Experiment - Provisioning an Azure storage account using the Azure portal

In this experiment, we'll provision an Azure storage account using the Azure portal. Azure Blob storage is one of the four storage services available in Azure Storage. The other storage services are **Table**, **Queue**, and **file share**.

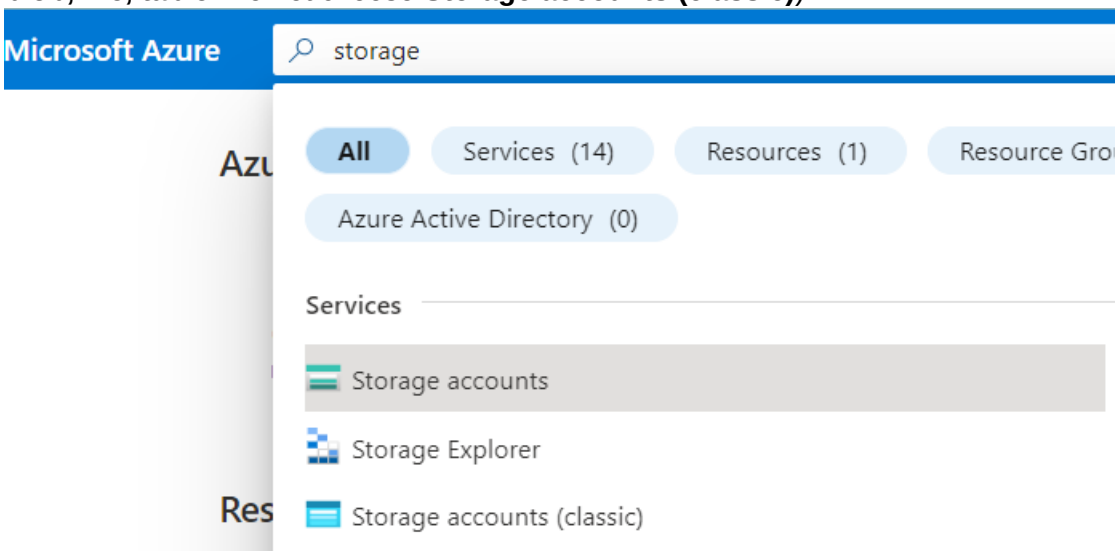
Getting read

Before you start, open a web browser and go to the Azure portal at **https://portal. azure.com**.

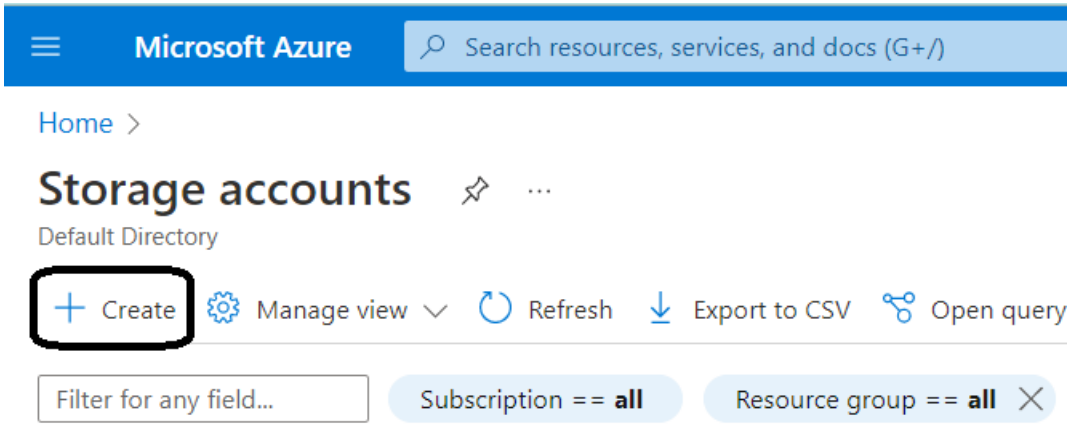
How to do it…

The steps for this experiment are as follows:

1. In the Azure portal service search dialog enter storage and choose **Storage account – blob, file, table.** Do not choose **Storage accounts (classic)**).

