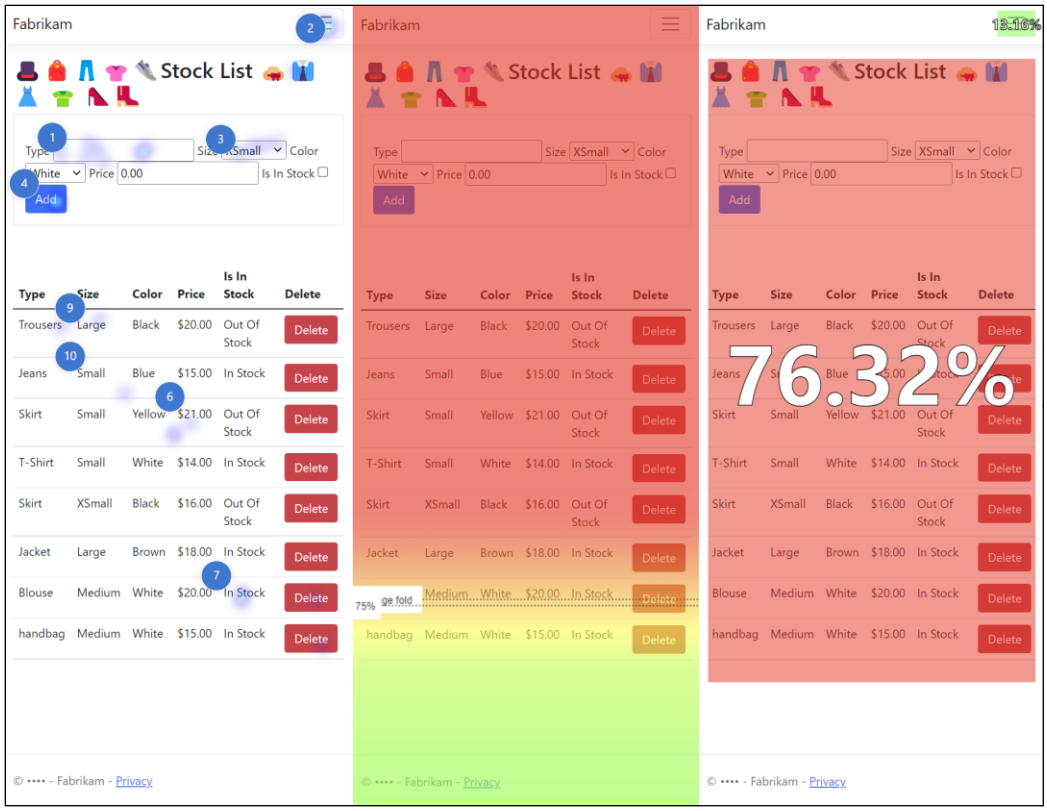


1. Go to https://clarity.microsoft.com/ and sign in if required.
2. Click on your project
3. Click on the Heatmap view and observe the contents in the view.
4. Switch between **PC**, **Tablet** and **Mobile** options to see the difference

