Apache spark

Apache Spark is a lightning fast, unified analytics engine for big data processing and machine learning.

Spark was built from the ground up to address the shortcomings of Hadoop.

Hadoop was slow and inefficient for interactive and iterative computing jobs, and it was too complex

to learn and develop. On the other hand, Spark offers a much simpler, faster and easier APIs to develop on.

Spark can be 100 X faster than Hadoop, for large scale data processing by exploiting In-memory computing and other optimizations.

* Similar to most other big data engines, Spark runs on a distributed computing platform.
* Spark has an unified engine to support varying workloads. For example, it uses a single engine for streaming and batch workloads.
* It doesn't have separate one for each of those.
* It comes packaged with high level libraries, including support for SQL queries, streaming data, machine learning and graph processing.
* These standard libraries increase developer productivity and can be seamlessly combined to create complex workflows

High Level Architecture of Spark

**Spark Core:** The Spark Core takes care of scheduling tasks, memory management, fault recovery, communication with storage systems, etc. It's also home to Sparks main programming Abstraction API called RDD or Resilient Distributed Datasets.

What’s RDD?

RDDs are a collection of items distributed across various compute nodes, in the cluster that can be processed in parallel. Spark Core provides the APIs to create and manipulate these RDD collections.

It was difficult to use for complex operations and it was difficult to, optimize for Spark and mainly

down to the developer to write the optimized code.

**Spark SQL Engine:** In order to optimize the workload,

Spark introduced the SQL engine. It includes the Catalyst Optimizer, which takes care of converting a computational query to a highly efficient execution plan and the Tungsten Project, which is responsible for memory management and CPU efficiency.

The higher level abstraction such as Spark SQL and the Dataset and the DataFrame APIs, make it easier

to develop applications and also benefit from the optimizations from the SQL engine. So the recommended approach to develop applications in Spark, is to use these higher level APIs rather than the RDD API.

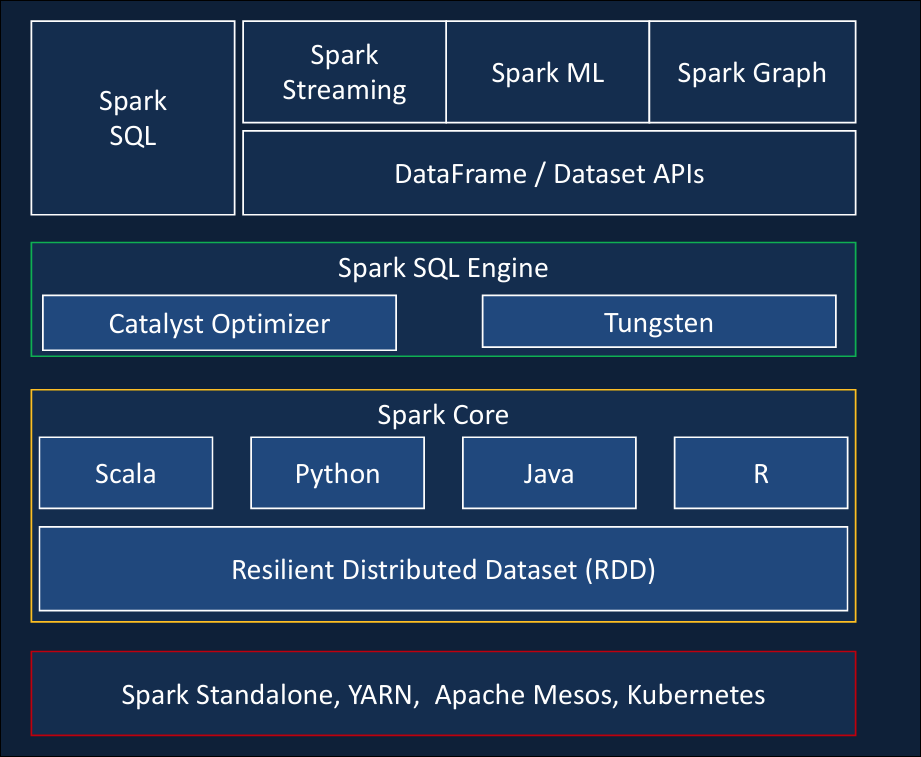
The Dataset and the DataFrame APIs can be invoked from any of the domain specific languages such as

Scala, Python, Java, or R. On top of this, we have the set of libraries such as Spark Structured

Streaming for streaming, ML Live for machine learning and also Graphics for graph processing.

Also, Spark comes with its standalone resource manager, but you can choose other resource managers

such as YARN, Apache Mesos and Kubernetes. Combining all of these, Spark provides the unified platform for doing streaming, batch, machine learning and graph processing workloads using a single execution engine and a standard set of APIs.



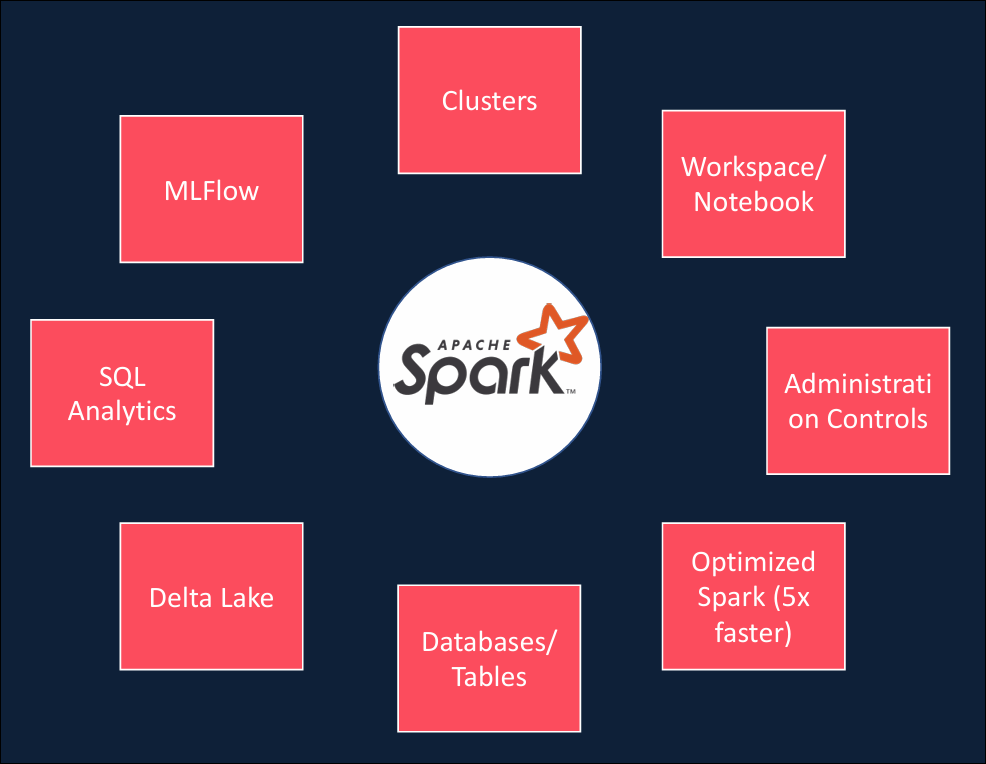
Now we know Spark is a fast execution engine with an easy to use set of higher level APIs. But, in order to work with Spark, we have to set up our own clusters, manage security, and also use third party products to write our programs. That's where Databricks comes in.

**Databricks**

Databricks is a cloud-based platform designed for big data processing, machine learning, and analytics. It provides an integrated environment for working with data using tools like Apache Spark, SQL, and other popular data science and machine learning libraries.

In simple terms, Databricks helps companies process large amounts of data quickly and efficiently, perform advanced analytics, and build machine learning models, all within a collaborative workspace. It's often used to scale and streamline data engineering and data science workflows in the cloud.

* In order for Spark to do its distributed computing, we need to spin up Clusters and install the software.
* It provides a Jupyter Notebook style IDE with additional capabilities to create your application. Collaborate with your other colleagues and also integrate with configuration management tools such as Git.
* It provides administration controls that you can use to restrict or provide access to your users, to the workspace, Clusters, etc...
* On top of this, Databricks provides the Spark runtime, which is highly optimized for the Databricks platform and known to be up to 5x faster than the Vannila Apache Spark.
* With the use of high metastore, Databricks also provides the ability to create databases and tables.
* In order to provide ACID transaction capability,Databricks also comes with the Open Source project Delta Lake,
* a recent addition to Databricks is the SQL Analytics, which provides the data analyst a SQL based analytics environment. This allows the analyst to explore data, create dashboards, schedule a regular refresh of the dashboard, etc..
* Also, it comes with managed ML flow on Databricks, which allows us to manage the machine learning lifecycle, including experimentation, deployment, model registry, etc..



But Azure's integration is deeper than others, Databricks is a first party service on Azure. What that means is, on Azure you will be buying Databricks directly from Microsoft and all support requests

are handled by Microsoft.

As a result, it provides a unified Azure Portal for Databricks and a single unified bill for all your Azure

services, including Databricks.

Azure Databricks leverages, Azure security and seamlessly integrates with Azure Active Directory

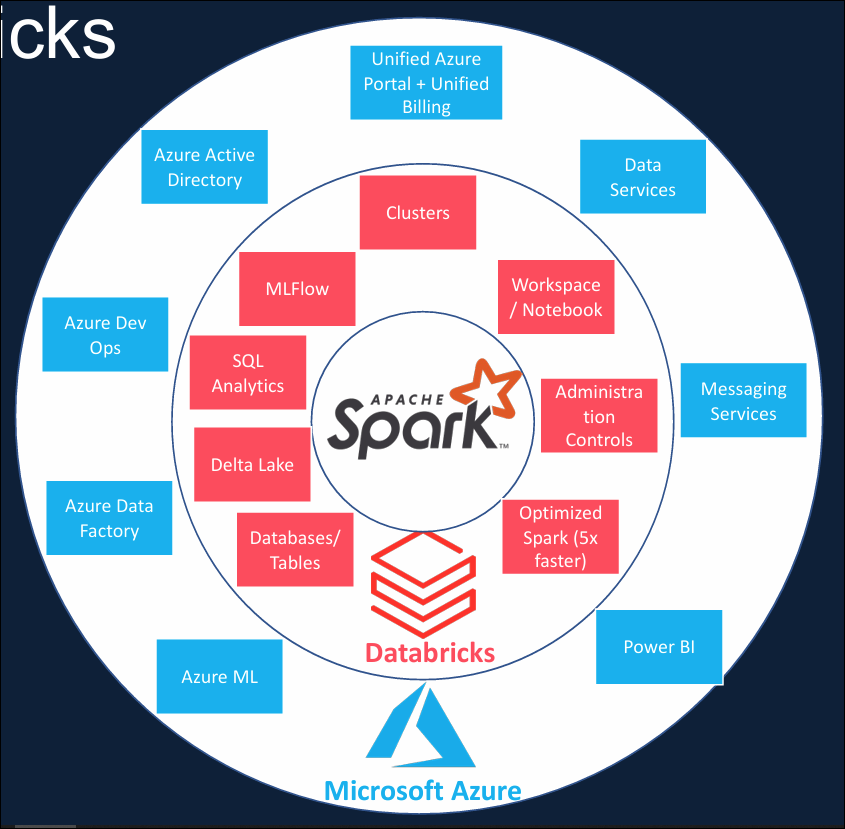
and single sign on.

It provides seamless integration and high-speed connectors between various Azure data services such as Azure Data Lake, Blob Storage, Cosmos DB, SQL DB and Synapse. Messaging services such as Event Hub and IoT Hub, Power BI and Azure ML, you can seamlessly run Databricks notebooks from Azure Data Factory and integrate with the rest of the data workflow in your data project.

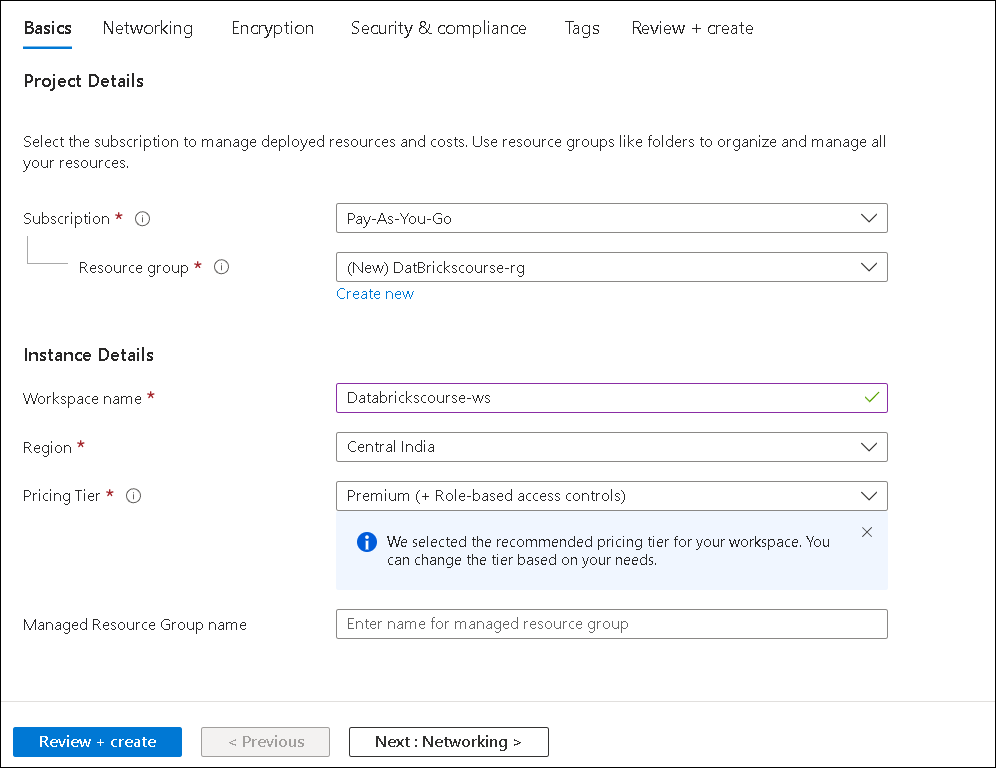
And finally, Databricks also connects with Azure Dev Ops to enable continuous integration and continuous

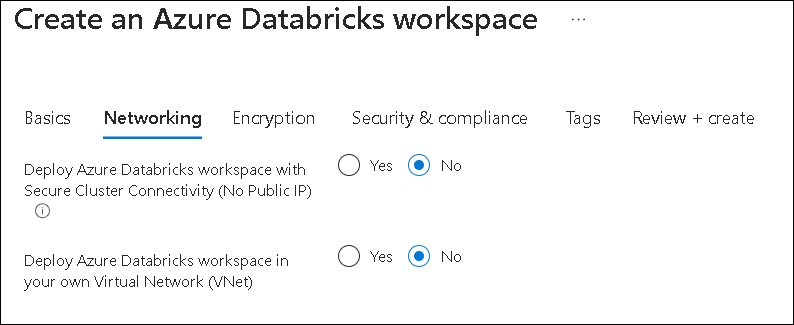
deployment.

So just to summarize, Azure Databricks is a spark based unified data analytics, platform as a service offering, that's optimized for the Microsoft Azure Cloud.

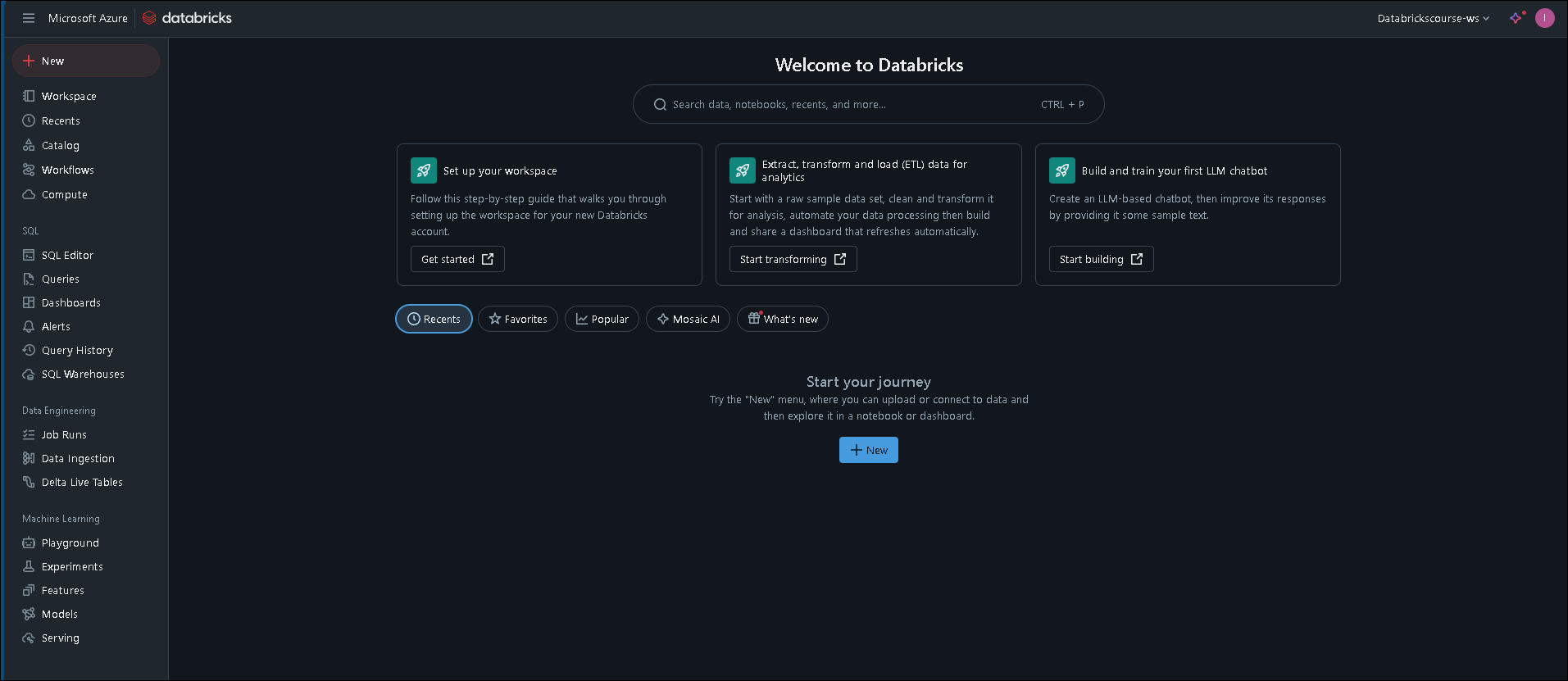


Create Data bricks workspace





There after just move to review and create pin it to dashboard and launch work space.



As we saw in the overview, Databricks offers multiple products to help with data warehousing, data engineering and machine learning.

This menu has also been designed to represent these product areas.

The section on the top of the navigation bar is the important one. It has the menu for accessing common functionalities across all three product areas we discussed.