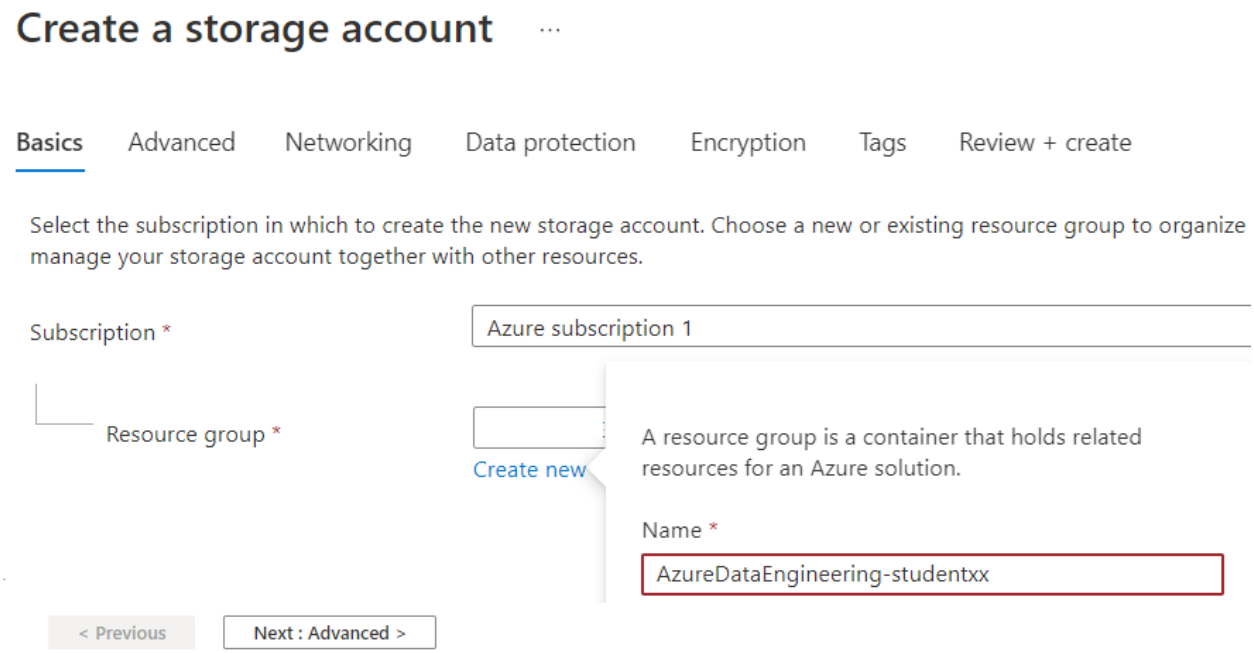


2. Choose Create

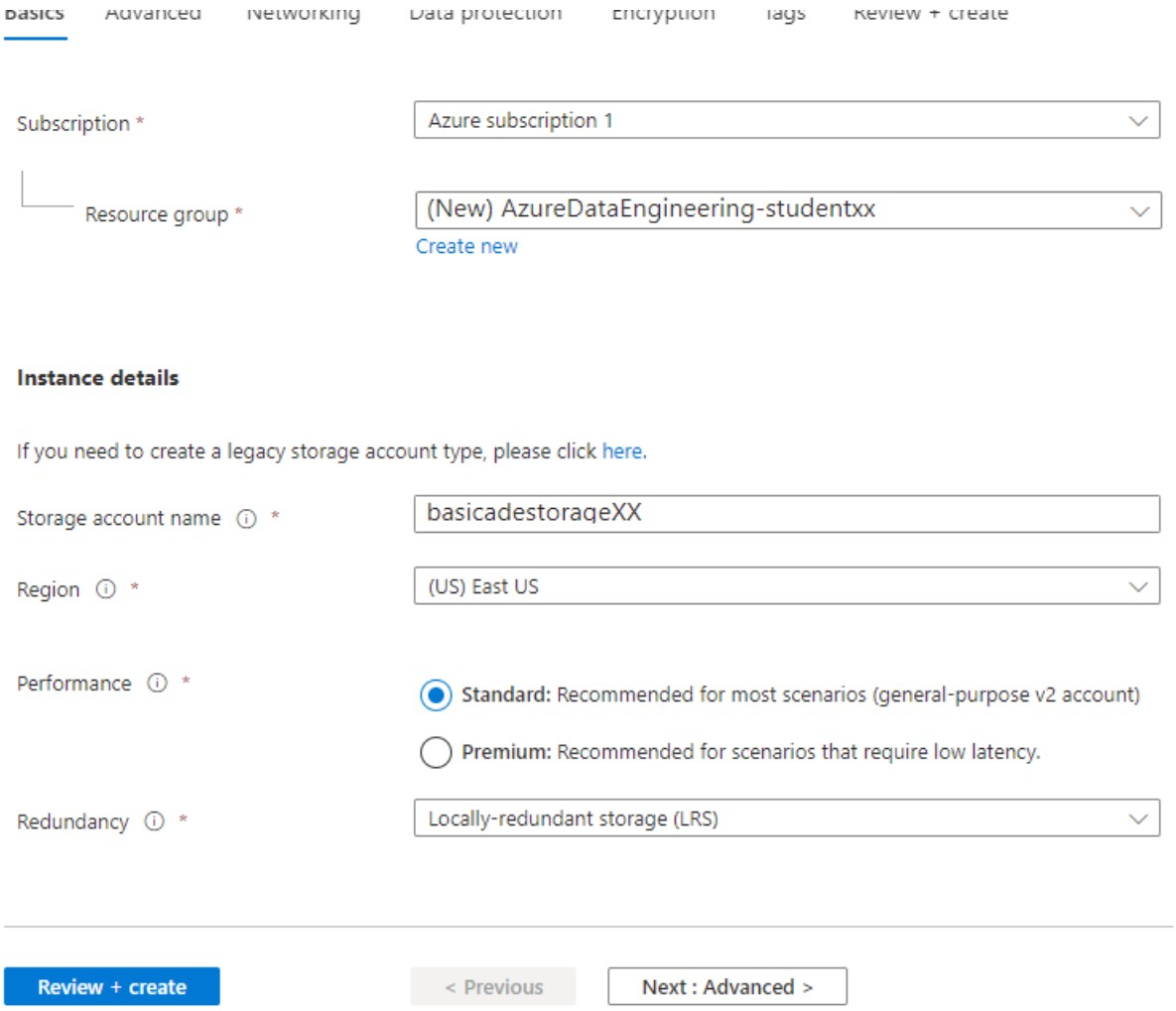


3. A new page, **Create storage account**, will open. There are seven tabs on the **Create storage account** page – **Basics**, **Advanced**, **Networking**, **Data Protection**, **Encryption**, **Tags**, and **Review + create**.

4. In the **Basics** tab, we need to provide the **Azure Subscription (defaults to our subscription)**, then on the **Resource group** select **Create new** and enter **AzureDataEngineeringXX, replacing the XX with your student number (student1 would be AzureDataEngineering01, student12 would be AzureDataEngineering12)** and select **OK**