5. Switch between **Click/Tap**, **Scroll** and **Area** options to see the difference

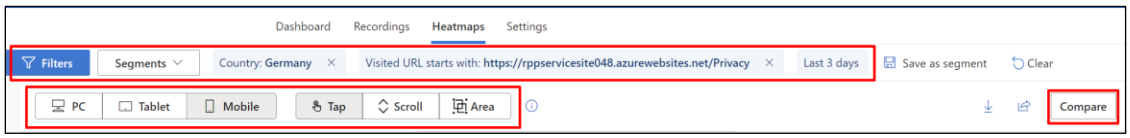
   



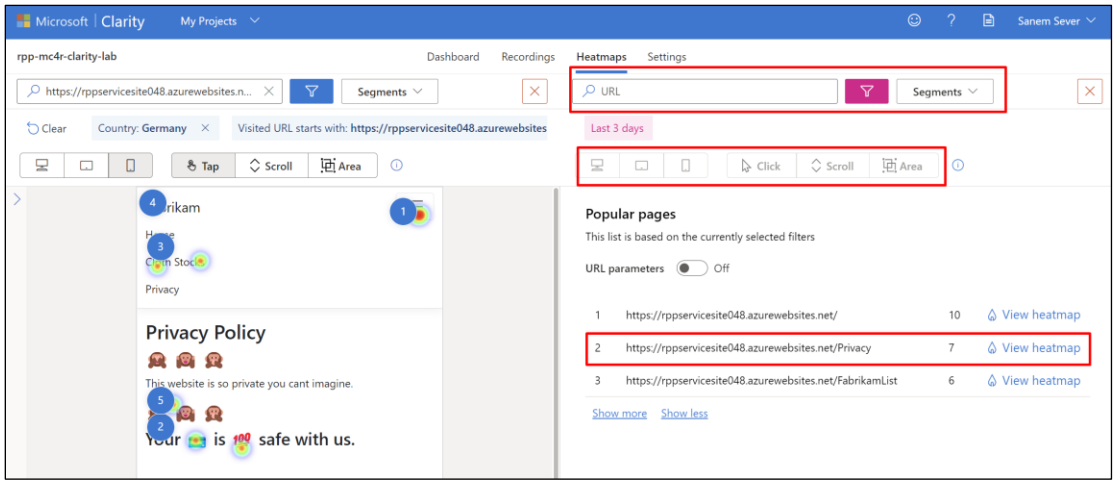
Click/Tap Map Scroll Map Area Map

- Click and tap maps help you see where users are clicking the most, or not at all.
- Scroll maps show you where users are scrolling to -what percent of users do and do not see parts of your site pages.
- Area maps are aggregate views of click/tap maps -they show you the click rate for a group of elements, instead of just one.

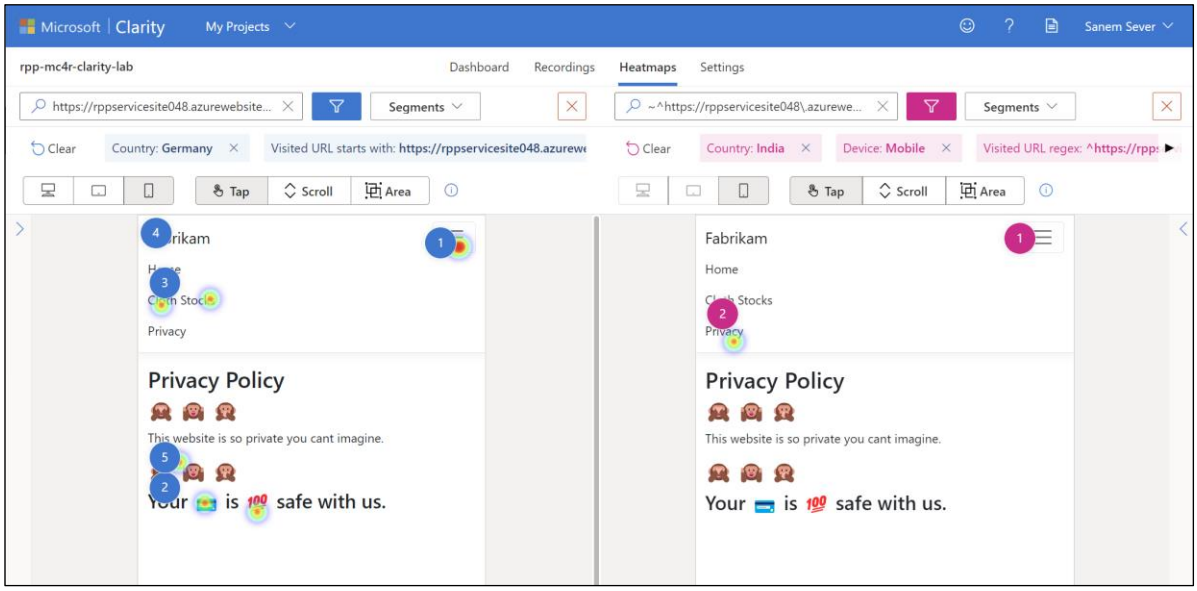
6. To **compare** two different **heatmap** for a **different** filtering click on **Compare**