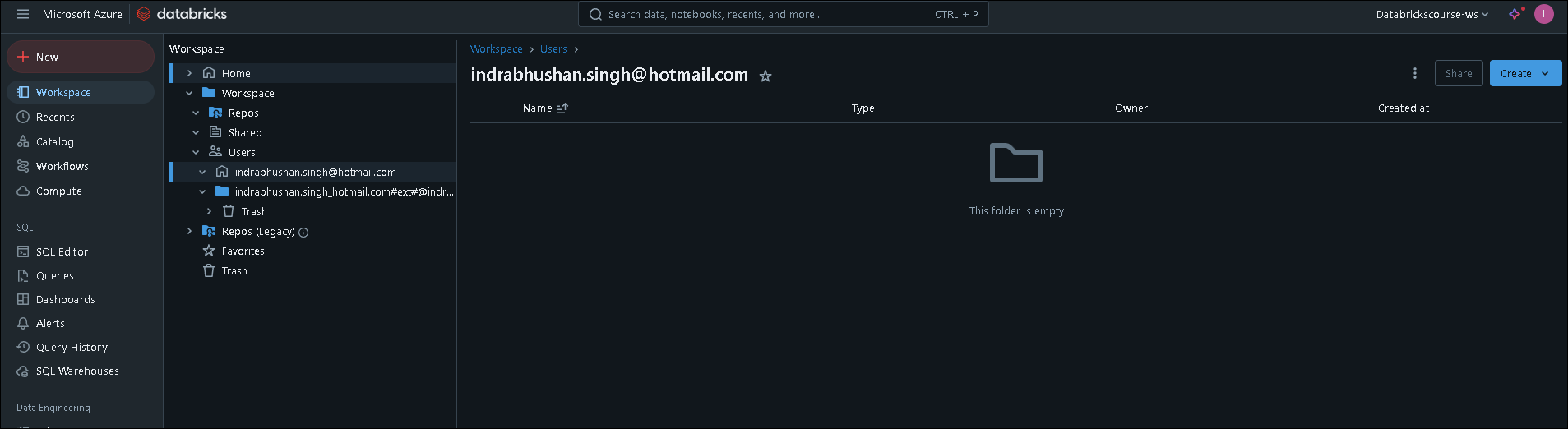
All the menu items you see under SQL relates specifically to Databricks SQL, which is Databricks' offering

to build SQL data warehouses on the Lakehouse platform.

The next section is data engineering, which includes specific products related to data engineering.

Similarly, the machine learning section includes products that are specific for machine learning workloads.

You can collapse these sections as you wish.



Workspace is basically a container for holding a set of folders, libraries, and files.

By default, each user has their own workspace and also there is a shared workspace which you can use to share assets amongst other users in this Databricks workspace. By right clicking on the workspace, you can create folders,

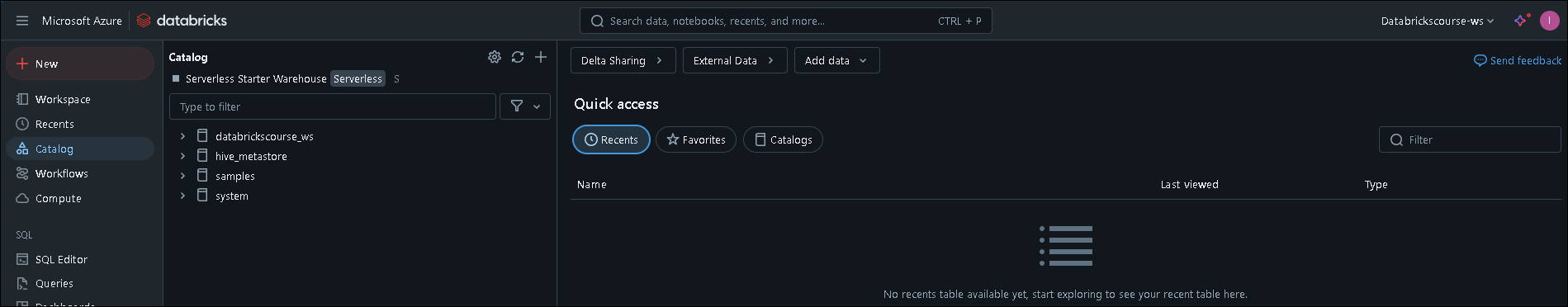
libraries, MLflow experiments, notebooks, et cetera. Also, you can import one of these type of assets from elsewhere. You also have the option to export your Databricks notebooks or folders into a Databricks file format called DBC or into source specific formats, for example, Python, SQL, Scala, et cetera.

Repos option here gives you a visual Git client within Databricks. Databricks lets you integrate this workspace with Git repositories offered by most of the Git providers, such as GitHub, Bitbucket, Azure DevOps Services, et cetera.

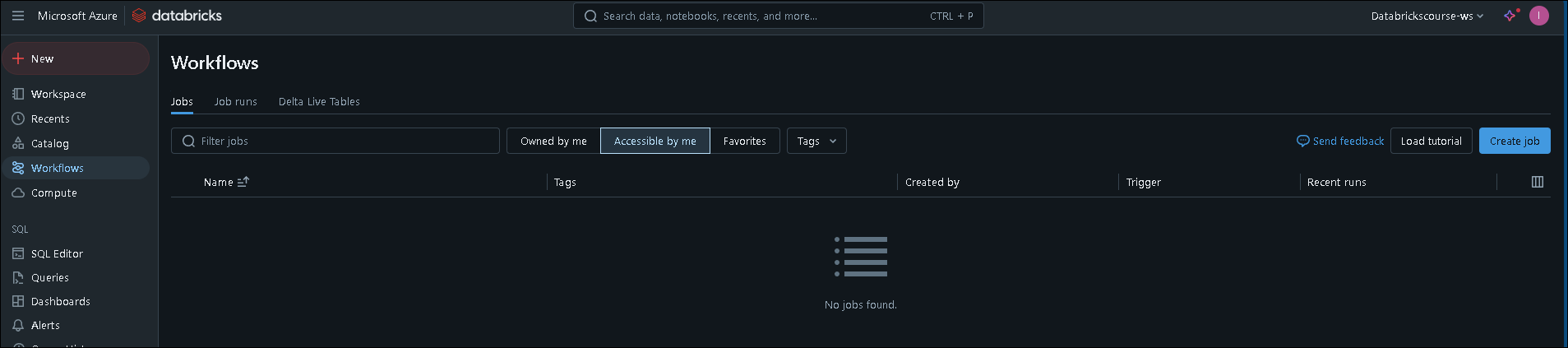
And it supports standard Git operations such as commit, push, pull, et cetera from a Git repository.

And finally deleted notebooks or files are kept in trash up to 30 days.

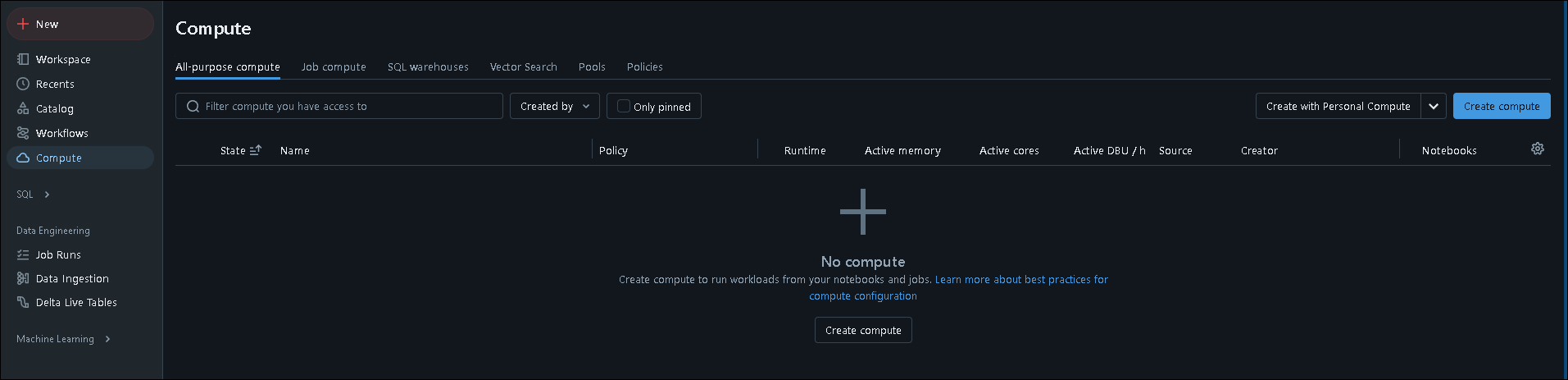
and any recently modified files will be listed under recents menu, so you can get quick and easy access to them.



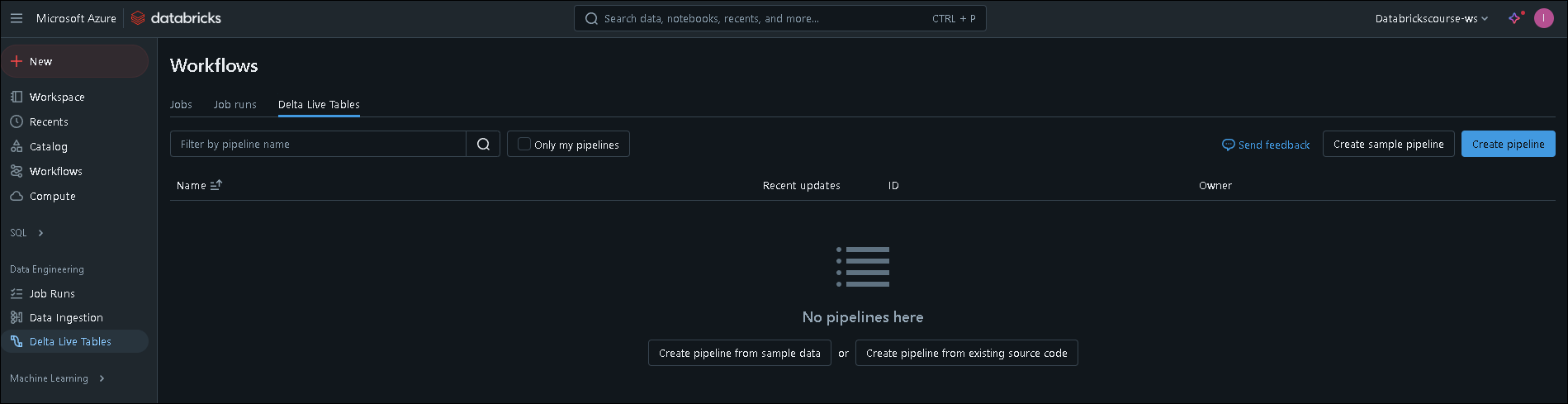
Catalog menu here lets us interact with any tables of use that you've already createdand also lets you create new tables in Databricks.Please note that this menu used to be called data,rather than catalog.



Workflows menu contains jobs, job runs and Delta Live Tables. Databricks jobs basically lets you schedule notebooks periodically via a scheduling system. You can create them here and monitor them under job runs. Delta Live Tables, or DLT, is a new offering by Databricks. This is an ETL framework that uses declarative approach to building data pipelines with automated testing and it is still evolving.



Compute menu here lets us create clusters, cluster pools and SQL data warehouses. You can either create an all-purpose or a job compute cluster.

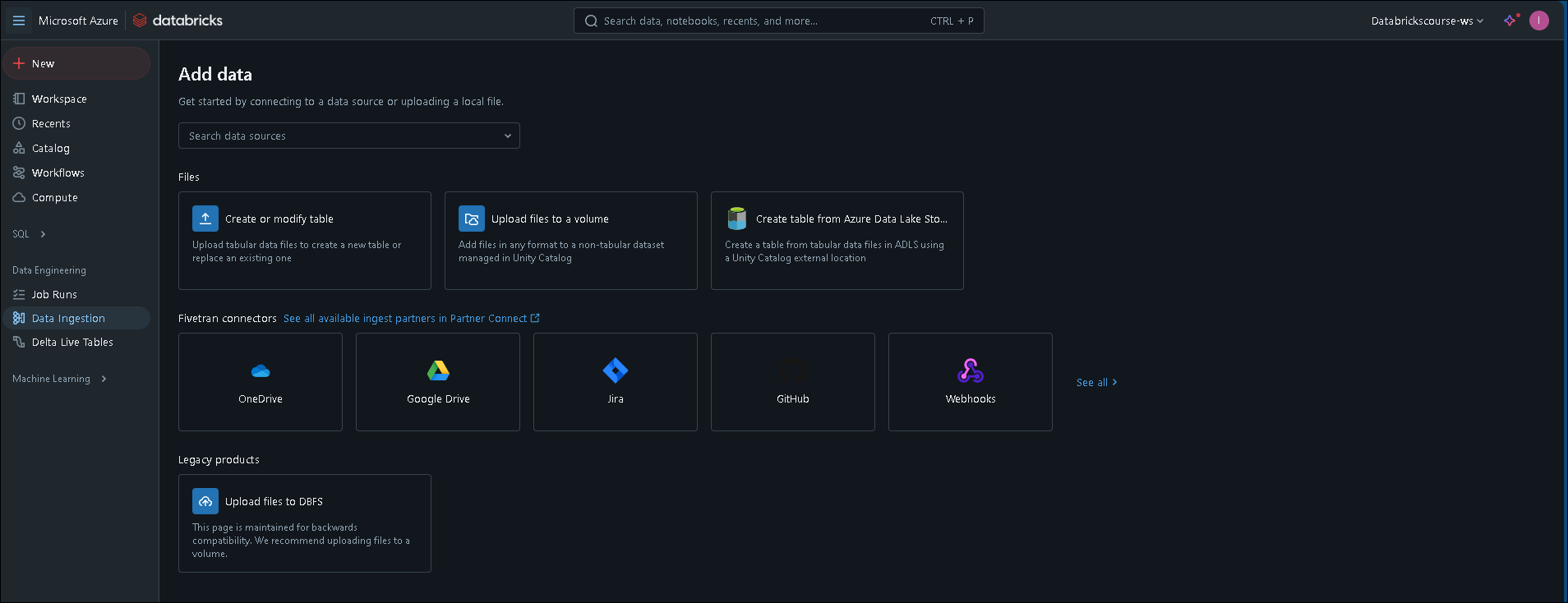


Let's now switch our focus to the data engineering section. To be honest, most of this is redundant and we've already seen them under workflows. Let me navigate to workflows. As you can see, you can navigate to job runs and Delta Live Tables from here. And the default page for workflows is jobs.

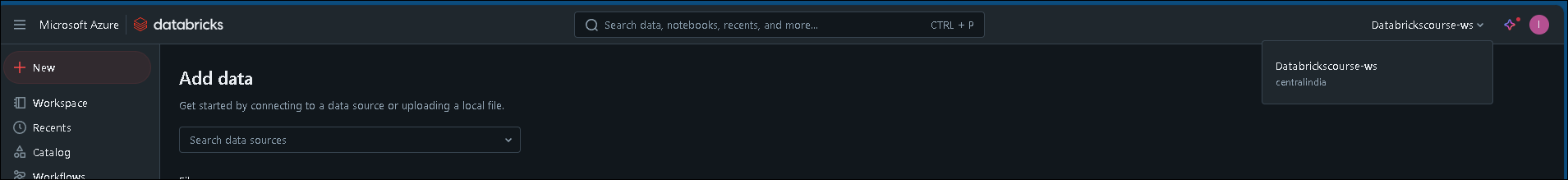
The job runs and Delta Live Tables here under the data engineering also takes us to this page. But when you click on job runs, the default page is job runs, and when you click on Delta Live Tables, the default page is Delta Live Tables but it is exactly the same as what you see under workflows.

And also, you may notice that when you come and click on Delta Live Tables, the title here still says workflows.

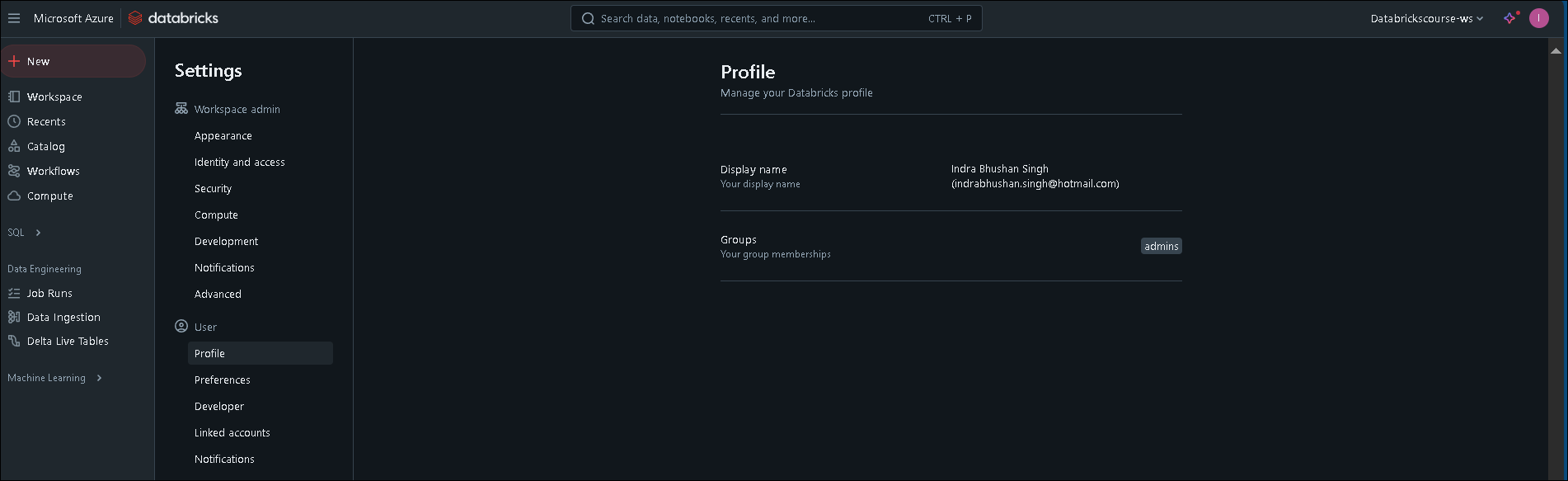
So it is basically a duplicate of what you see under workflows



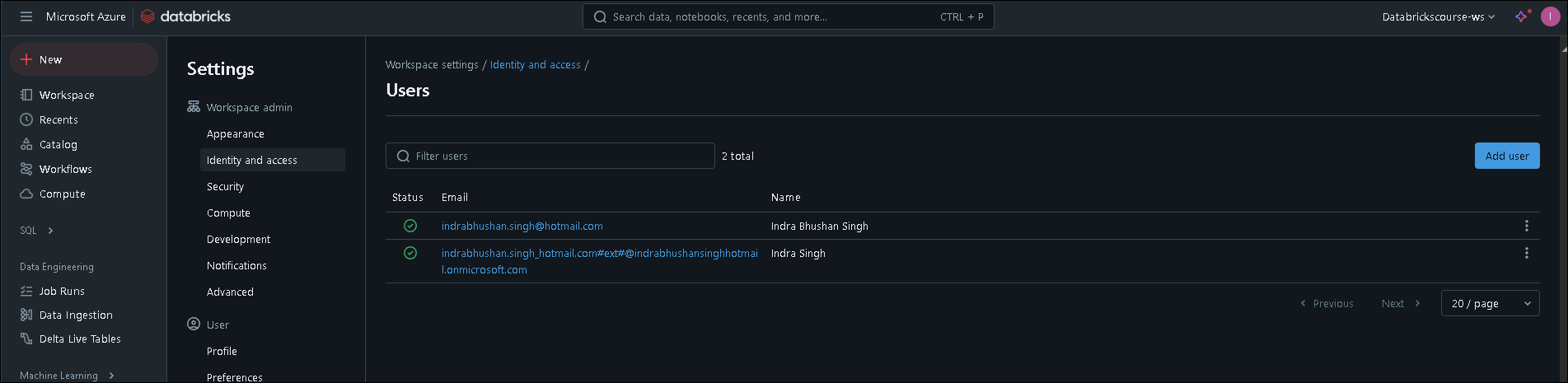
The menu data ingestion lets you create tables from data stored in your local computer. Also, you can get some start notebooks to ingest data from one of these sources and you can find some sample data here too. It also lets us develop starter ingestion pipelines using the third-party partner called Fivetran. But please note that Fivetran is outside of the scope of this course and it's a separate product all together.



The search bar at the top is very powerful and it lets you search within the entire workspace. By clicking on the advanced search, you can also choose to search within specific Databricks assets, such as notebooks, jobs, folders, et cetera. The menu option here lets you search to the other Databricks workspaces. We're currently in the Databricks course workspace. From here, we can switch to different workspaces without going to the Azure portal.

