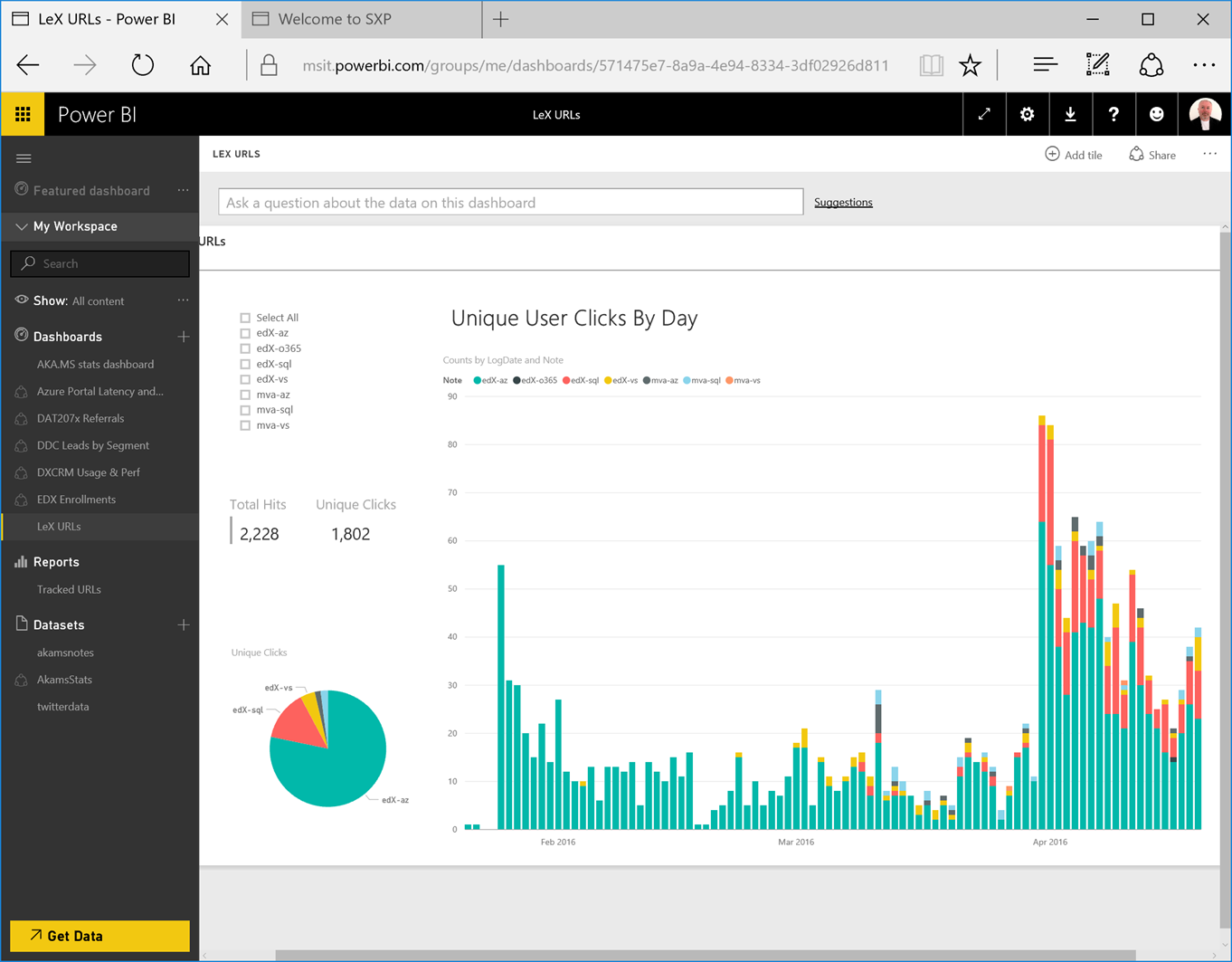
Creating Trackable Short URLs

As a group, one of our required contributions to Microsoft as a whole is to drive affinity for Microsoft products and services – particularly cloud services. One way we can do that is to refer students directly to URLs from where they can *install* or *subscribe* to Microsoft products/services.

There are two levels at which URL referrals can be tracked – there’s a Microsoft-wide standard for tracking web activity through a Web Tracking Marketing Campaign ID (*wt.mc\_id*) URL parameter, which enables marketing teams to drill into clickstream details, and there’s a simpler *aka.ms* click-tracking system that records total and unique-user clicks. Our approach is to combine these two techniques in a simple to implement solution that enables easy high-level reporting of clicks through Power BI, while still retaining the ability to do deep analysis of clickstreams through the *wt.mc\_id* parameter.