7. You can target a specific **Segment**, **filter**, **device**, **URL** or **behavior** as your choice.



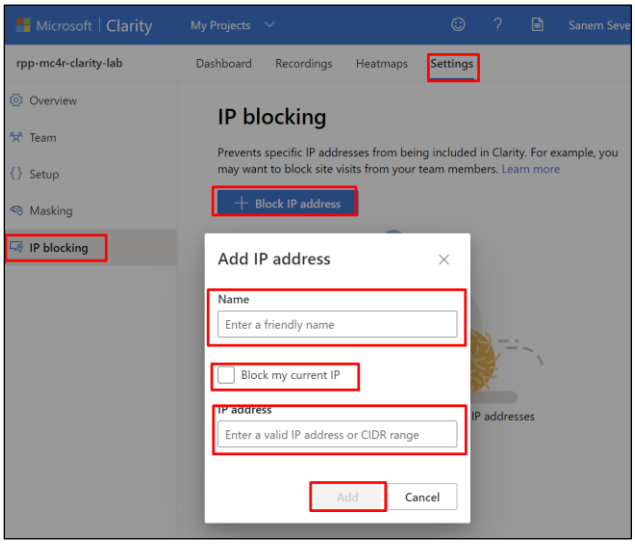
8. You can see an example comparison below:



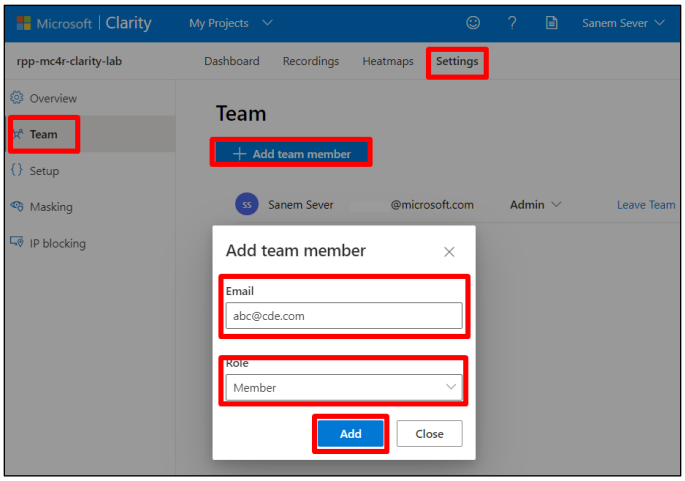
Task 5: Discover other settings for Clarity

Clarity also provides you with additional controls like **IP blocking** and **Team** creation.

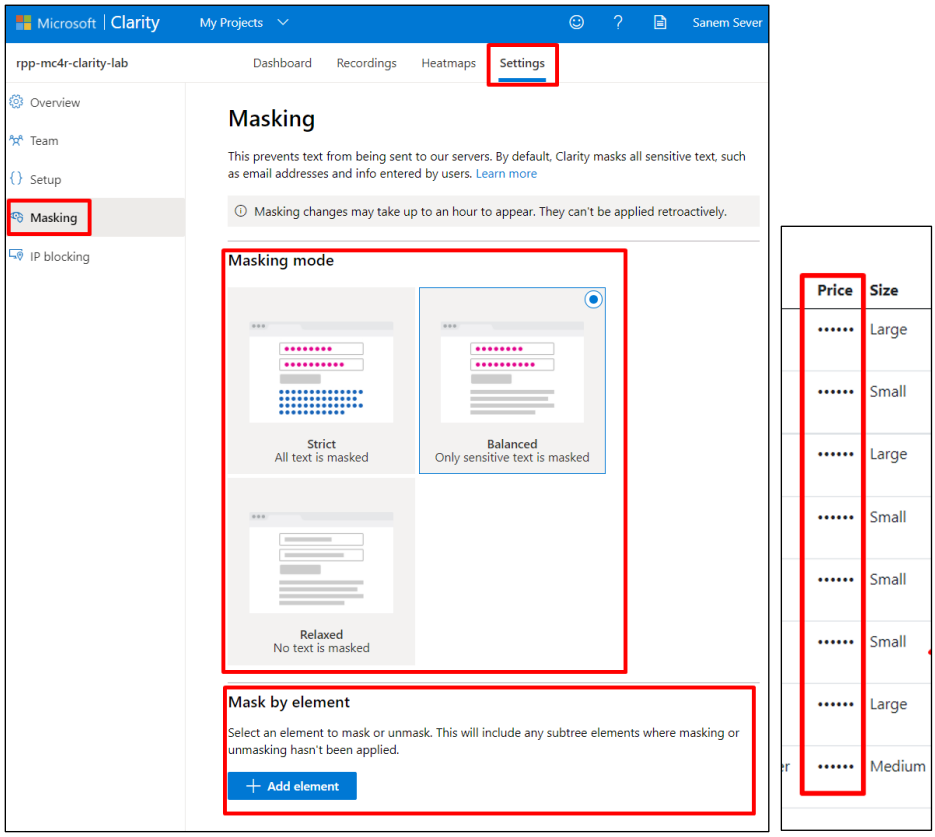
1. If you want to **eliminate specific IP's** usage from getting captured, you can use **IP blocking** feature. To do that:
    a. From **Settings** go to **IP blocking** and click on **+ Block IP address.**
    b. Fill a descriptive **Name** for the **IP** you are blocking and click **Apply**.