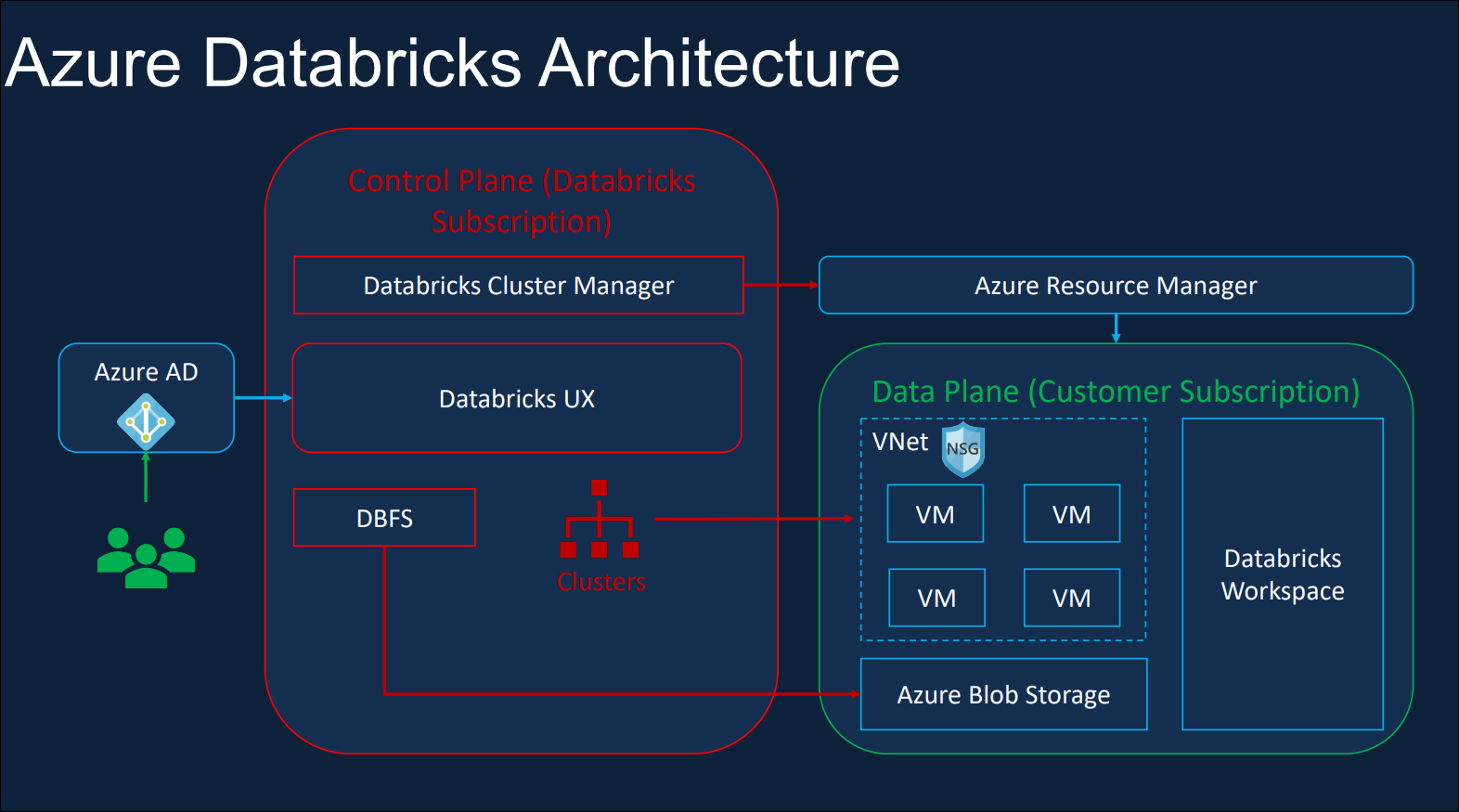


User settings lets us set the preferred language and also, you can set the developer settings here. You can also enable Git integration for this workspace



and the identity and access here gives us the ability to manage users and groups, as well as workspaces and warehouse settings.

**Azure Databricks Architecture**



Databricks Architecture is basically split into two parts, one called the Control Plane and another one called the Data Plane. Control plane is located in Databricks own subscription. This contains the Databricks UX and also the Cluster Manager. It's also home to the Databricks File System (DBFS) and also metadata about Clusters, Files mounted, etc. Data Plane is located in the customer subscription.

When you create a Databricks service in Azure, there are four resources created in your subscription, a Virtual Network and Network Security Group for the Virtual Network. Azure Blob Storage for the default storage and also a Databricks Workspace.

We've just created the Databricks Service.

So let's switch over to the Azure Portal and have a look at those.The Databricks uses such as Data Engineers, Data Scientist and Data Analyst, will use Active Directory Single Sign On, to access the Databricks service. When a user request for a Cluster to be created, the Databricks Cluster Manager will create the required virtual machines in our subscription via the Azure Resource Manager.

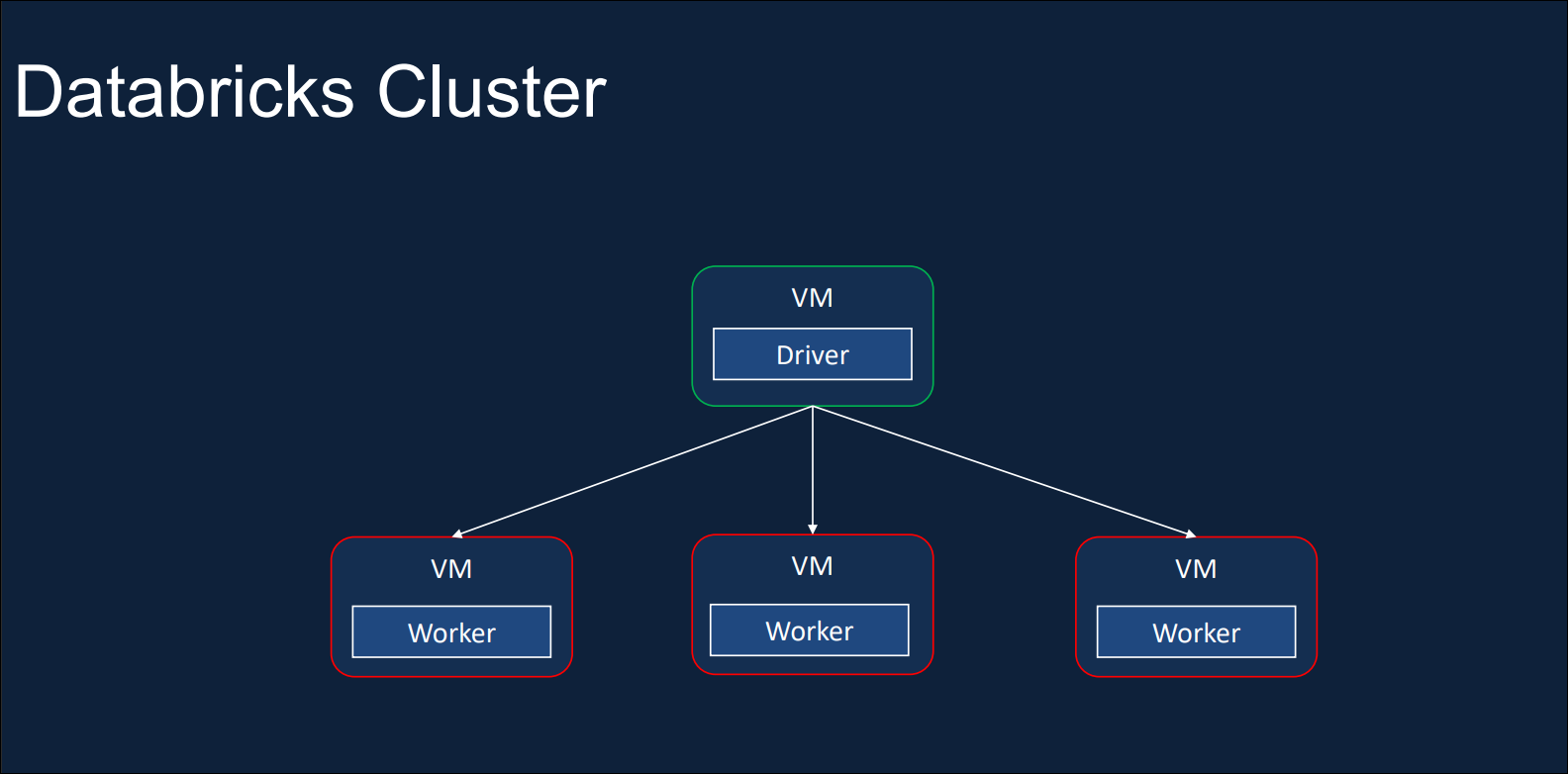
So none of the customer data leaves a subscription. Temporary outputs such as running a display command or data for manage tables, are stored in the Azure Blob Storage, and the processing also happens within the VNet in our subscription.

Blob Storage, and the processing also happens within the VNet in our subscription.

The Azure Blob Storage we have shown here is the default storage or otherwise called the DBFS a route,

and it's not recommended as a permanent data storage.

**Databricks Clusters**



A Cluster is basically a collection of Virtual Machines. In a Cluster, there is usually a Driver node,

which orchestrates the tasks performed by one or more worker nodes.

Clusters allow us to treat this group of computers, as a single compute engine via the Driver node.

Databricks Clusters enable us to run different types of workloads, such as ETL for Data Engineering,

Data Science and Machine Learning workloads.

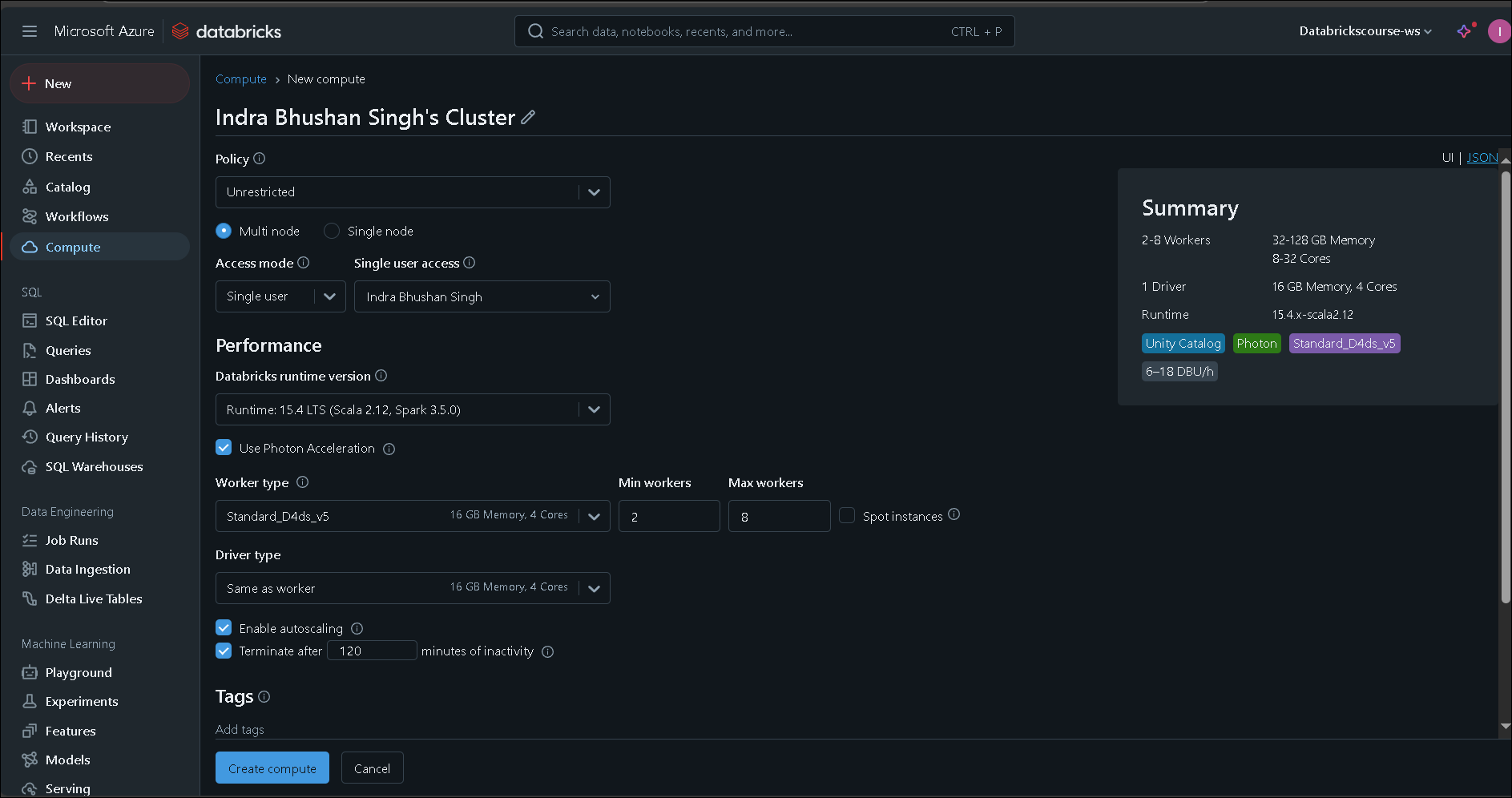
Databricks offers two types of Clusters.

* The first one is the **All purpose Cluster**, which is created manually, via the Graphical User Interface, the CLI or the API. Whereas, the **Job Clusters** are created when a job starts to execute, and the job has been configured to use a Job Cluster.
* All Purpose Clusters are persistent, they can be terminated and restarted at any point in time, whereas the Job Clusters are terminated at the end of the job. They cannot be restarted. So they're no longer usable once the job has been completed.
* All Purpose Clusters are suitable for interactive and ad-hoc Analysis workloads. On the other hand, Job Clusters are suitable for automated workloads, such as running an ETL pipeline or Machine Learning workflow at a regular interval.
* All Purpose Clusters can be shared among many users, and they are good for collaborative analysis, whereas the Job Clusters are isolated just for the job being executed.
* All Purpose Clusters are expensive to run compared to the Job Clusters.
* In summary, All Purpose Clusters are great for interactive analysis and ad-hoc work, whereas Job Clusters are great for repeated production workloads.

**Cluster Pool:** Cluster Pools give you the ability to set aside some ready to use compute capacity, so that when you create an All Purpose Cluster, it can be created quickly. Usually when you create a Cluster, it takes about 5 to 6 minutes to spin up a Cluster. In order to speed up that time, you can have a pool of resources waiting for you via Cluster Pools. And that's where Cluster Pools come in.  
  
**Cluster Policy:** When creating a Cluster, there is a ton of configuration options to specify.

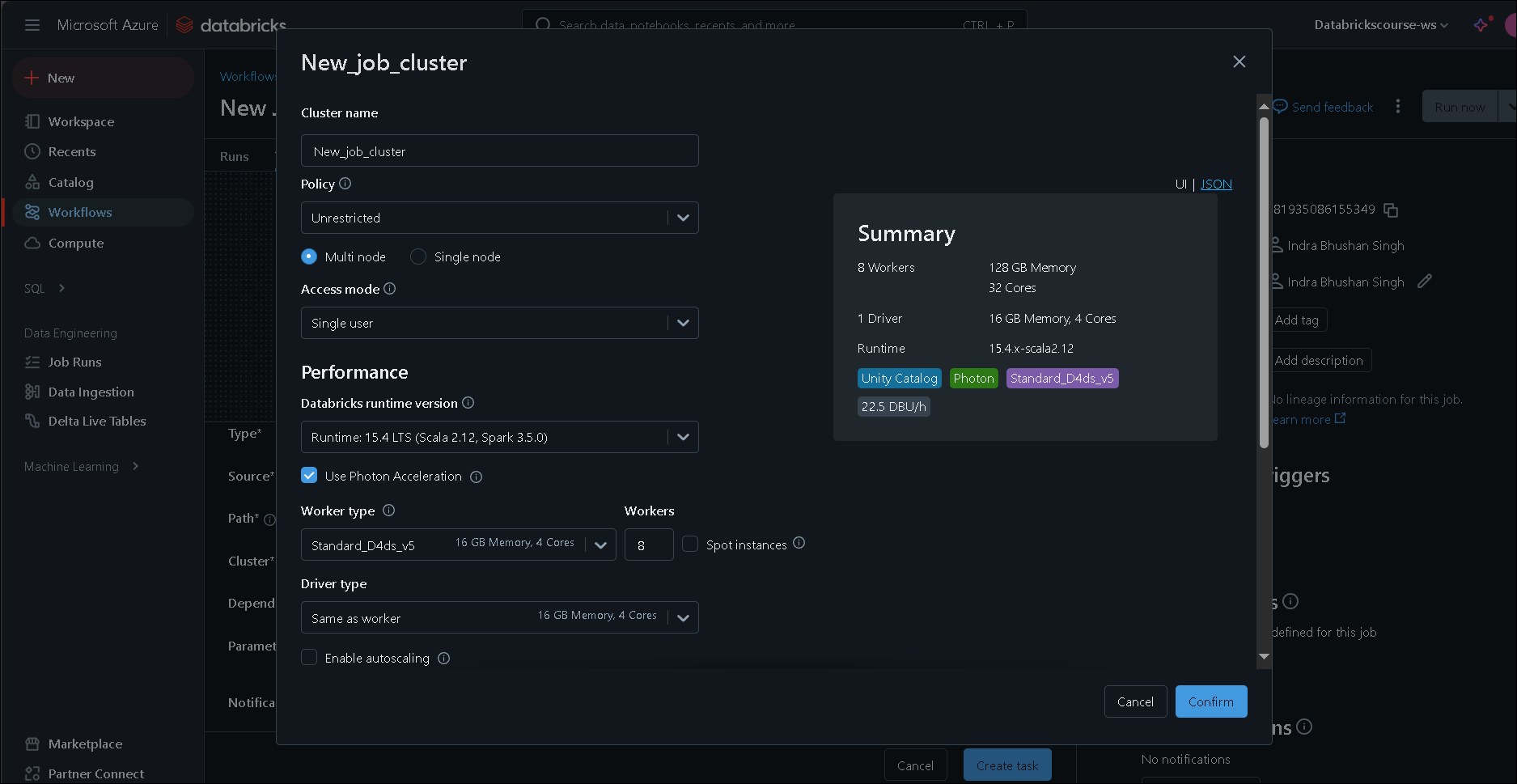
Cluster Policy helps us pre-configure some of these details, so that creating a Cluster becomes simpler,

and also, it helps restrict the maximum size of the Clusters being created to keep the cost under control.



As we said before, All Purpose Clusters can be created manually, and we do that by clicking the Create

compute button here, and we can specify the required configuration details here.



Let's now navigate to the Job's compute. As you can see, Job compute is missing the Create button.

That's because the Job Clusters cannot be created manually, Job Clusters are created when a job starts to run and it's destroyed as soon as the job completes. You can create Databricks Jobs, as part of workflows by selecting the Workflow icon here on the sidebar. As you can see, you can create Jobs and the Job Cluster by selecting the New Job Cluster menu here. As we said, the Cluster will be created when the job runs and destroyed as soon as it completes.

**Create Cluster**

When we come to create the Cluster, We will be presented with a number of configuration options as shown here. In this lesson, I'll walk you through each one of these in detail. First, we have the option to choose whether we want to create a Single Node or a Multi Node Cluster.