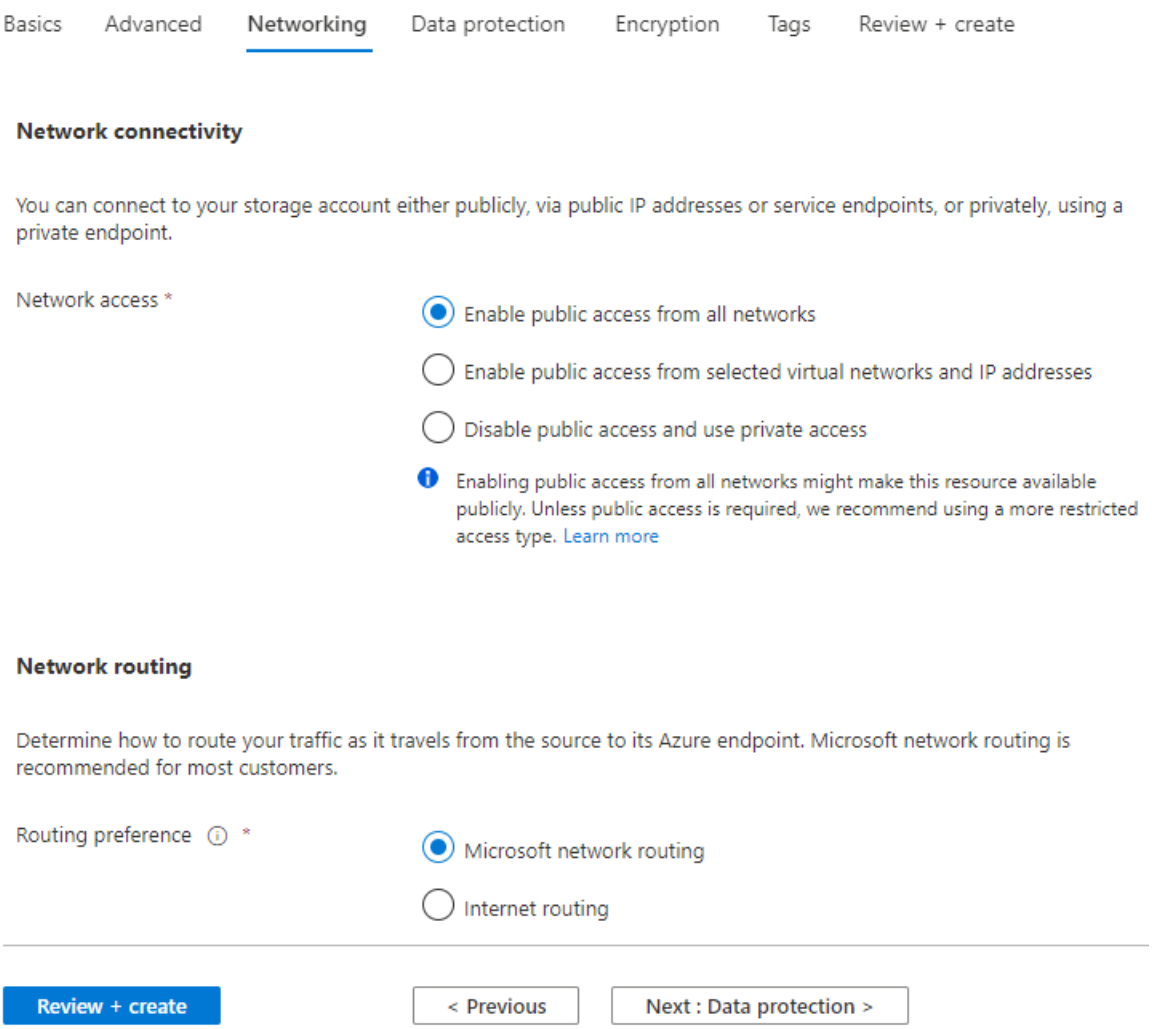
5. We'll complete the remaining values for the **Storage account name**, **Region**, **Performance**, and **Redundancy** values, as shown in the following screenshot, then select **Next: Advanced >. For the Storage account name ensure that you append your student number, basicadestorageXX, replacing the XX with your student number (student1 would be basicadestorage01, student12 would be basicadestorage12)**