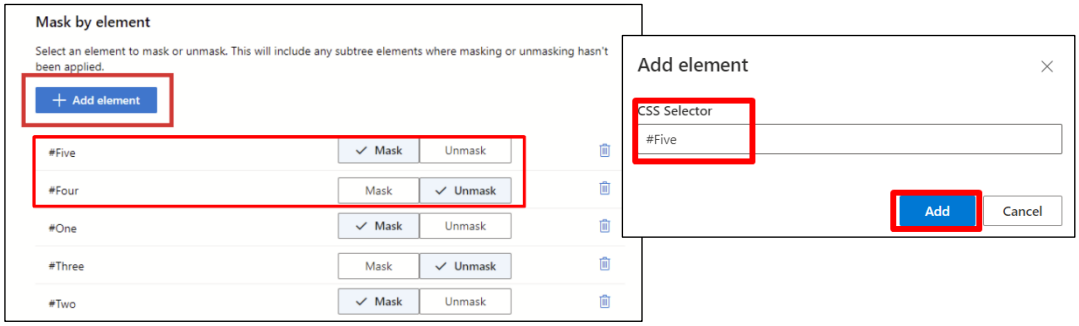
2. If you want a team of people to monitor and manage Clarity you can use **Team** management feature.
    a. From **Settings** go to **Team** and click on **+ Add team member.**
    b. Fill the **Email** for the **User** you are adding and select a role, options are **Member/Admin** and click **Apply**.



3. You may want some information to **masked** in your **Recordings**. Clarity masks the information it perceives as confidential, **by default** with **Balanced** masking. You can change the masking mode by selecting from three provided options, **Strict**, **Balanced**, **Relax**. *Changes do not get reflected to previous recordings and can take an hour the changes to appear.*



*Info Note: You can also mask using specific **CSS elements** from your web site code. From **Mask by element** you can see above, you can enter the **CSS selector** for the element to be masked and click **Add**. For example, enter .class_name for a class, #id_value for an ID, and element for a type. From here you can adjust whether you want to mask or unmask these specific CSS elements.*



Congratulations! You completed the **Clarity** lab for **Microsoft Cloud for Retail**.

Summary

Nice work! You have completed **Lab 03 Microsoft Clarity**

In this lab, you learned how to do the following:

- Deploy a Website to Azure with Visual Studio Code
- Create a Microsoft Clarity Project
- Integrate a Microsoft Clarity Project with a custom website
- Integrate a Microsoft Clarity Project with a Dynamics 365 E-Commerce website
- Monitor Website usage with Microsoft Clarity capabilities.

Completing this lab concludes the Shopper and Operations Analytics scenario part of the Microsoft Cloud for Retail In a Day training.