Here’s an example of the kind of report that can be seen in Power BI:



With the full Microsoft marketing web tracking system, more specific details can be tracked – for example, how many references to Azure resulted in paid subscriptions, or how many references to download Visual Studio resulted in actual product installations.

For this to work successfully, we need to ensure that all URLs from online content (MVA, edX, etc.) use the consistent approach described in this document. Note that the same approach could also be adopted by MOC, MOC-On-Demand, MS Press etc to create a consistent, organization-wide solution.

# Creating a Tracked URL

Use this process **only to create tracked URLs where students install Microsoft products or subscribe to Microsoft services** – you can create aka.ms links to other targets, such as lab files, TechNet/MSDN articles, etc, if you wish – but please **don’t** use the wt.mc\_id metadata described below!

## Determine the Target URL

1. Identify the target URL from which the software is installed or the service is subscribed.
2. Construct a suitable **wt.mc\_id** parameter value. This should be in the following format:

DXLEX\_*channel*\_*coursenum*\_*optionaltag*

For *channel*, use the following values:

|  |  |
| --- | --- |
| **Where the course is published** | **Channel** |
| Microsoft Virtual Academy (MVA) | MVA |
| edX.org | EDX |
| Microsoft Official Curriculum (MOC) | MOC |
| Microsoft Press (MS Press) | MSP |
| MOC-On-Demand | MOD |
| Other | *Devise a suitable code and add it to this document (or email* ***gmalc****)* |

For example, course DAT202.1x in edX.org should have the following **wt.mc\_id** parameter:

DXLEX\_EDX\_DAT202.1X

The optional tag is used when you want to track an additional piece of custom information (for example, to track clicks from a specific page within your course separately from other pages in the same course.) You can omit it if it’s not required (which it usually isn’t).

Course 12345 in Microsoft Virtual Academy with an optional tag of “setup” should have the following **wt.mc\_id** parameter:

DXLEX\_MVA\_12345\_SETUP

Note that your *wt\_mc\_id* tag should ***always*** start with DXLEX, so we can identify links from LeX content.

1. Add the **wt.mc\_id** parameter to the URL and verify that it still opens the expected page or starts the expected download. For example, if the target URL is <https://azure.microsoft.com/en-us/pricing/free-trial/> then the full destination URL could be <https://azure.microsoft.com/en-us/pricing/free-trial/?wt.mc_id=DXLEX_EDX_DAT201.1X> (note that you may need to add the trailing “/” to the target URL to make the link work.

If the target URL already includes parameters, append the **wt.mc\_id** parameter. For example, for the target URL <http://go.microsoft.com/fwlink/?LinkID=691979>, the full destination URL could be <http://go.microsoft.com/fwlink/?LinkID=691979&wt.mc_id=DXLEX_MVA_12345_SETUP>

## Determine the Vanity URL

1. The vanity URL is a short friendly URL based on the following format:

*channel*-*coursenum*-target

For *channel*, use the same codes as for the **wt.mc\_id** parameter but in lower case. For *target*, use the following values:

|  |  |
| --- | --- |
| **Product or Service** | **Target** |
| Azure | az |
| Visual Studio | vs |
| Office 365 | o365 |
| Power BI | pbi |
| Azure Machine Learning | aml |
| SQL Server | sql |
| Windows Server | ws |
| Windows client | win |
| Excel | xls |
| Microsoft Sway | sw |
| SharePoint Server | sp |
| Other | *Devise a suitable code and add it to this document (or email* ***gmalc****)* |