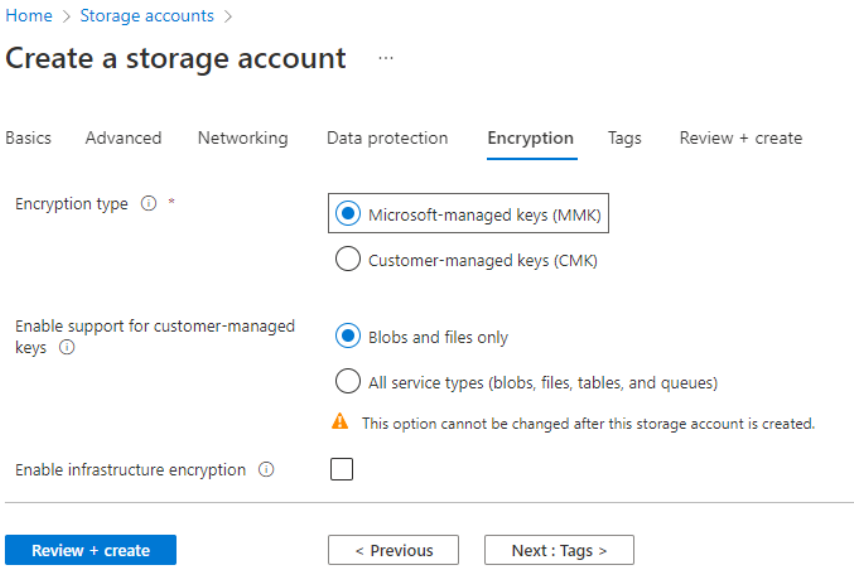
6. Leave the defaults for Advanced related to **Security, Blob Storage, Azure Files and Data Lake Storage Gen V2**. Select **Next: Networking >**.
7. The default is **Enable public access from all networks**, this is not following the principal of least privilege and should be used with caution in real world scenarios, change to :

Figure 1.3 – Create storage account – Advanced

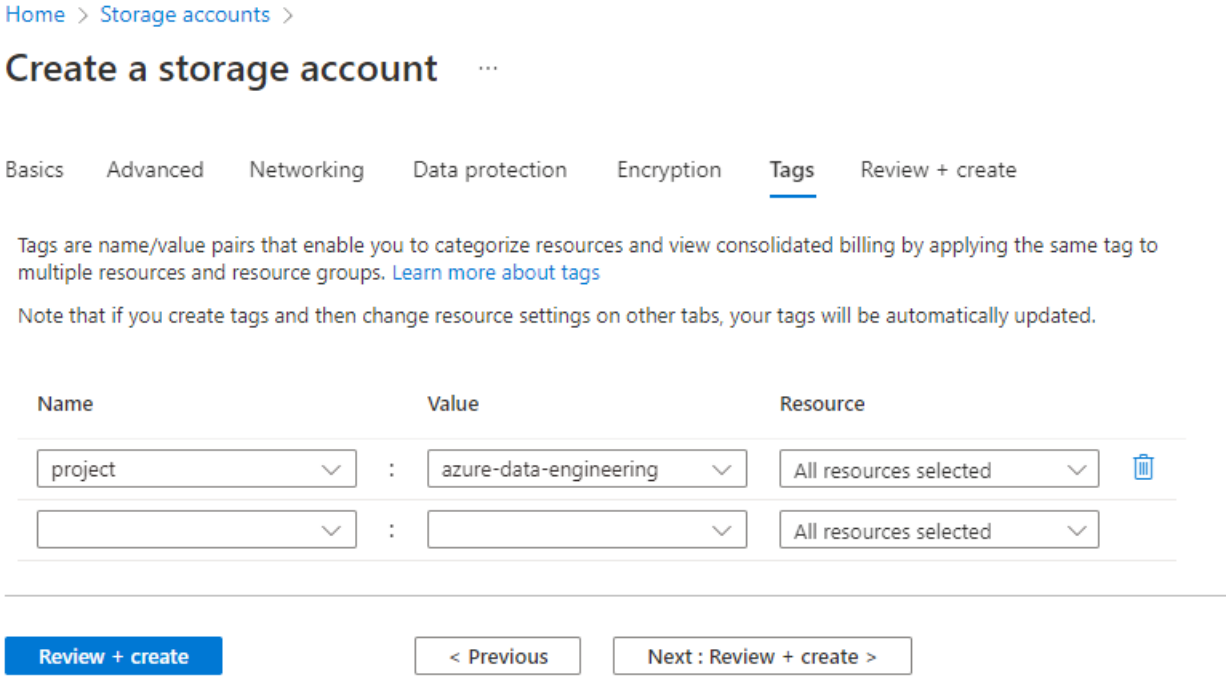
8. Select **Next: Data protection >**.