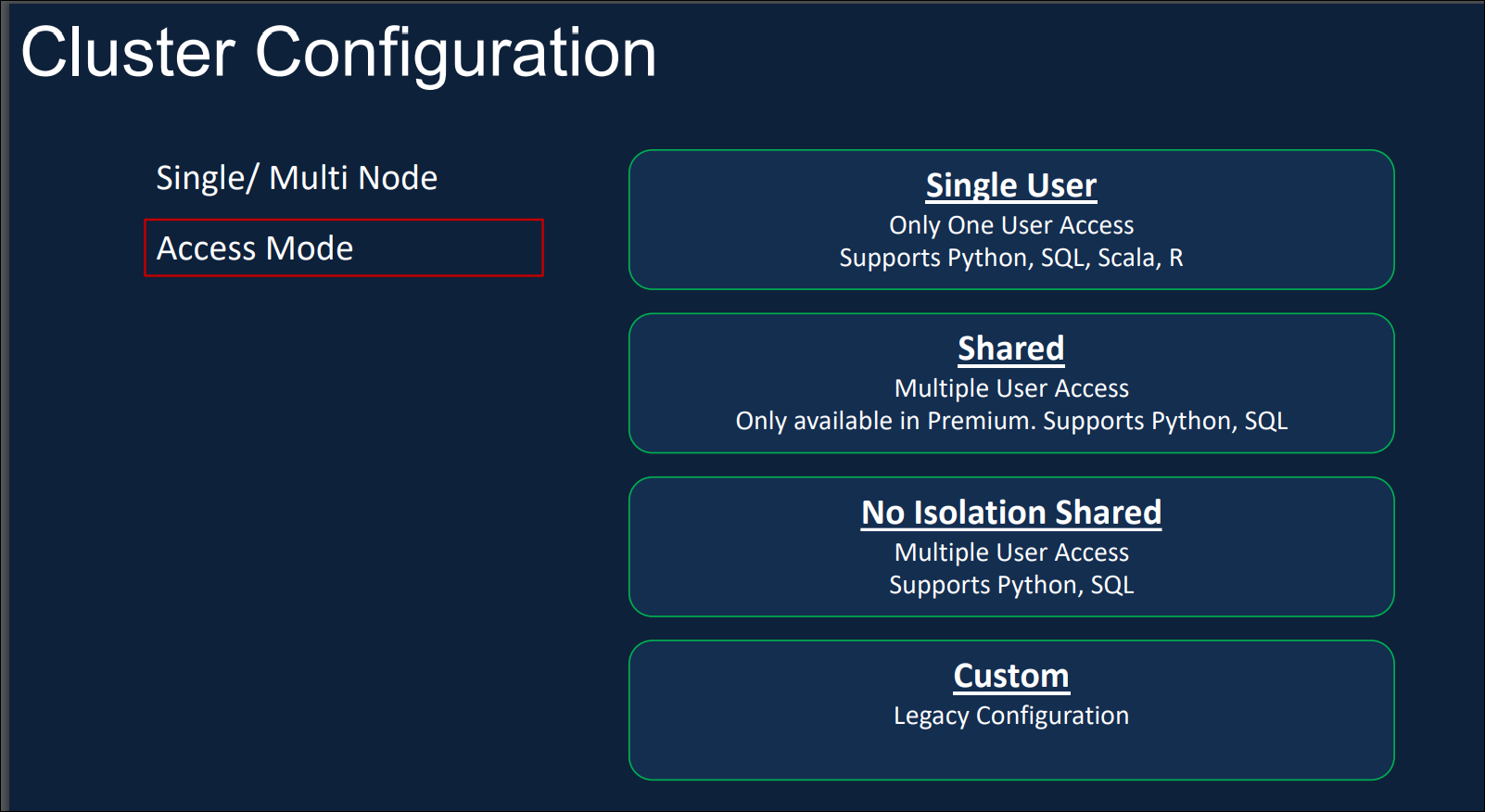
* Multi Node Cluster will have one Driver Node and one or more Worker Nodes. When you run a Spark Job against a Multi Node Cluster, the Driver Node will distribute the tasks to run on the Worker Nodes in parallel, and returns the result. They give us the ability to horizontally scale the Cluster depending on your workload. We can basically keep adding Worker Nodes as we need. These are the default type of Clusters used for Spark Jobs and suitable for large workloads.
* On the other hand, Single Node Cluster will have only one node, which is the Driver Node and there are no Worker Nodes. Even though, there are no Worker Nodes, Single Node Clusters also supports Spark workloads. When you run a Spark Job, the Driver Node acts as both the driver and the worker. As there are no Worker Nodes, the Single Node Clusters are not horizontally scalable, so they're not suitable for large ETL workloads. They're mainly targeted for lightweight Machine Learning and Data Analysis workloads which don't require, any distributed compute capacity.

**Create Access Mode**

We then need to define the Access Mode.

There are four different types of Access Modes available at the moment for the Cluster.

* As the name suggests, Single User access mode only allows a single user to access the Cluster. It supports all four languages Python, SQL, Scala, and R.
* Shared access mode allows the Cluster to be shared amongst more than one user, but it provides process isolation. Each process gets its environment, so one process can't see the data or the credential used by the other one. It's only available on premium workspaces. Also, it only supports Python and SQL workloads.
* No Isolation Shared also allows the Cluster to be shared amongst more than one user. It's available on both standard and premium workspaces. Also, it supports all four languages Python, Scala, SQL and R.
* The main difference between this and the Shard access mode is that, No Isolation Shared access mode doesn't provide any process isolation. So failure in one user's process may affect the others.
* Also, they don't offer any task preemption, so one running process may use all the resources and the others may fail. And most importantly, as everything is shared, it's considered less secure.
* Custom access mode is not an option,



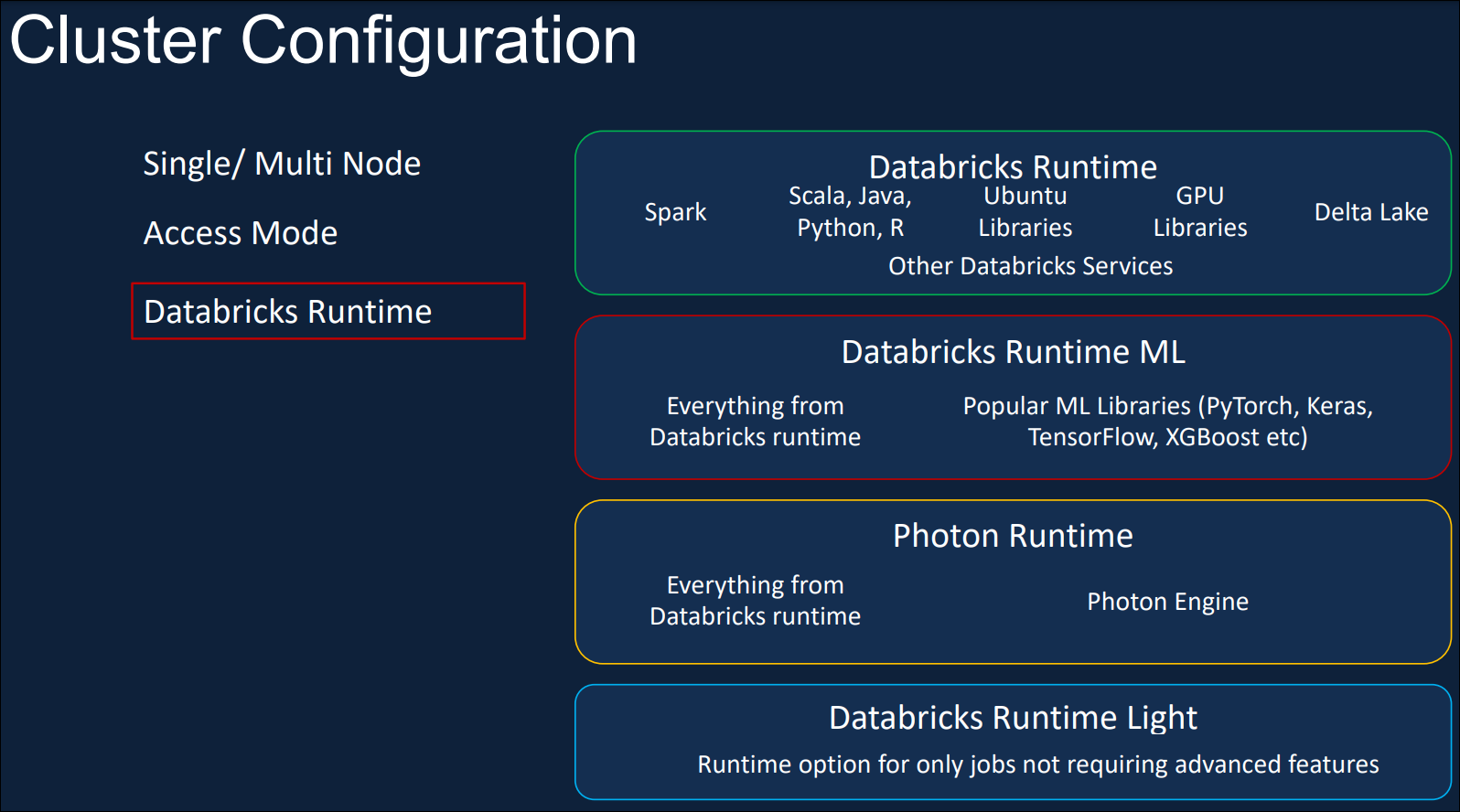
**Databricks Runtime**

. Databricks runtimes are the set of core libraries that run on Databricks Clusters. At the time of recording, Databricks offers four types of runtimes.

**Databricks Runtime, Databricks Runtime ML, Photon Runtime and Databricks Runtime Light.**

Let's go through each one of these.

* Databricks Runtime includes an optimized version of Apache Spark Library. Java, Scala, Python and R Libraries, Ubuntu and its accompanying system Libraries, GPU Libraries for GPU enabled Clusters, Delta Lake Libraries and also other Libraries for Databricks services that integrate with other components of the platform such as Notebooks, Jobs and Cluster Manager.
* Databricks Runtime ML includes all the libraries from the Databricks Runtime, plus the popular ML Libraries such as PyTorch, Keras, TensorFlow, XGBoost, etc..
* Photon Runtime also includes all the libraries from the Databricks runtime, plus the Photon Engine, which is the Databricks native vectorized query engine, that runs SQL workloads faster and reduces your cost per workload.
* Databricks Runtime Light is the runtime option for only jobs not requiring advanced features such as auto scaling, reliability and improved performance. Also, it's only suitable for Automated Workloads. You can't use it for Interactive Workloads or Notebook Jobs.
* Auto Termination is a nice feature that will avoid unnecessary costs on idle Clusters.



**Auto Termination**

Auto Termination is a nice feature that will avoid unnecessary costs on idle Clusters. It's especially useful on Ad-hoc clusters for preventing them, running during evenings and weekends when they're not in use. You can specify when to terminate your Databricks Cluster, if the cluster has not been in use. It will be terminated after the number of minutes specified. Default value for Auto Termination is 120 minutes, but you can change the value. The accepted values range from 10 to 10000 minutes.

When you create a Multi Node Cluster, you can specify the minimum and the maximum number of Worker

Nodes.

**Auto Scaling:**

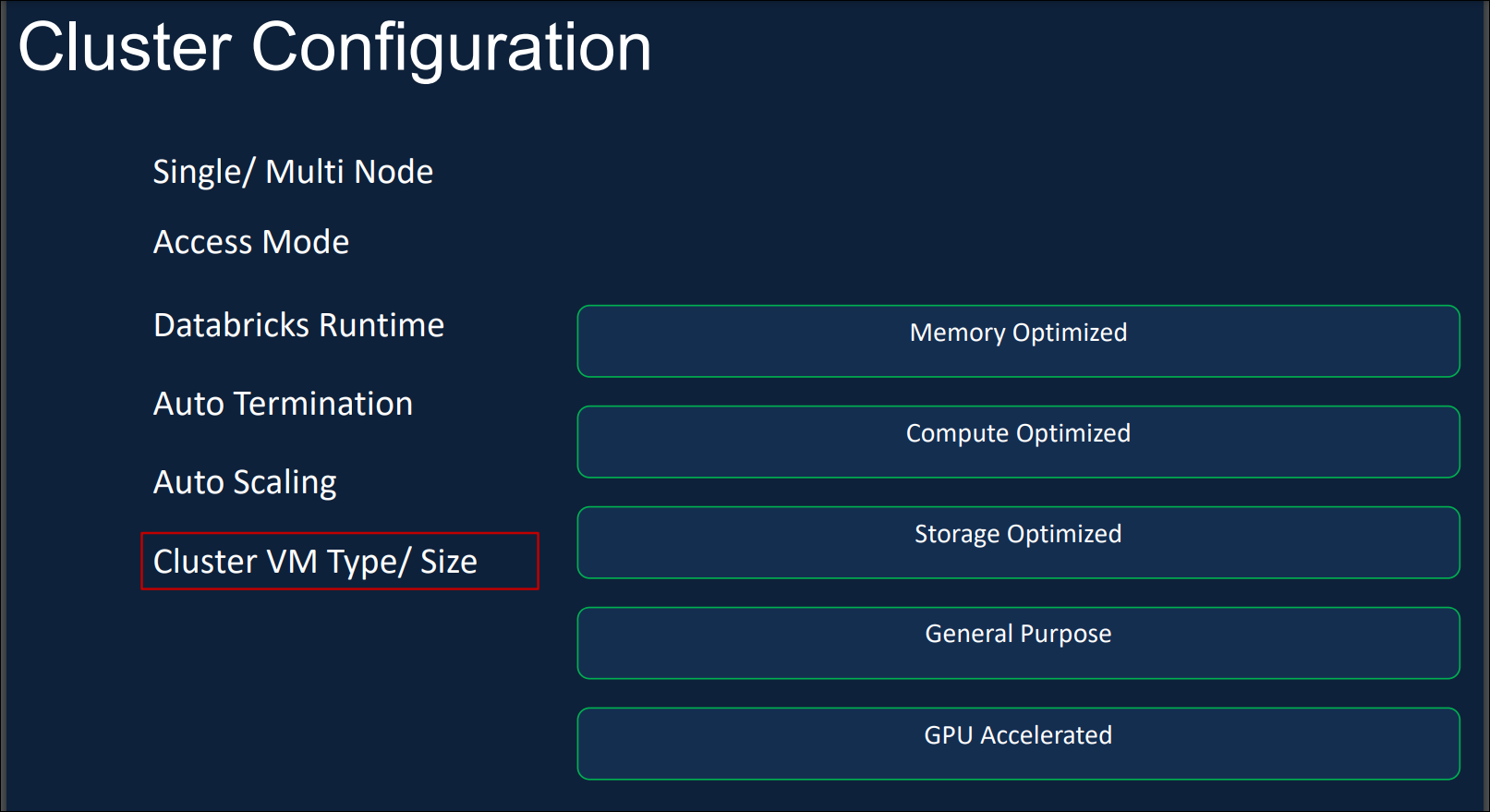
Auto Scaling will automatically add or remove nodes from the Cluster depending on your workload. This can result in optimum utilization of the Cluster. This is especially useful if you're unsure about the workload upfront or your workload changes throughout the process. They're not recommended for streaming workloads, even if specified Databricks defaults to the maximum number of Worker Nodes.

**Cluster VM Types and Size:**

There are a wide array of Azure VM types available for us to use.

Databricks groups them into small number of easy to understand groups.

* **Memory Optimized** instance types are recommended for memory intensive applications. For example, a Machine Learning workload that caches a lot of data in memory.
* **Compute Optimized** instance types can be useful for structured streaming applications, where you need to make sure that the processing rate is above the input rate at peak times of the day. These can also be used for Distributed Analytics and Data Science Applications.
* **Storage Optimized** instance types are recommended for use cases requiring high disk throughput and I/O.
* **General Purpose** instance types are recommended for Enterprise Grade applications and analytics with In-memory caching.
* **GPU Accelerated** instance types are recommended for Deep Learning Models, that are data and compute intensive.



**Cluster Policy:**

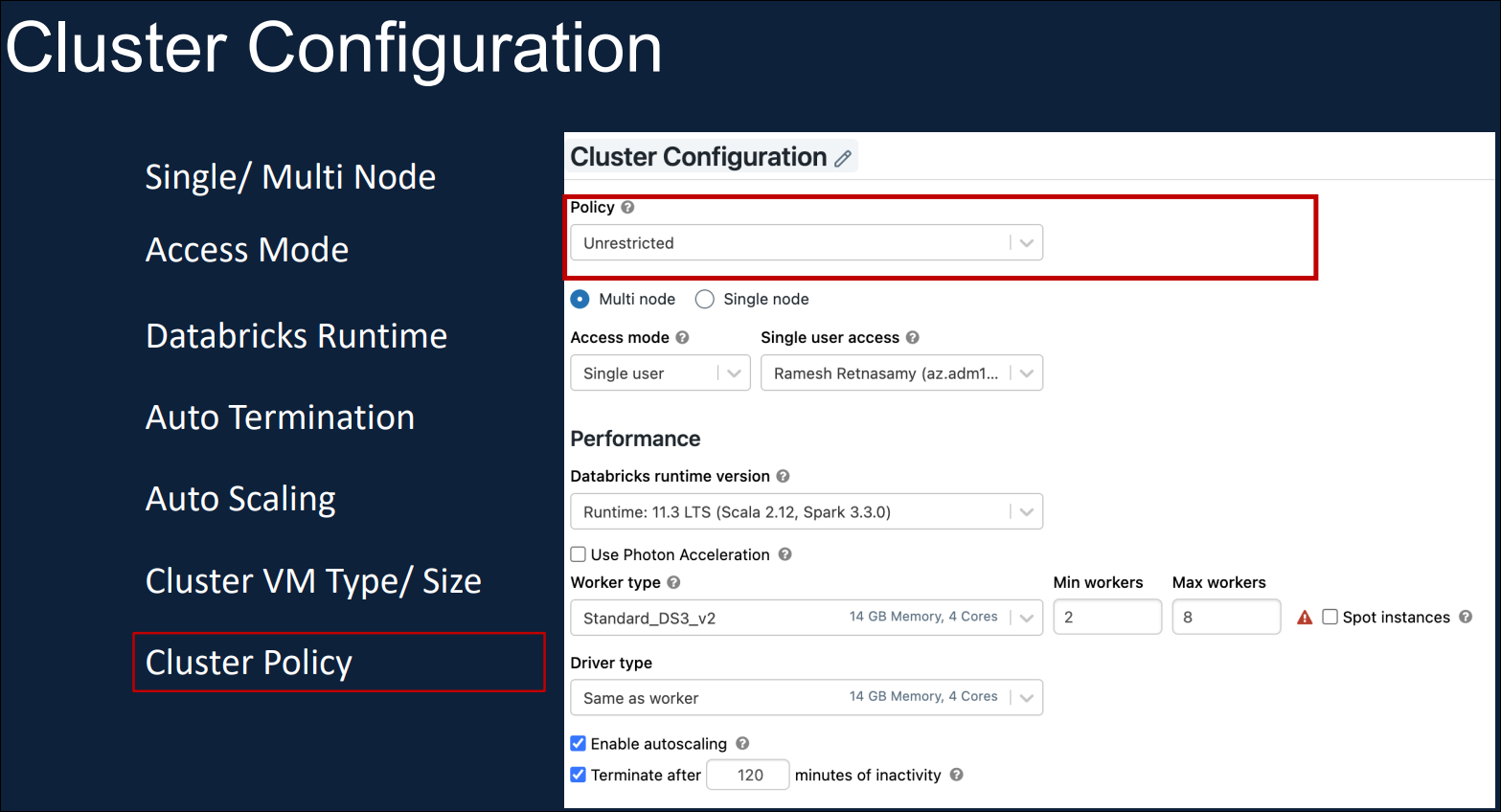
The final configuration option is Cluster Policy. As you have seen, there are a lot of options to choose from when you are configuring a Cluster. This could easily overwhelm a Data Engineer or a Machine Learning Engineer, and creating Clusters become the sole responsibility of the administrator. Because it's too difficult to configure for a Standard Data Engineer or a Machine Learning Engineer.

Also, without careful consideration, users could accidentally create Clusters which are oversized

and too expensive to run.