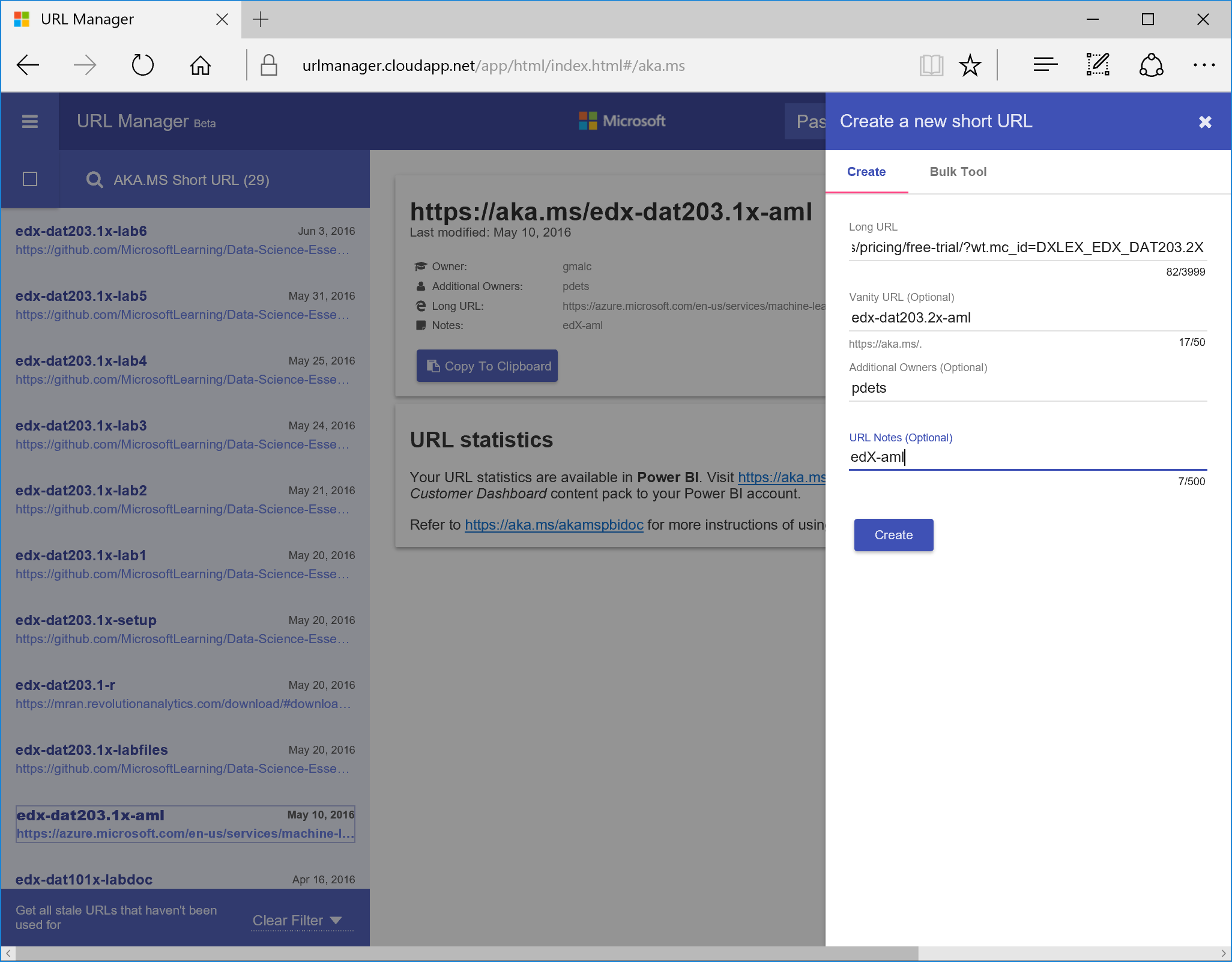
For example, a vanity URL for an Azure trial subscription link in edX course DAT202.1x should look like this:

*edx-dat202.1x-az*

## Create the short link

1. To start, open the URL Manager tool at <https://aka.ms>.
2. Create a new short URL with the following properties:
   * **Long URL**: The full destination URL (i.e. the target URL with the **wt.mc\_id** parameter). For example, <https://azure.microsoft.com/en-us/pricing/free-trial/?wt.mc_id=DXLEX_EDX_DAT201.1X>.
   * **Vanity URL**: http://aka.ms/*vanity\_url*. For example, <http://aka.ms/edx-dat202.1x-az>.
   * **Additional Owners**: pdets (this is important so that the PDETS team can manage all LeX URLs)
   * **URL Notes**: *channel*-*target*. For example, **edx-az** or **mva-vs**. Use a minus (“-“) character to delimit the channel and target fields, and do not include any spaces or additional text.

In the URL Manager tool this should look like this:

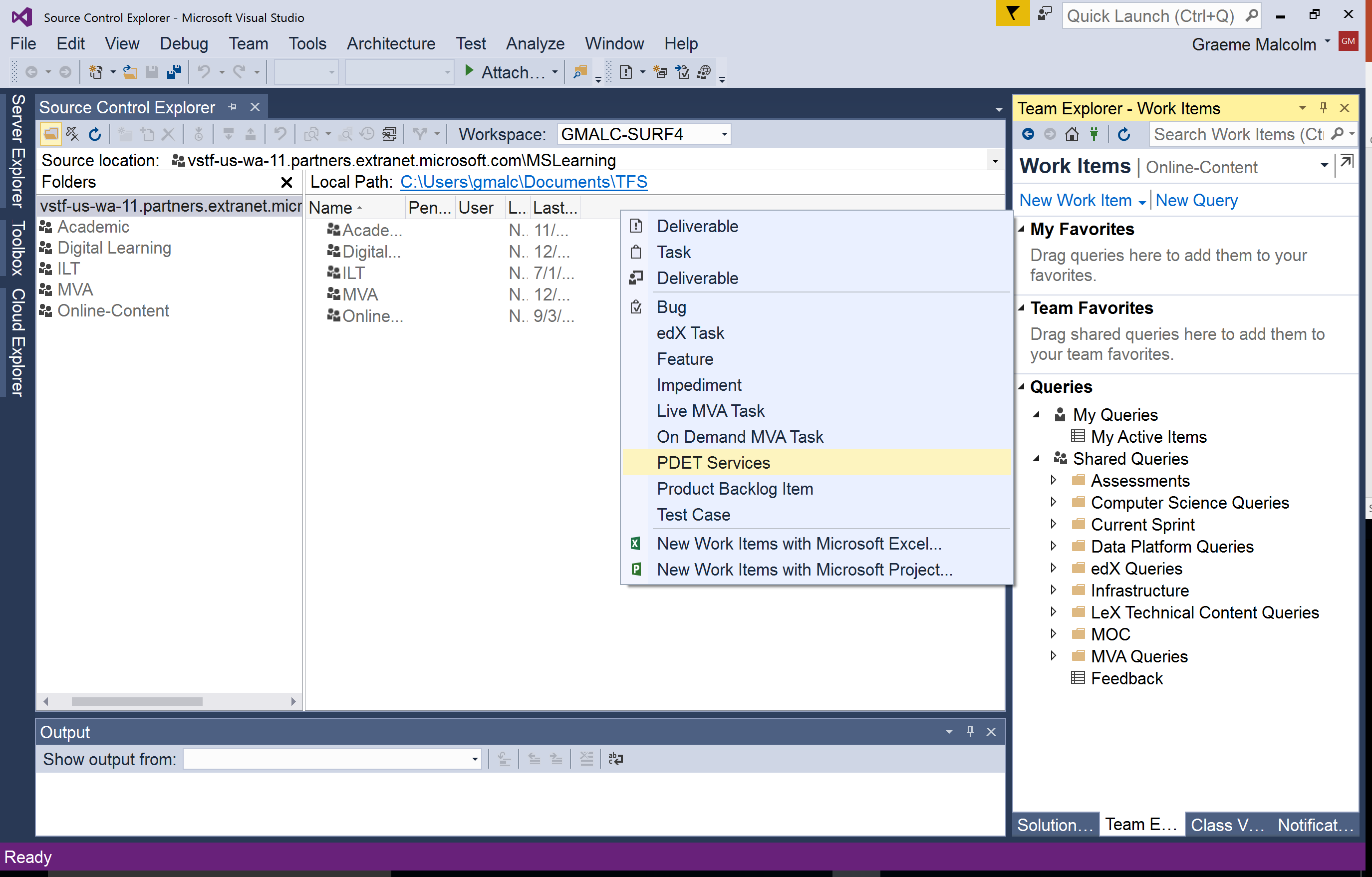


1. Submit the short URL request, and then test the short link that is generated to verify that it opens the expected page or starts the expected download.
2. Use the short URL in your course materials.
3. Don’t forget to add short URL to the **Referral URL Tracking** area of the deliverable in TFS that corresponds to the course number.

# Requesting URLs via PDETS

If you don’t fancy doing all the work yourself, you can request that the LeX Product Dev Engineering and Tools Services (PDETS) team do it for you.

Just create a new PDET Services task in TFS:



And then create a task assigned to *LeX Product Dev Engineering and Tools Services* specifying the following details:

* **Target URL** (without a wt.mc\_id parameter – PDETS will create this for you). You can choose this from a pre-populated list of commonly used URLs, or choose None of the Above and enter a custom URL.
* **Channel**. The delivery channel in which the URL will be published (for example edX or MVA) – note that if you need to link to the same service from multiple channels, you will need to create a separate URL for each.
* **Product/Service**: The Microsoft product or service that the URL links to (for example Azure)
* **Tag** (optional): A custom tag to be added to the URL as described above.
* **Notes**: Any additional notes you want to give to the PDETS team – for example an instruction to add the URL to a Resources page in a course they are assembling for you.

Then submit the task and wait for the URLs to be generated – they’ll appear in the **Referral URL Tracking** area of the deliverable in TFS that corresponds to the course number.