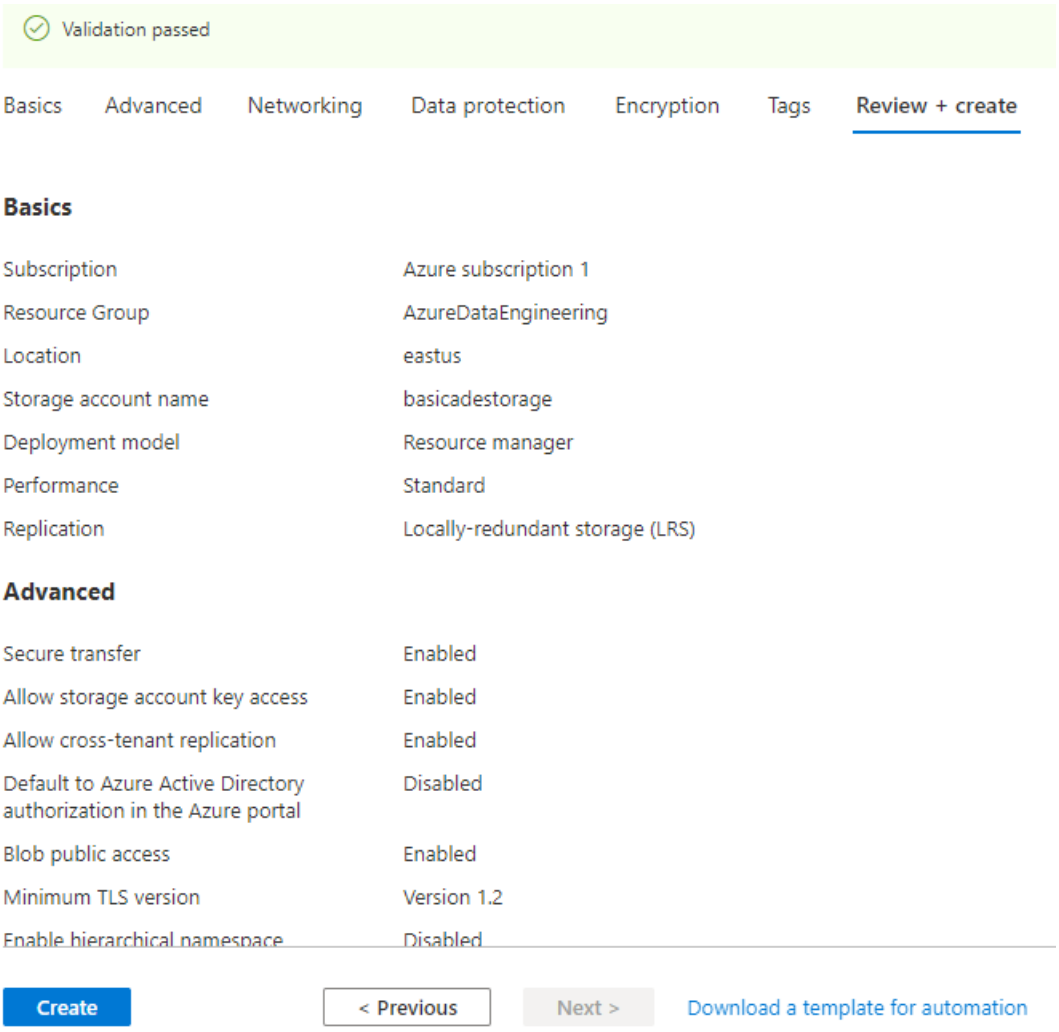
9. Leave the defaults for Recovery, Tracking and Access Control. **Select Next: Encryption >**.