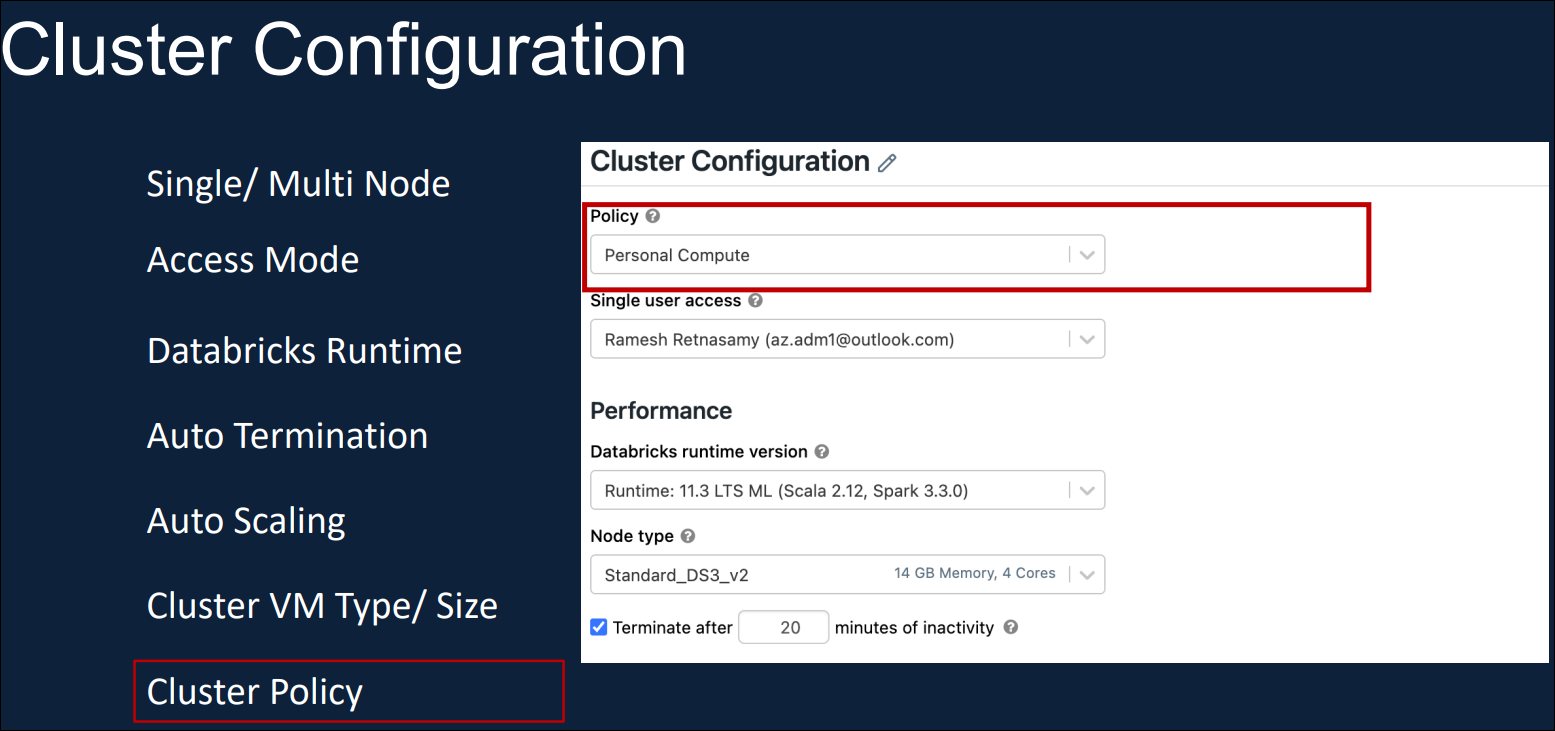
Cluster Policies help us avoid these common issues. Administrators can create Cluster policies with restrictions, and assign them to users or groups.

In this example below , we haven't selected a Cluster Policy and it's left as unrestricted. As you can see, when a Cluster Policy is selected, the configuration becomes much more simplified.



In this example, a Personal Compute cluster policy has took up the option of Multi Node, and the user

can only create a Single Node Cluster. Also, it defaulted the runtime version to ML Runtime, limited the node types and also Auto Termination set to 20 minutes.

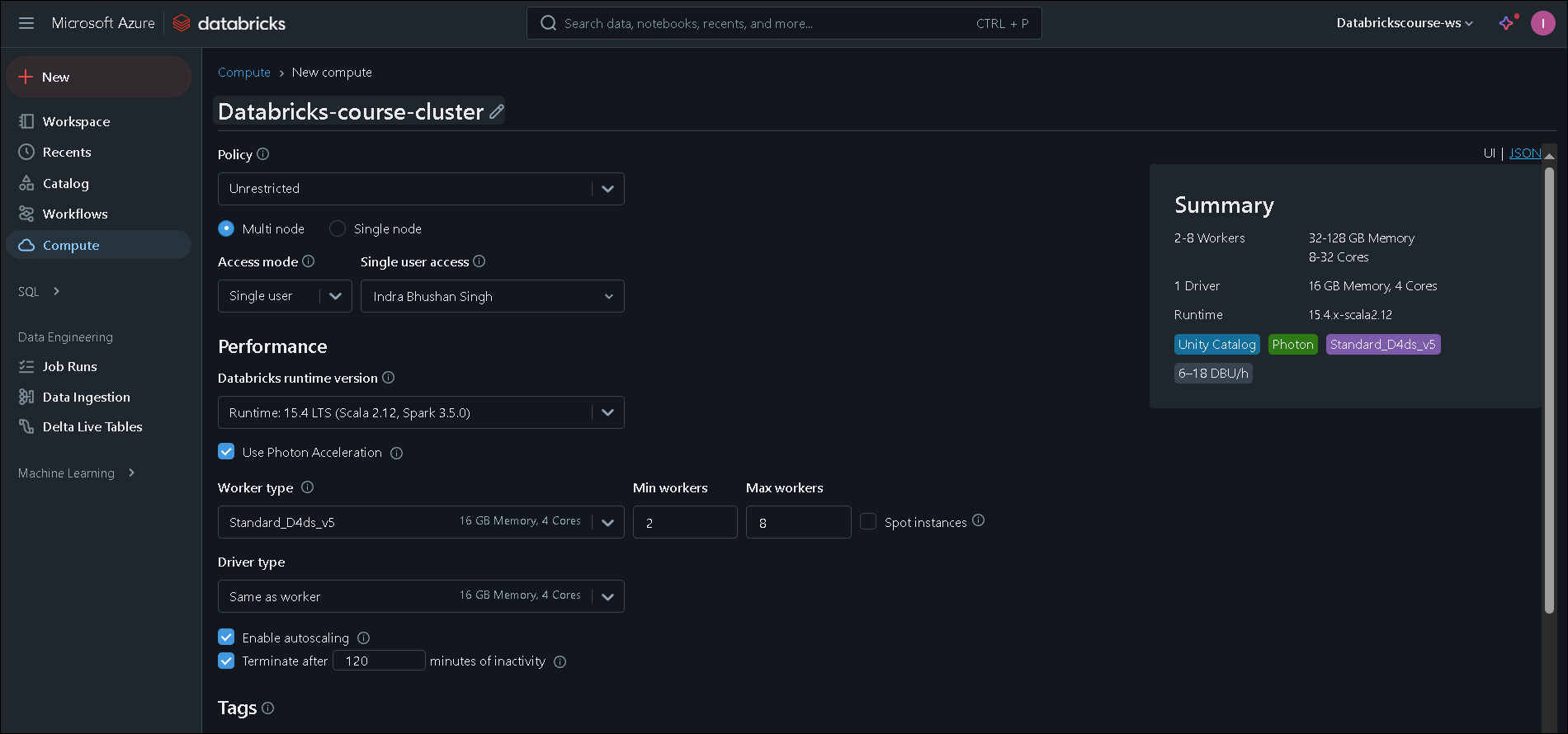


We can assign this to ML Engineers and they'll be able to create only these kinds of small Clusters.

In summary, Cluster policies simplify the user interface, thus enabling standard users to create Clusters

and take away the need for administrators to be involved in every decision. And most importantly, it achieves cost control by limiting the maximum size of the Clusters. This only available on premium tier.

**Creating Cluster**



If your workspace is premium tier, you will see Cluster Policy as the first configuration option.

As we said earlier, standard tier doesn't offer Cluster policies. As you can see, there are three Cluster policies already made available by Databricks to choose from. Any new policies we create will also be made available here for us to choose from, but we're going to leave it as unrestricted, so we can go through each of the configuration options ourselves. As we discussed before, we have the option to choose between Multi node or a Single node cluster. For this course we just need a Single node cluster as we are dealing with only a small amount of data, and it's also cheaper to run a Single node cluster. Also, students on free tier and student subscription will only have sufficient quota available to create a Single node cluster. So we'll create a Single node cluster. But before that, I want to talk you through the additional configuration options for creating a Multi node cluster.

As you can see, selecting between a Multi node and a Single node changes the form, with different configuration options. The main difference between the two forms is around node types and being able to auto scale a Multi node cluster.

You can select the Compute option for the Worker node and specify the minimum and the maximum number of Worker nodes. You can change the minimum and the maximum values depending on your workload.

In this example, when the cluster starts up, it will be allocated with two Worker nodes and additional

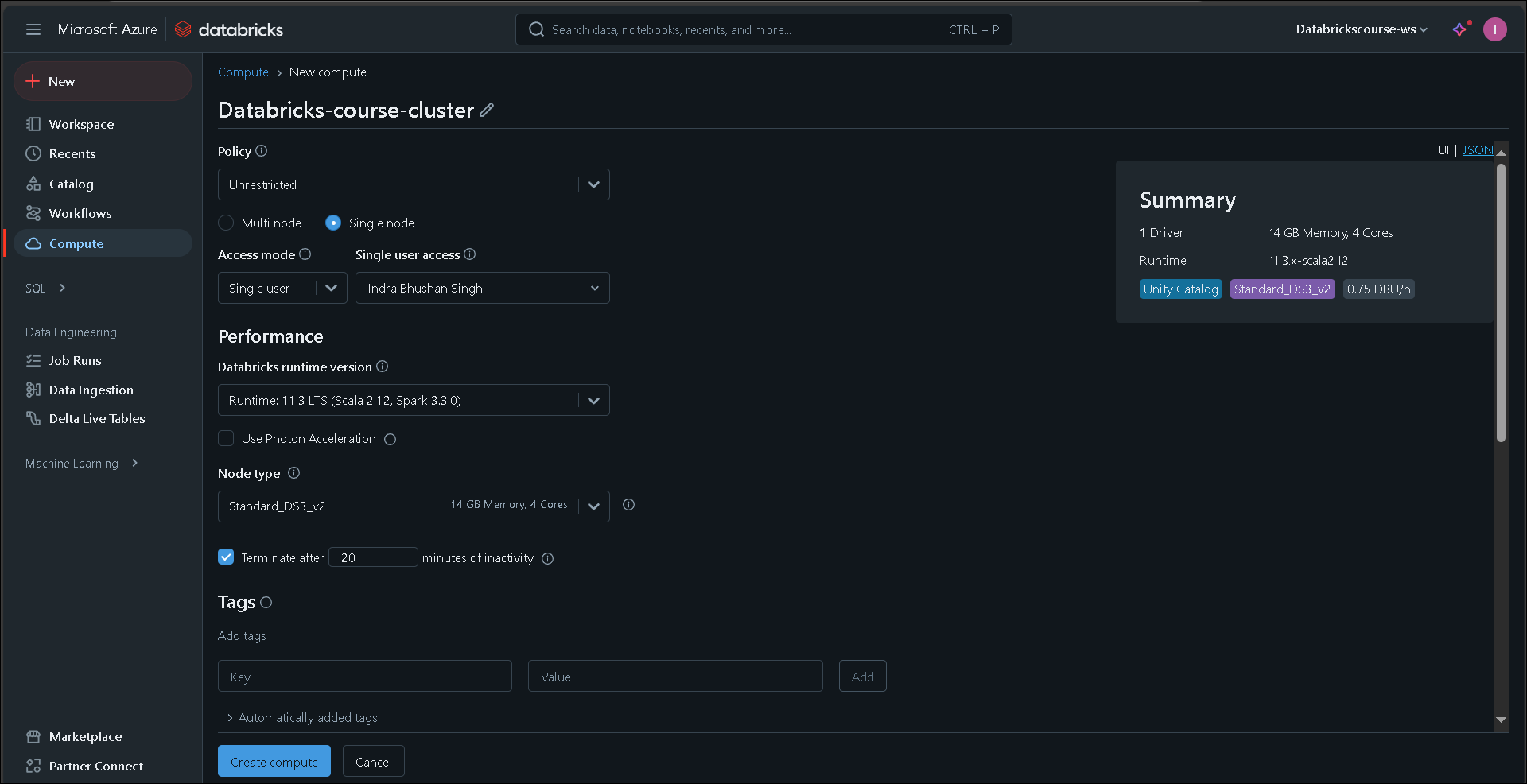
workers are added depending on the workload. But it'll never exceed eight. This ensures that you're only paying for additional capacity when there is a need, and also your expenses are capped at eight nodes.

You can disable auto scaling by clicking on the tick box here. A typical use case for this is a streaming workload in which the time it takes to scaling up a cluster, might not be acceptable for the real time workload. You can also request to use unused Azure capacity via Spot instances if it's available, by selecting

the tick box here. This will save you cost of running your application, but please be mindful that you could be evicted, from Spot instances when they become unavailable.

I would recommend using them, for development purposes or, non critical workloads only.

Now that we've discussed the configuration specific for Multi node cluster,



let's switch to Single node cluster and that's what we want to create. As we discussed in the previous lesson, there are three types of Access modes. Single user, Shared and No Isolation Shared.

If you are using standard tier, you will only be seeing Single user and No Isolation Shared, because

Shared is only available in a premium workspace. For the most part of our course, we just need the Single user access, because your Active Directory account is the only one which needs access to this Cluster.

Single user access won't be sufficient for that. So let's select the No Isolation Shared access mode.

We can also select Shared, but I'm using No Isolation Shared, so that I'm using the same access mode

used by students on standard workspace.

Let's now select the Databricks runtime version. As you can see, they're nicely grouped into Standard and ML. Runtime versions are generally suffixed with ML. We're doing an ETL project, so let's select a Standard runtime. You may notice that, some of the runtime versions are suffixed with LTS. LTS stands for Long Term Support. LTS versions are released every six months and supported for two years with updates and fixes, so

select them for production workloads. Let's pick 11.3 LTS.

If you wanted to use the highly performing new vectorised query engine called Photon, you can tick the box here.

Clusters with Photon Engines are normally more expensive, but for larger workloads you're likely to save cost as the query is finished quicker and you can terminate the Cluster sooner. We're going to leave this as unticked as we don't have such a large workload.

Let's now select the Node type. As you can see, we have a vast array of nodes ranging from general purpose to optimized for storage or memory as well as GPU enabled. We just need a small node, so let's pick the DS3\_v2 Standard node with 14 gigabytes memory and four cores.

Please note that, sometimes all the nodes are not available in all of the regions, especially for free subscription.

So please select one without a warning sign next to it. Also DS3\_v2 might not be available when you come to do this course. In that case, just select another one with 4 cores.

The tick box here allows you to say when to terminate the Cluster. Please ensure that it is ticked, otherwise the Cluster can only be terminated manually. Within this box you specify, the number of minutes of inactivity before the Cluster can be terminated.

The minimum value is 10 minutes, but I would suggest you giving 20 minutes, so that you will have time

to watch a lesson and then come back to the Cluster to do the lap session, and the Cluster is still running.

It can be annoying to start the Cluster after every lesson because it takes about 5 minutes to restart

a Cluster.

Here you can specify any tags you may want to include. Databricks automatically add some tags by default, but you can also add others to help with your billing and also attributing the cost of running the Cluster with a specific project, for example.

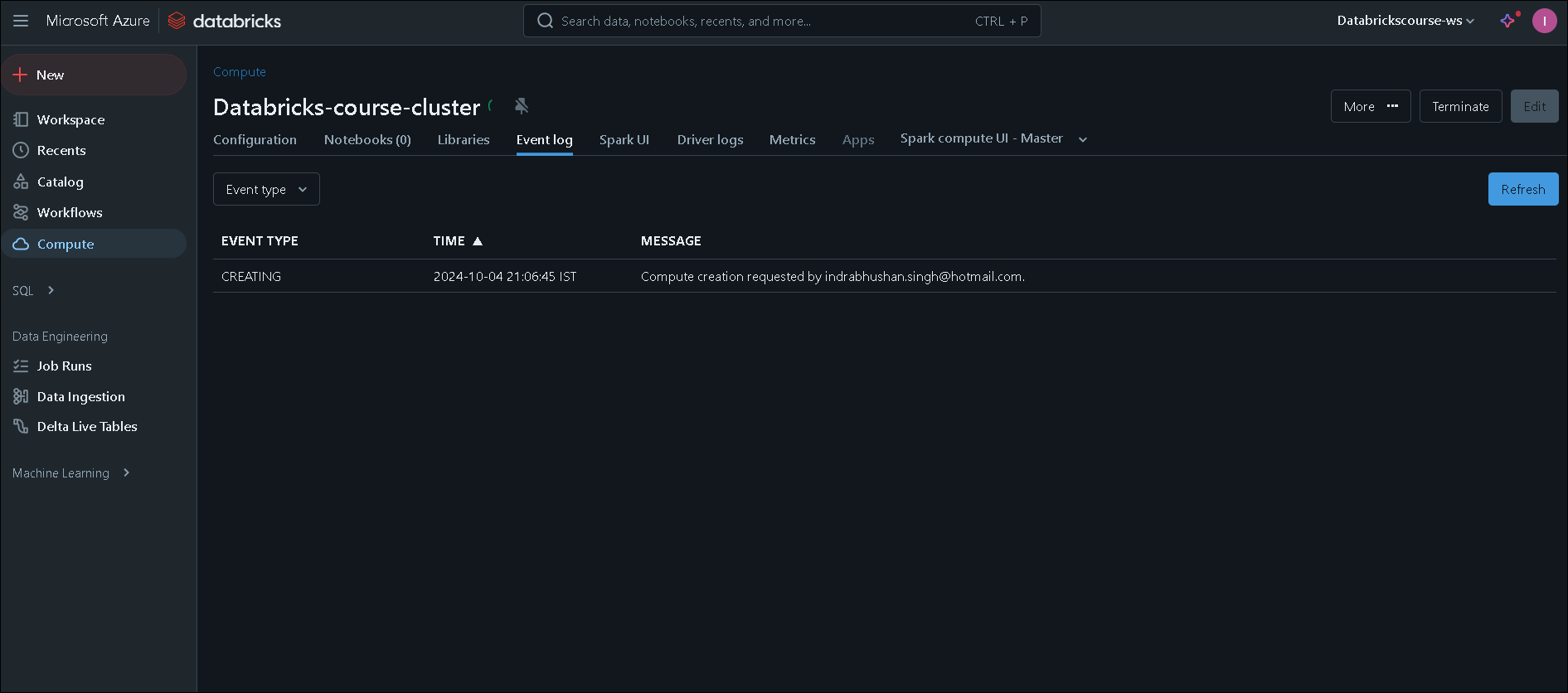
Under advanced options, you can specify your Spark configurations. For example, if you wanted to change your default format from Delta to parquet,you can set up your config here.

You can also set your environment variables here.

Let's navigate now to logging. If you want to save the logs to a DBFS location, you can specify that here and that will keep the logs in long term. I'm going to put that back to none.