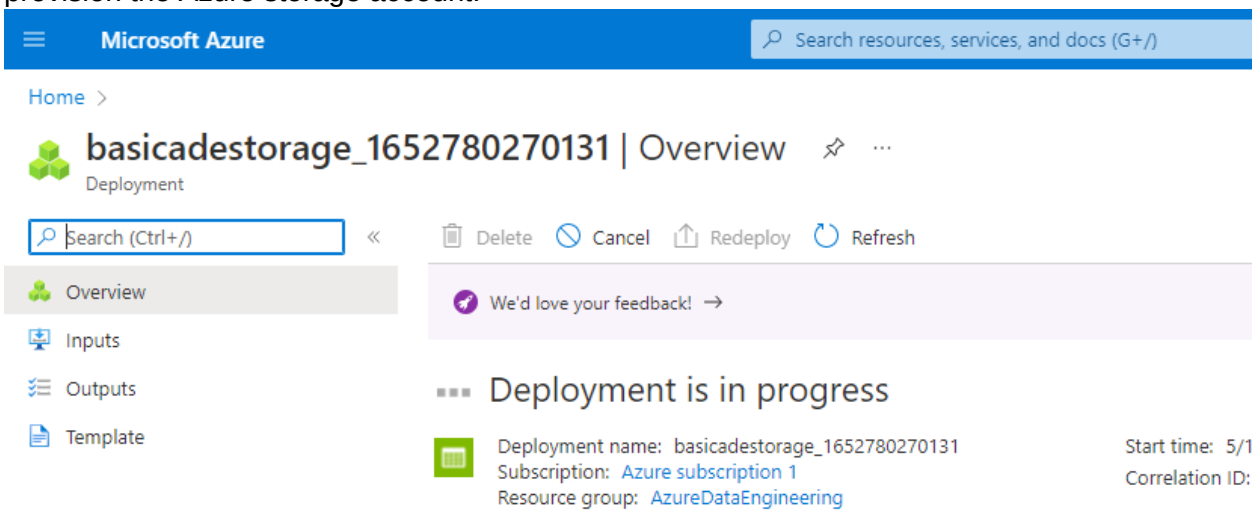
10. Leave the default to use Microsoft-managed keys (MMK). Select **Next: Tags >**



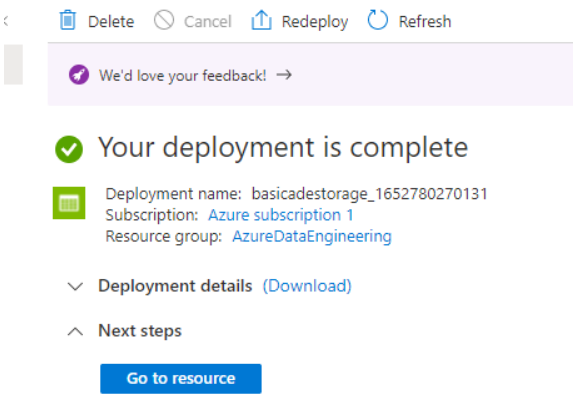
11. It is a best practice to tag all resources for show back, charge back or bill back in real world scenarios. Enter **project** for Name and **azure-data-engineering** for Value of our tagging for this resource. Choose **Next: Review + create >**



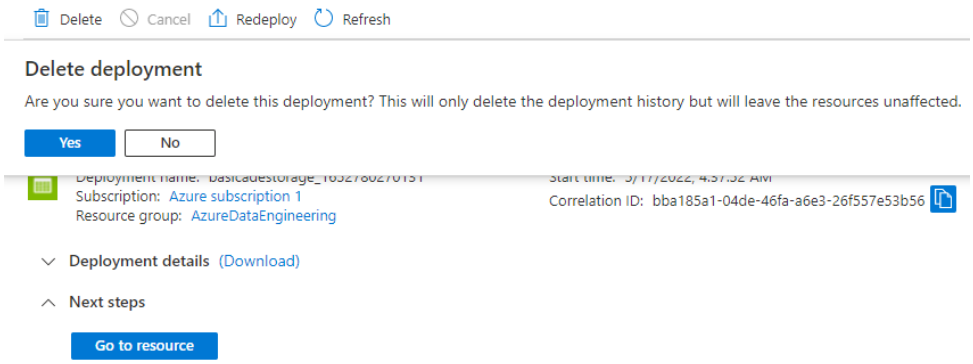
12. In the **Review + create** tab, review the configuration settings and select **Create** to provision the Azure storage account.