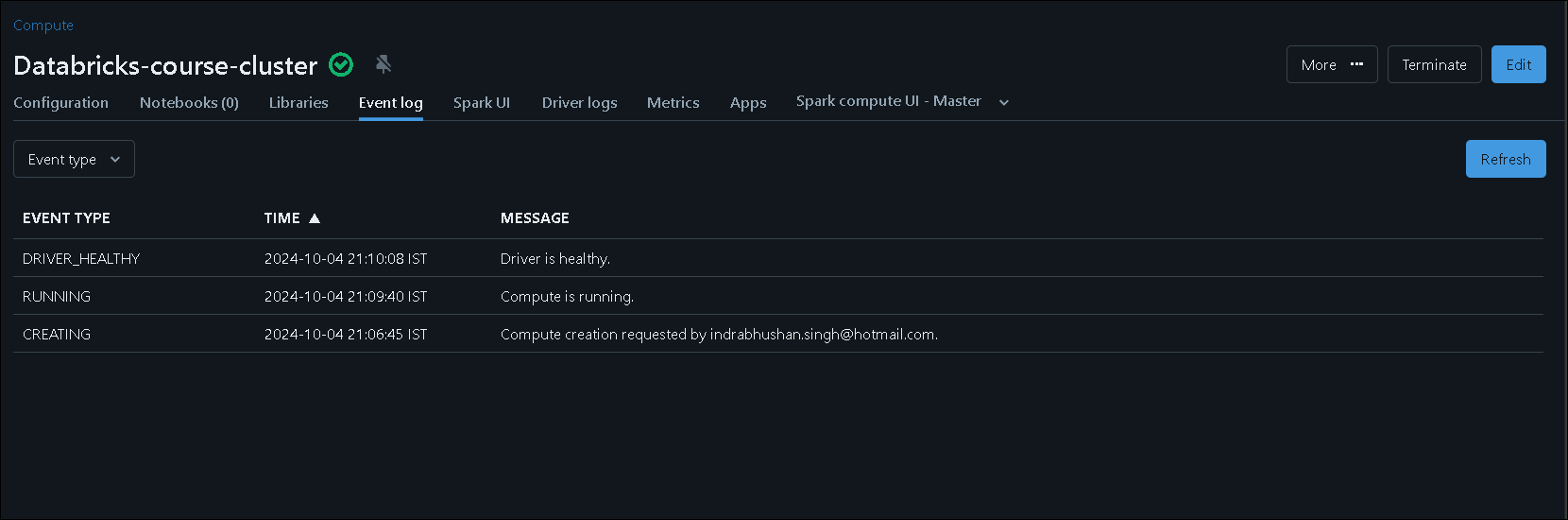
And if you have any Init Scripts, you can add them here. For example, you may want to install some Python packages as soon as the Cluster starts up, so all your developers get the same environment for them to work on. Now that we've configured the cluster.

Let's click on Create Cluster.



As you can see above, the Cluster is now being created, the icon here indicates that it is in progress.

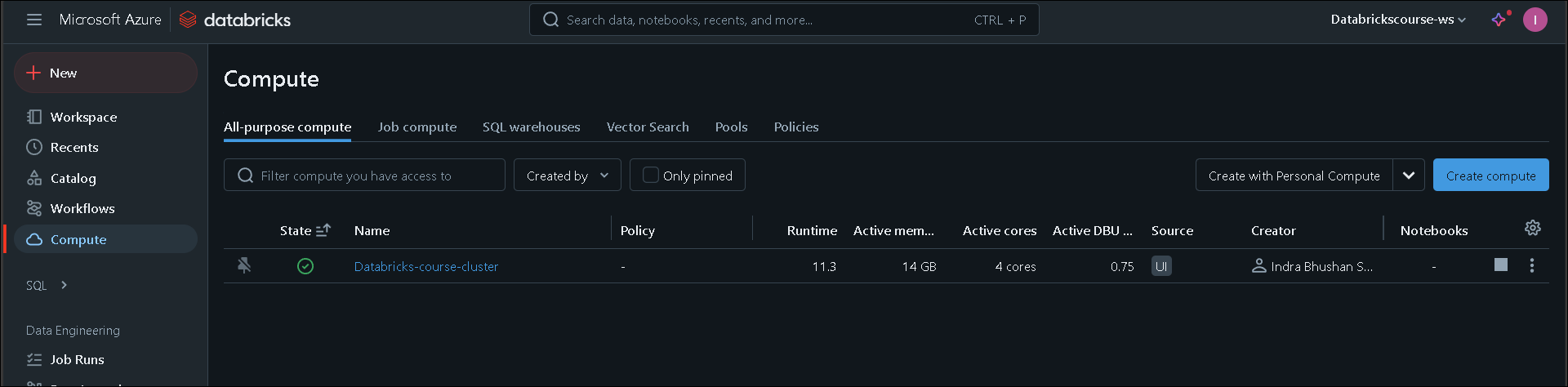
You can also go to Event log and see what's happening. As you can see, it's creating at the moment.



But once it's created, you will see a separate message for that. As you can see, the Cluster is now up and running. It took about 5 minutes. The green tick here indicates that it is running.

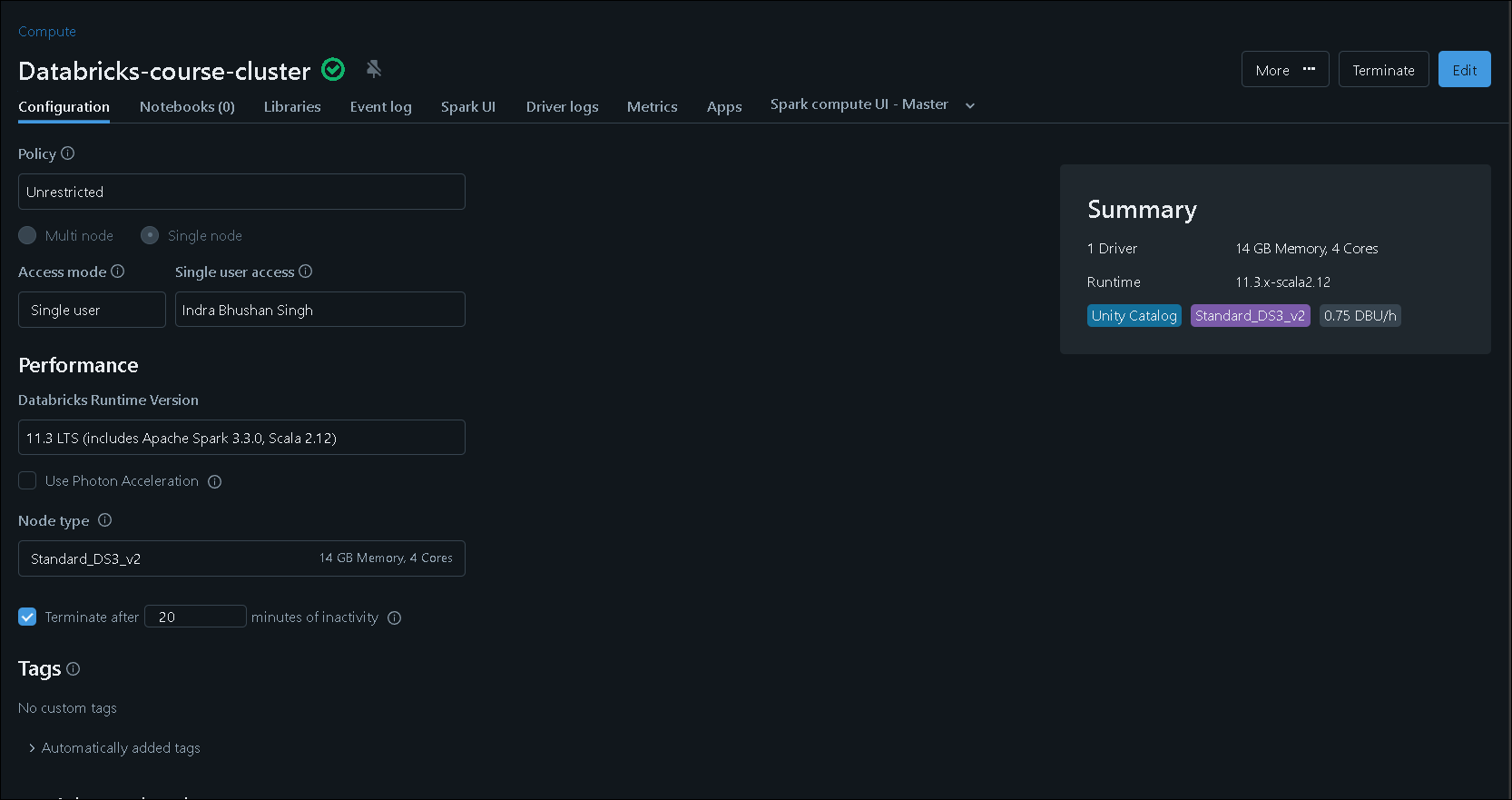
You can also refresh the Event log, so if I do a refresh, you'll see that it's now running and the Driver node is healthy.

By going to Compute, you can see all the Clusters that are being created.

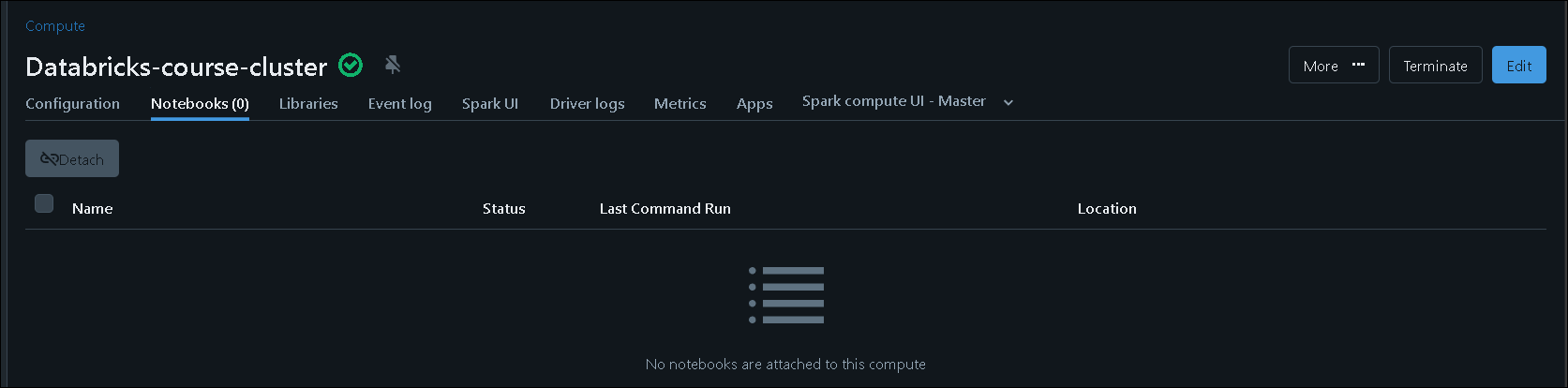


This page also shows you the summary of the cluster details. As you can see, we've got 14 gigabytes of Active memory in this Cluster and it's got 4 cores as well. And we're being charged 0.75 DBU/h.

And you've got the option to stop the Cluster, you can just click on the stop button here or you've got the option to Restart, Clone or Delete the Cluster as well from here.

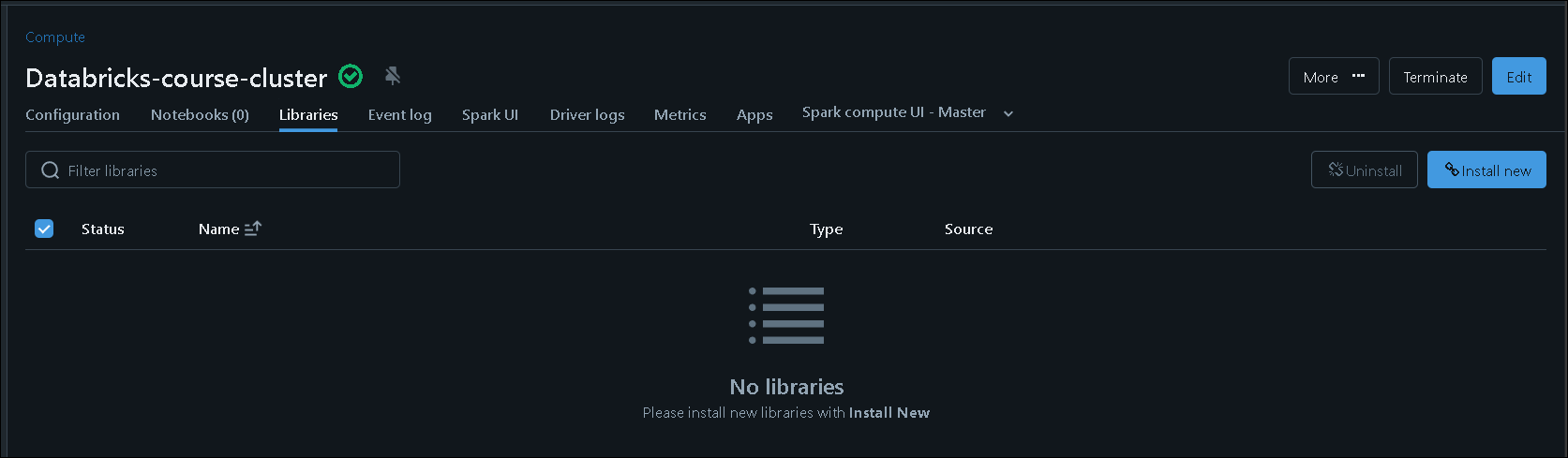


By selecting the name of the Cluster here, you can edit the Cluster, so you are only allowed to edit some of the configuration details, but please note that most of the configuration changes will require the Cluster to be restarted.

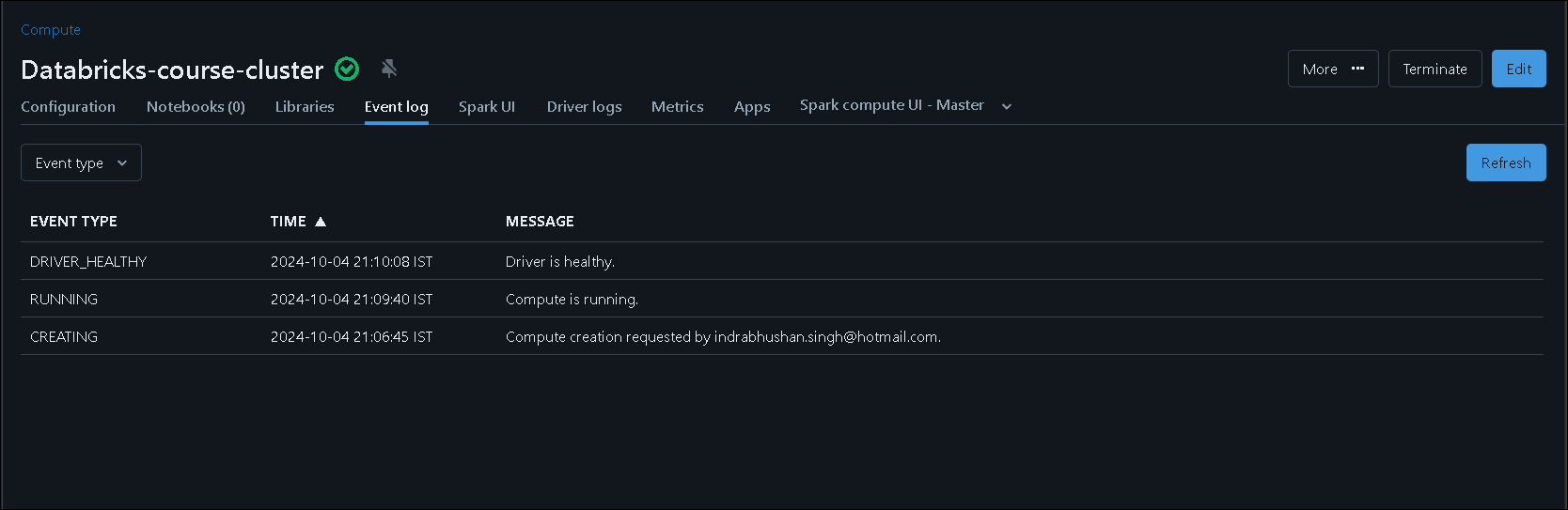


If you had any Notebooks attached to the Cluster, you will see them here. And you can detach them if

you no longer want them to be attached.

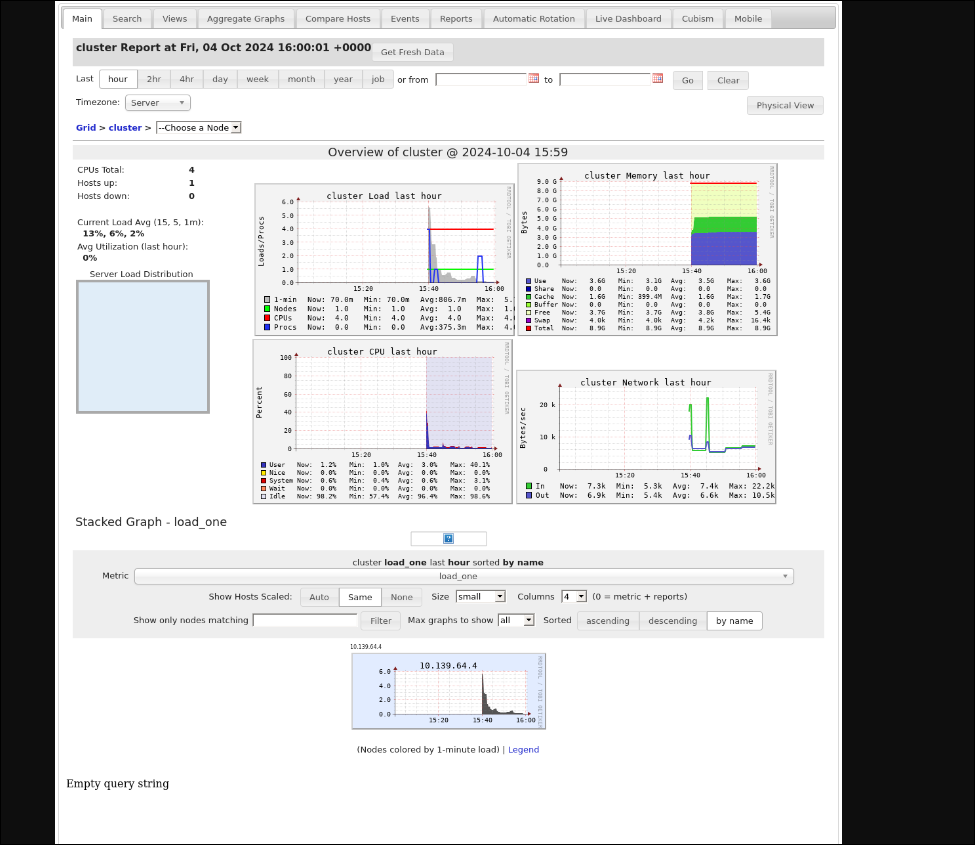


If you want to install any external libraries, you can do that here. You can simply drag and drop any libraries here or add them to the Azure storage and specify the location or provide things like Maven coordinates here.



Even log basically logs any major events that happen to the Cluster. We are seeing standard things here at the moment, but if you had any issues with the Cluster, this is where you will come to start your investigation.

For example, you try to start the Cluster and it doesn't start, you should come here and select the recent event to see the details. At the moment it's just a simple message of Driver is healthy, but if you had an issue, you will see more details here and the error messages and some codes as well, which will help you investigate the issue.



Also, we have the option to view the Spark User Interface, Driver logs and also for Metrics you can use the Ganglia UI. This will be really useful to understand how well the nodes are performing, are they being fully utilized, etc, when you're working with any performance related issues.

Let's go back to the Cluster. Finally, if you look at the summary, you've got the details about the Cluster itself and also the summary includes pricing.

As you can see, this Cluster is charged at 0.75 DBU/h.

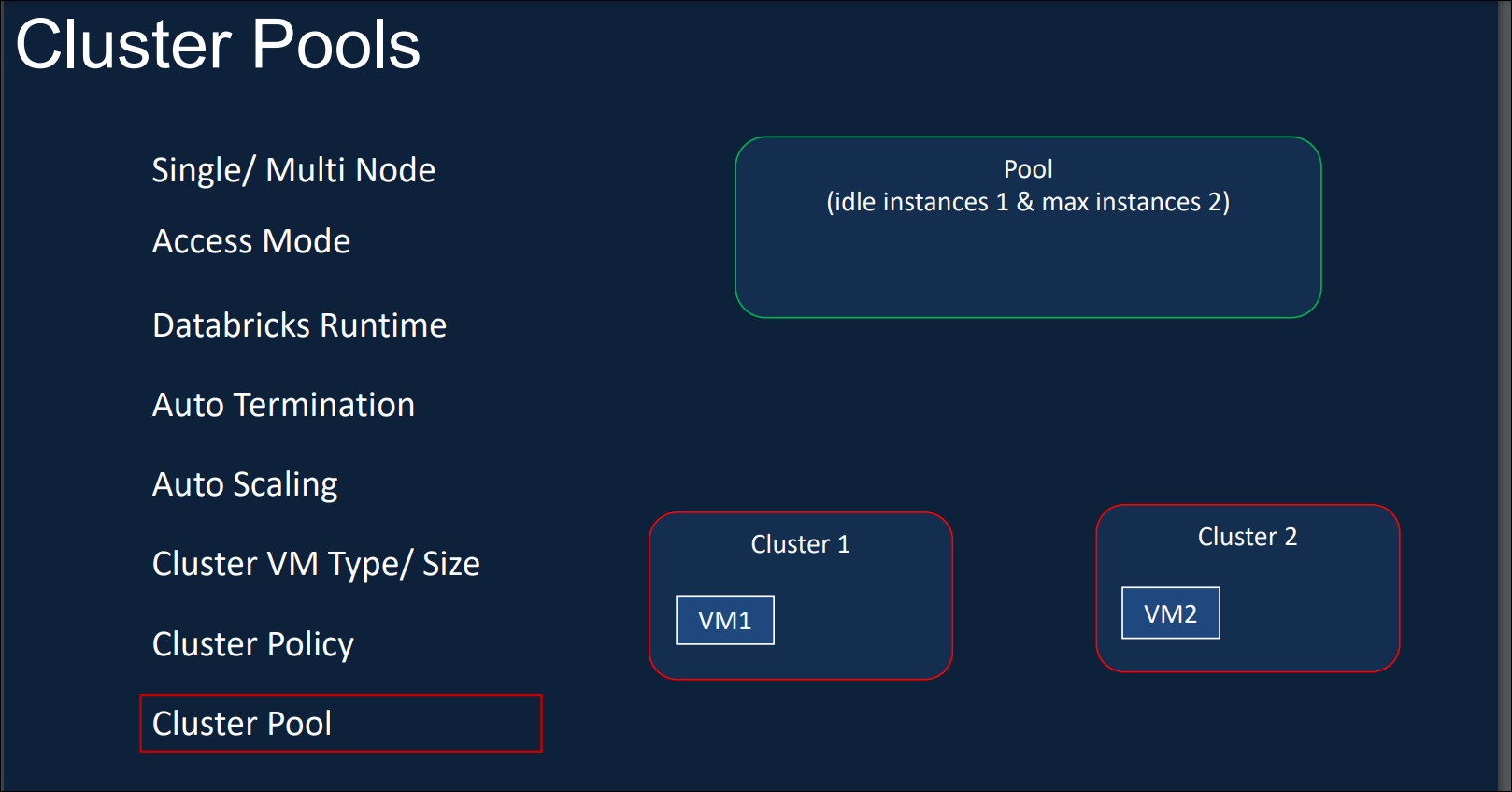
**Cluster Pools:**

In this lesson, we're going to look at Cluster Pools.

Clusters usually take time to start up an Auto Scale. In order to minimize that time, we can use Pools.

A Cluster Pool is basically a set of idle ready to use virtual machines, that allow us to reduce the

Cluster start and Auto Scaling times.



For example,

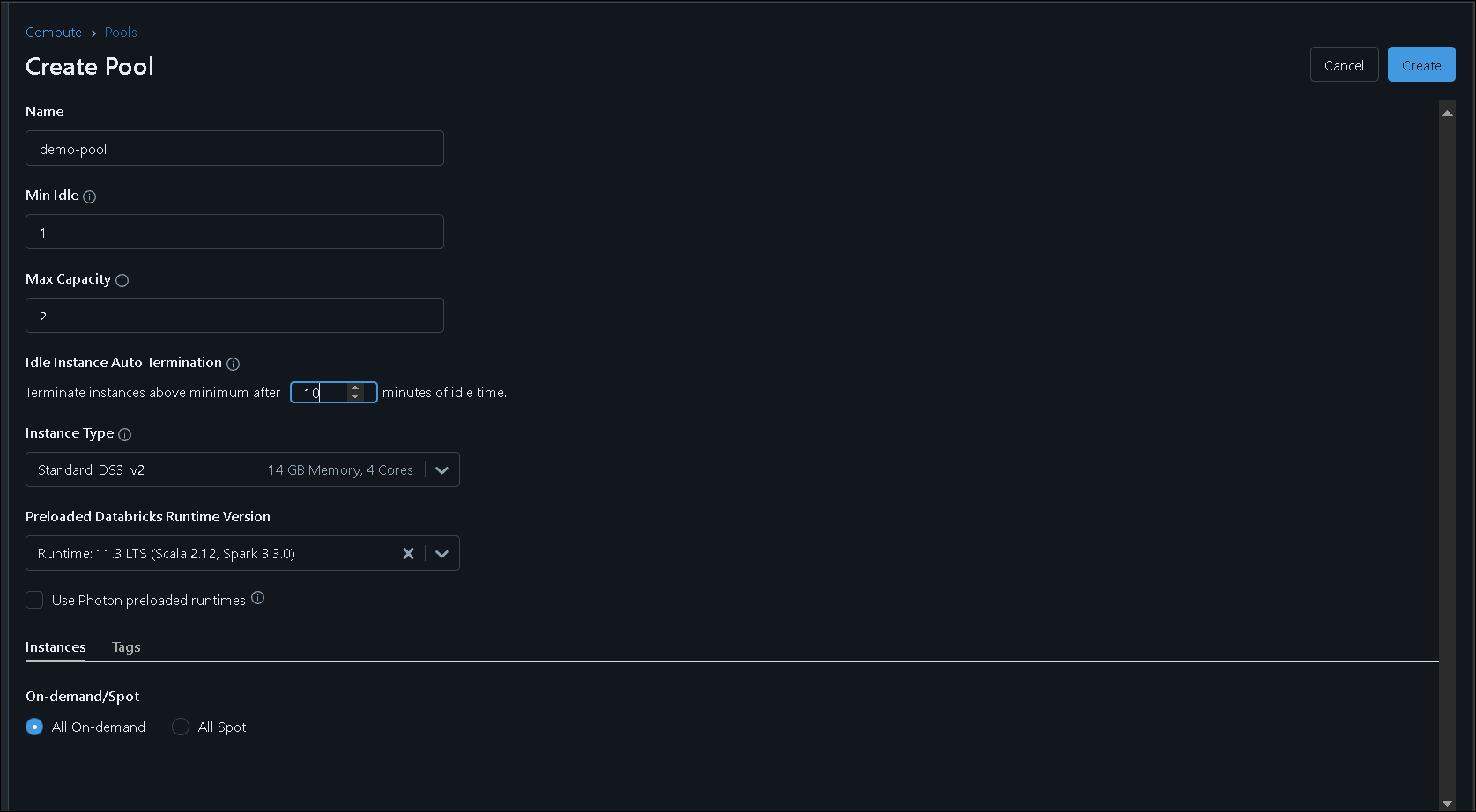
let's assume that we have created a Pool with a minimum of 1 idle instance and 2 maximum instances.

What this means, is the pool will always keep at least one instance ready for Clusters to consume until,

it reaches the maximum limit, which is two in this case.

Let's say Cluster 1 comes along and requests for one node. Pool gives the one instance to the Cluster and spins up another one ready. If Cluster 2 requests for one more node, then it gives that one to the Cluster 2.

But instead, if Cluster 2 requests for two more, it will fail to start with error that there are enough instances available to allocate. Hope that's given you an understanding about Cluster Pools.



I'll now demonstrate creating a Cluster Pool and a Cluster that uses the pool so that we can see the improvement in Cluster start time. Having said that, if you are using free tier or student subscription, you won't have sufficient quota required to create a Pool. But please don't worry, we don't need the Cluster pools for the rest of this course. I'll also be deleting the pools at the end of this lesson because we don't need them anymore.

Let's switch over to Databricks workspace and get started. Here we are in the Databricks workspace.

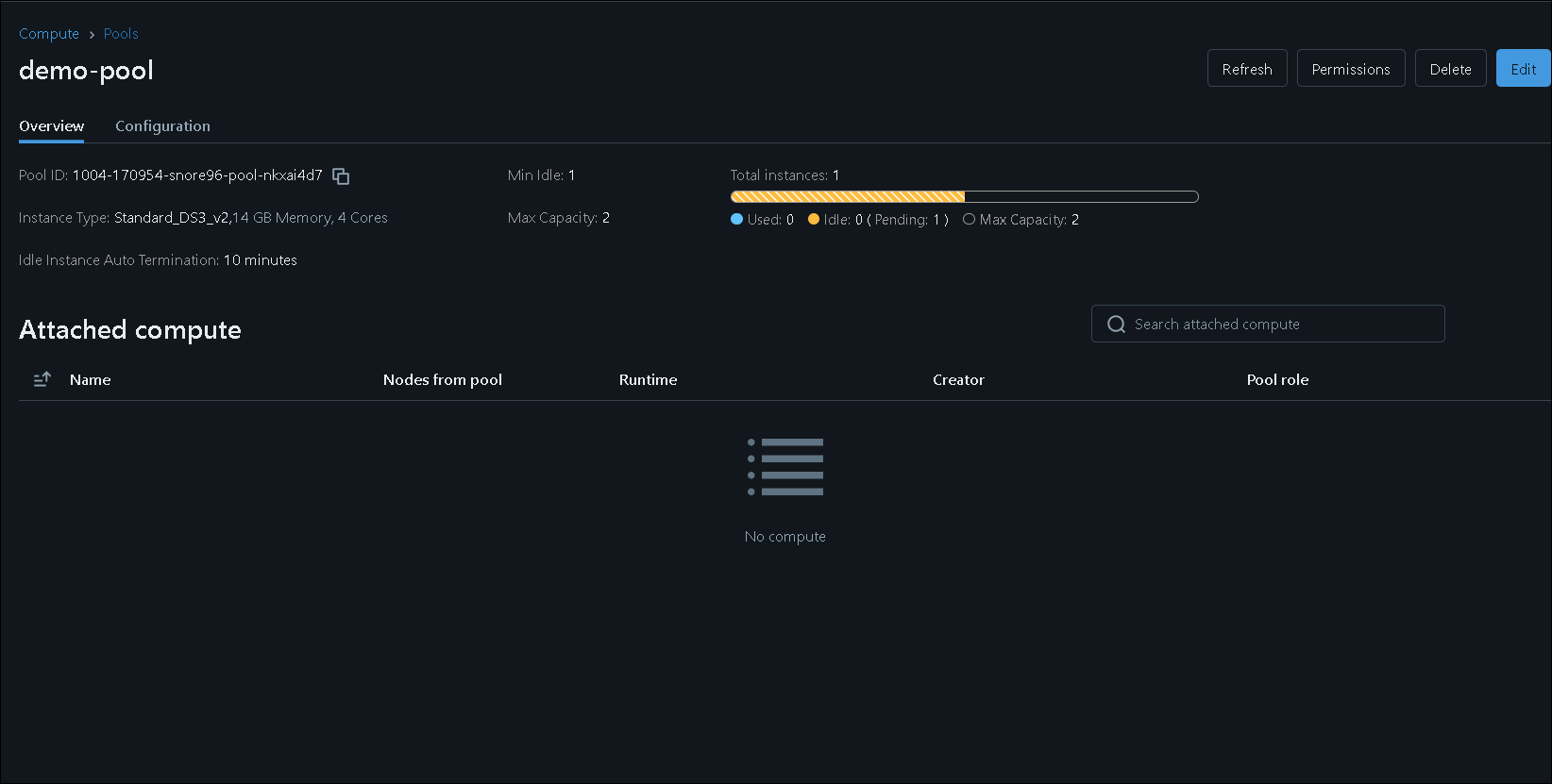
Let's select Compute from the sidebar and select Pools and click Create Pool to create the Cluster Pool.

Let's name our Cluster Pool as demo-pool. And I'm going to specify the minimum idle instances as 1 and max capacity as 2. Let's also specify the Auto Termination time as 10 minutes. Please note that the 10 minutes we are specifying here doesn't apply to the, idle instances here. So that means one instance will always be up and running, and the auto termination only applies to the second instance. Let's select the instance type as DS3\_v2, also preload the Databricks runtime version to the Cluster Pool.

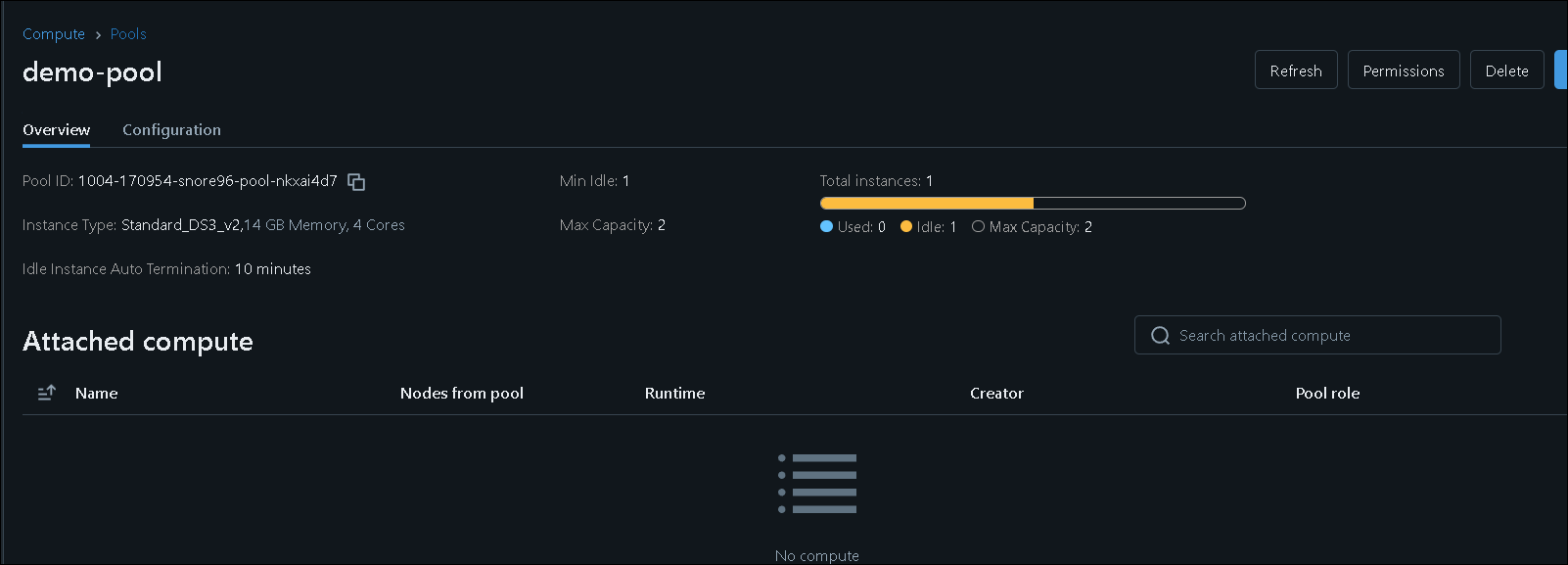
So that's going to be 11.3 LTS. So if I create a Cluster to use this pool, and the cluster is configured to use 11.3 LTS as the runtime, it'll already have an instance ready to be used and it will be quickly provisioned and we can specify whether to use this Spot instances, and also you can specify Tags.

Let's ignore them for now and click Create.

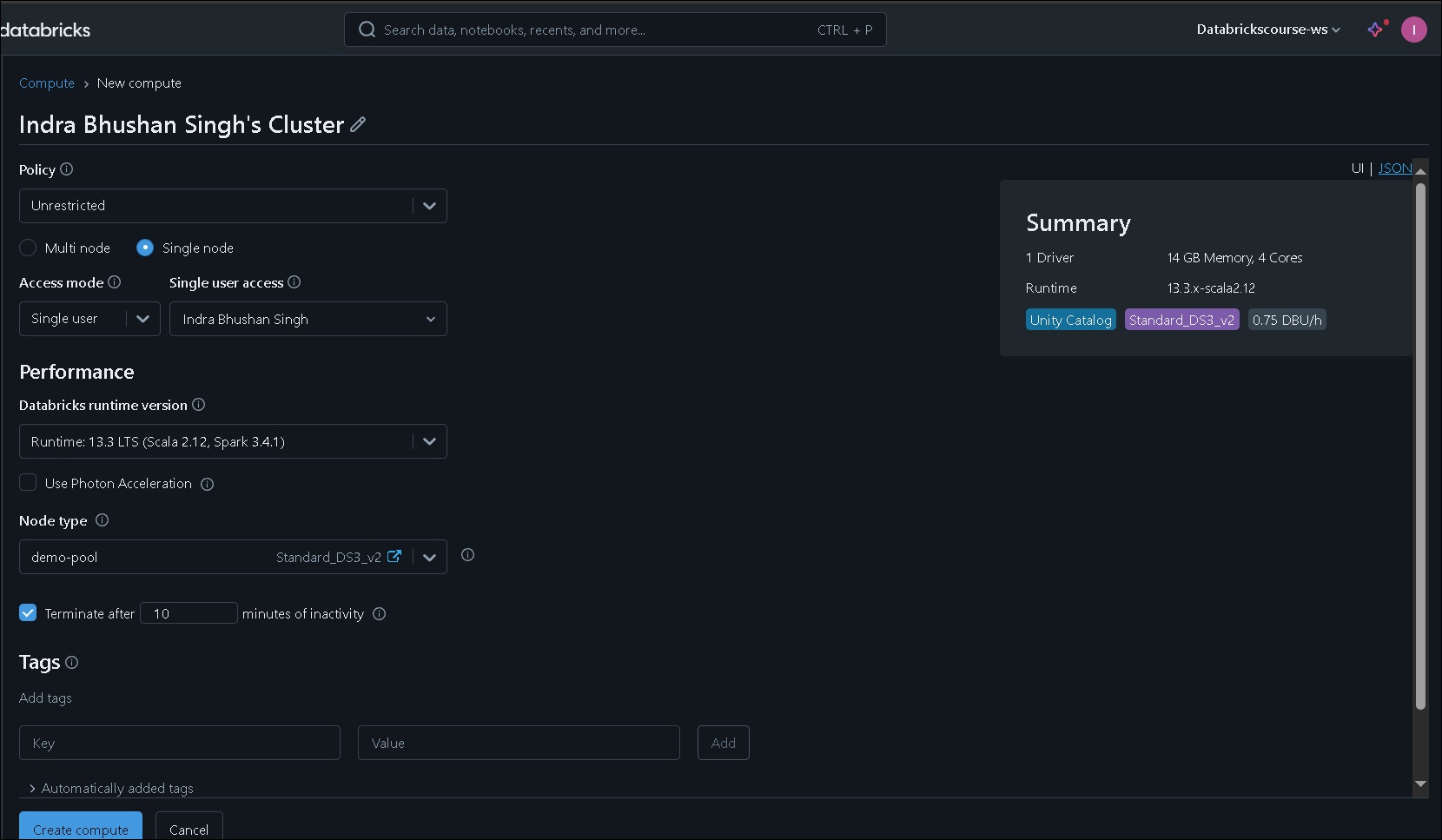
It's going to take a couple of minutes to make the VM available, so let's wait for that.



As you can see, it's trying to provision the first node and it's still in pending, so we want to wait that to be available.