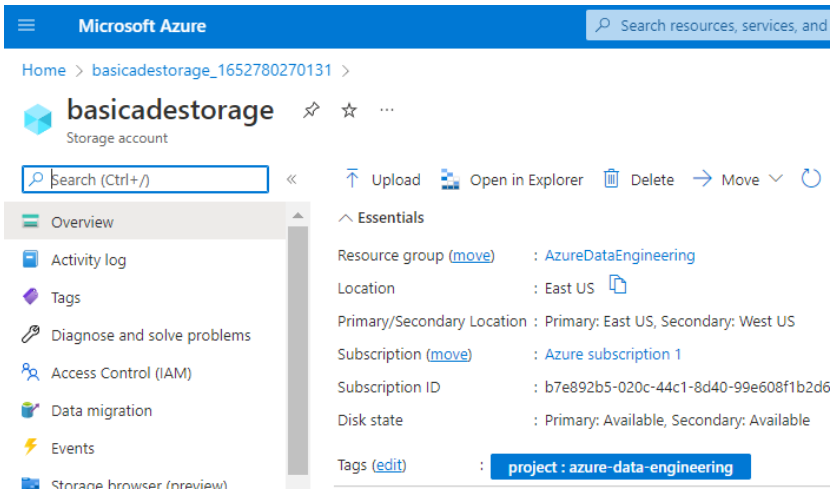
13. We'll see that the Deployment is in progress and should complete shortly.



14. Congratulations, you're created an Azure Storage Account. Like all good stewards in cloud usage and following Cloud Financial Management best practices, since this was an experimental resource we'll now remove it.
15. Choose **Delete**



16. Note that what we just decided to do was delete the deployment of our resource but not the actual resource, choose **No**.
17. Select **Go to resource**



18. Now choose **Delete.** Type to the name of the storage account to confirm and select **Delete**.

19. Thank you for being good stewards. Browse to the Storage Accounts blade home. Note that we'll still see our **basicadestorageXX** but it will be 404 Not Found. In the list of options for the accounts, under **Support + troubleshooting**, we can see **Recover delete account**. This would allow us to recover for a period of time configured in the Recovery section of the creation for our Azure Storage account, by default 7 days.
20. Congratulations on this exploration of the storage account service. Experiment complete.

How it works…

The Azure storage account is deployed in the selected subscription, resource group, and location. The **Performance** tier can be either **Standard** or **Premium**. A **Standard** performance tier is a low-cost magnetic drive-backed storage. It's suitable for applications such as static websites and bulk storing flat files. The **Premium** tier is a high-cost SSD-backed storage service. The **Premium** tier can only be used with Azure virtual machine disks for I/O-intensive applications.

Account kind is of two types, **general purpose** (StorageV1 and StorageV2) and **Blob storage**. General purpose provides one account to manage multiple Azure storage services, such as blobs, files, disks, queues, and tables. The Blob storage account only supports block blob and append blob storage. It doesn't support blobs, tables, or queues. There are six replication types available for different high availability and durability requirements. **Locally redundant storage** (**LRS**) is the cheapest and minimal durability option. The data is replicated three times within the same data center in the primary region.

Azure storage accounts can be accessed publicly over the internet, publicly through selected networks (selected IPs, IP ranges), and from private endpoints. Encryption can be using Microsoft-managed keys (MMK) or Customer-managed keys (CMK). MMK is a managed service, and CMK requires you to have a key management solution providing the keys to use in encryption of your storage account files.

Tagging is a best practice, and all resources should be tagged when using the portal, CLI, or for Infrastructure as Code solutions, e.g. Terraform.

Standards should be used in naming so that tags, resource groups, storage account naming and any field that allows manual text entry follows the organizational guidance for those entries. Failure to adhere to naming has down stream impacts in sustainability and maintainability.