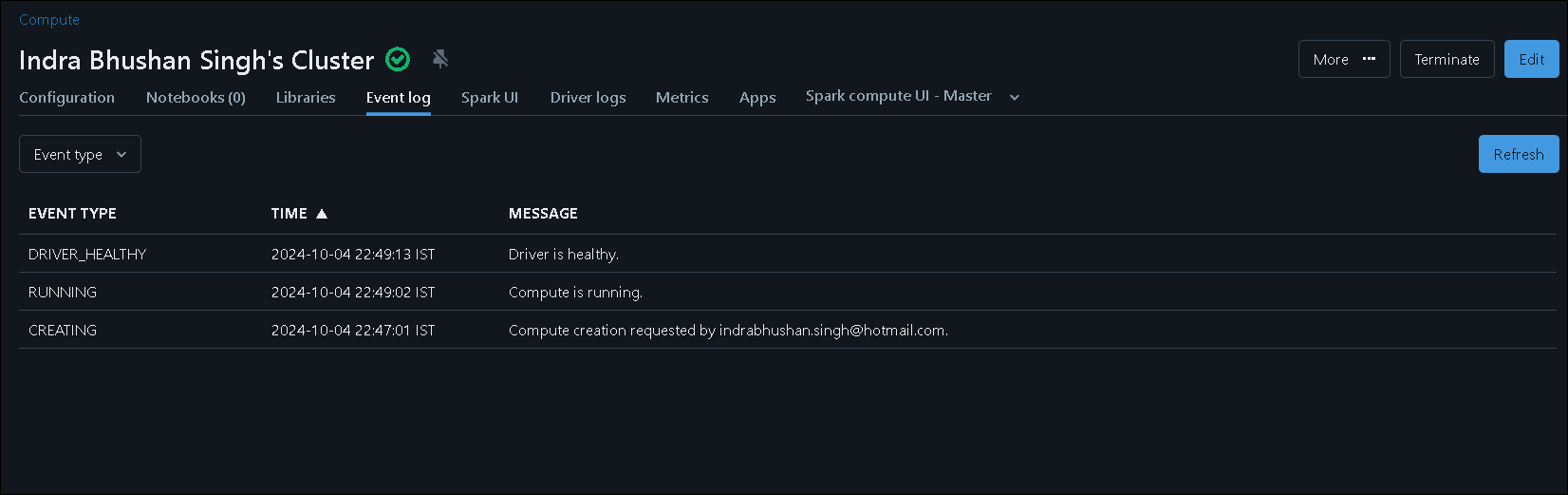


As you can see here now, it's got the one node provisioned and ready to use. Now, let's see how long it takes to create a Cluster that uses the node from this pool.

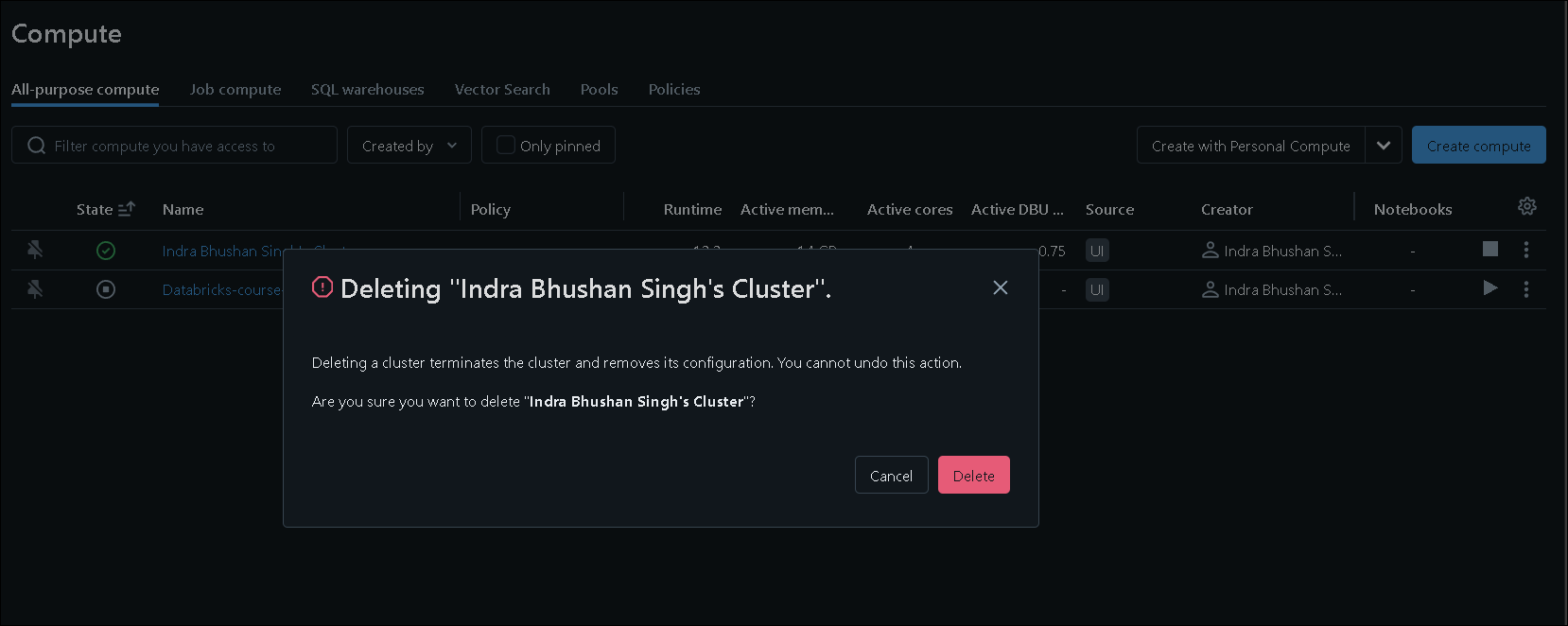


Let's navigate to All Purpose compute and create the Cluster. So, you can click on the Compute on the sidebar again and then come to All Purpose compute and click Create compute. So let's go for a Single node Cluster because we only have one node provisioned there and the only thing you do differently is when you pick your node type, you select the Pool.

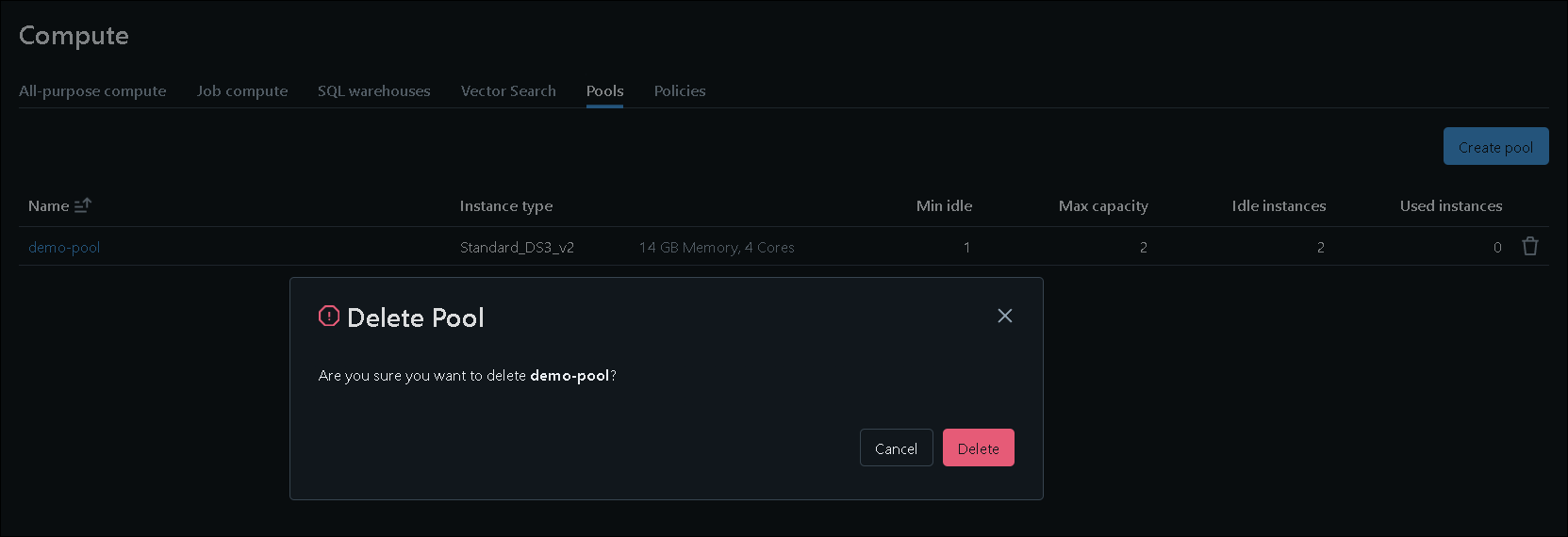
As you can see now, our Pool is being listed as well. So which is our demo pool. So let's select that. And that's got standard DS3\_v2 and that's what we want. And also we wanted the runtime to be 11.3 LTS, that's what our nodes already got. So the creation of the Cluster should be really quick. Let's change the Auto Termination to 10 minutes and click Create and see how long it takes. As you know, it took about 5 minutes previously to provision a Cluster and let's see how long it takes for this Cluster to be provisioned.



As you can see, the Cluster is already provisioned. So let's refresh here and see how long it took. So it only took a minute, as supposed to the 5 minutes for the Cluster that we created earlier. So this is the way you speed up the creation of your Clusters. I hope now you understand the purpose of Cluster Pools and how to create them and configure Clusters to use the pool. Before we finish this lesson, let's delete the Cluster and the Cluster Pools because we don't need them, and also, as I said before, the Cluster Pool is going to always have one node idle. So that means you are going to be charged for one node throughout the period.



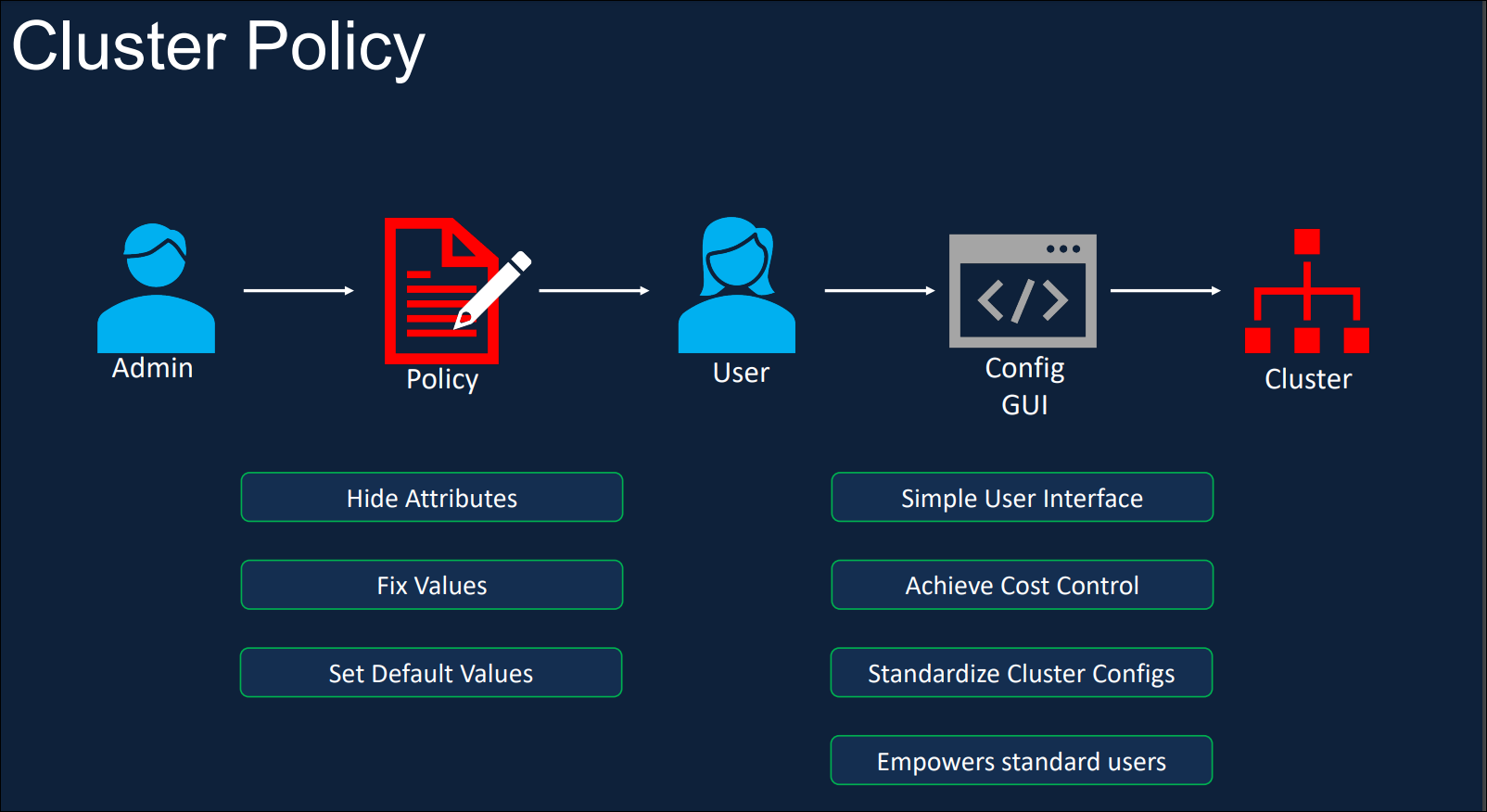
So I'm going to delete this Cluster first. So in order to do that, I'm going back to Compute and I'm going to delete this Cluster, so you click on the three dots here and then Delete.



And also, let's go to Pools. Let's click on the X here and let's click on Delete. So that should delete the pool as well.

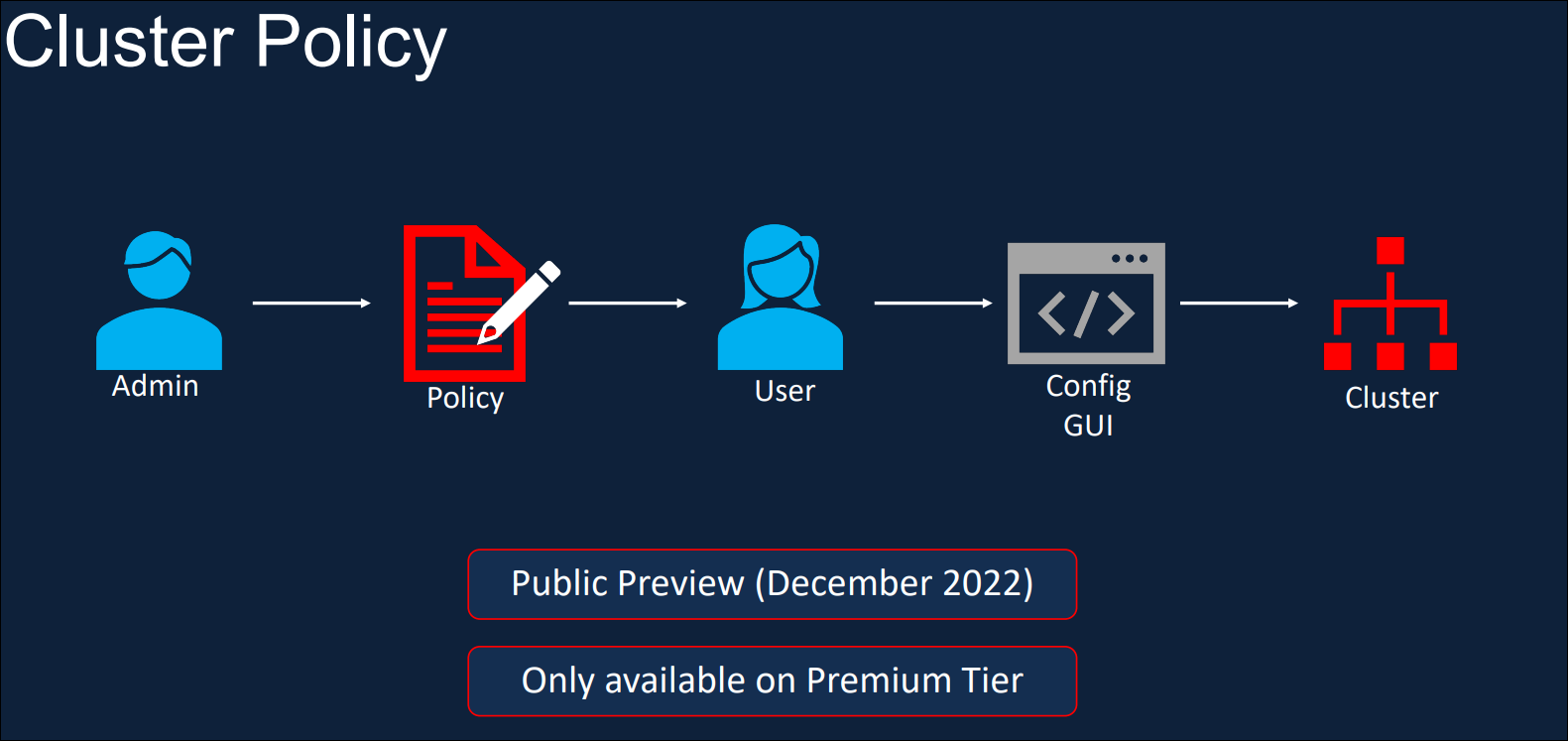
**Cluster Policy:**

Creating the wrong type of Cluster could increase the cost of running the project and also can affect data security.



So if we expect standard users to create the Cluster and keep the cost down, we should simplify the interface and make it easier for them. That's where Cluster Policies come in.

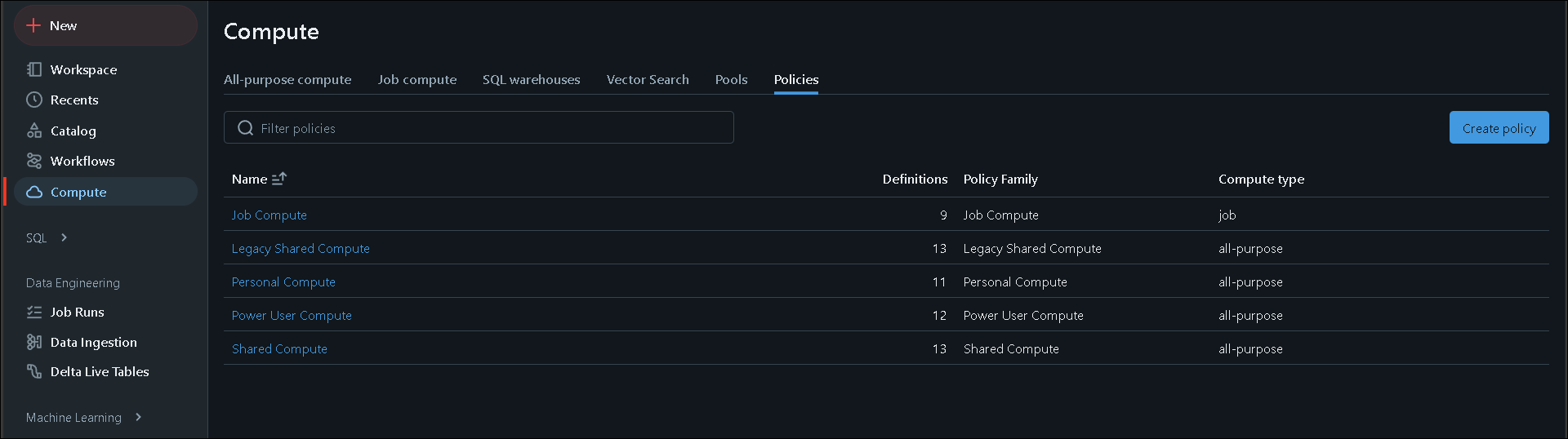
* Policies are basically a set of rules which allows us to hide or take out the configuration options,that are not required on the user interface.
* They allow us to fix the values of some of the configuration parameters and restrict access for the users so that they're not changing them.
* Also, in a set of values, a policy allows us to select default values on the user interface.
* An administrator can create a policy and assign that to a set of users. When the user chooses the policy, they'll now have a Simpler User Interface because some of the options are now hidden or pre-populated with fixed values. Only allowing the users to select certain type of nodes or setting Auto Termination
* by Default, allows us to keep the cost down.
* It helps us having standard type of Clusters created by all users in a specific group.
* And finally, cluster policies take away the need for an administrator to be involved in every clustercreation decision and empowers standard users to create them.

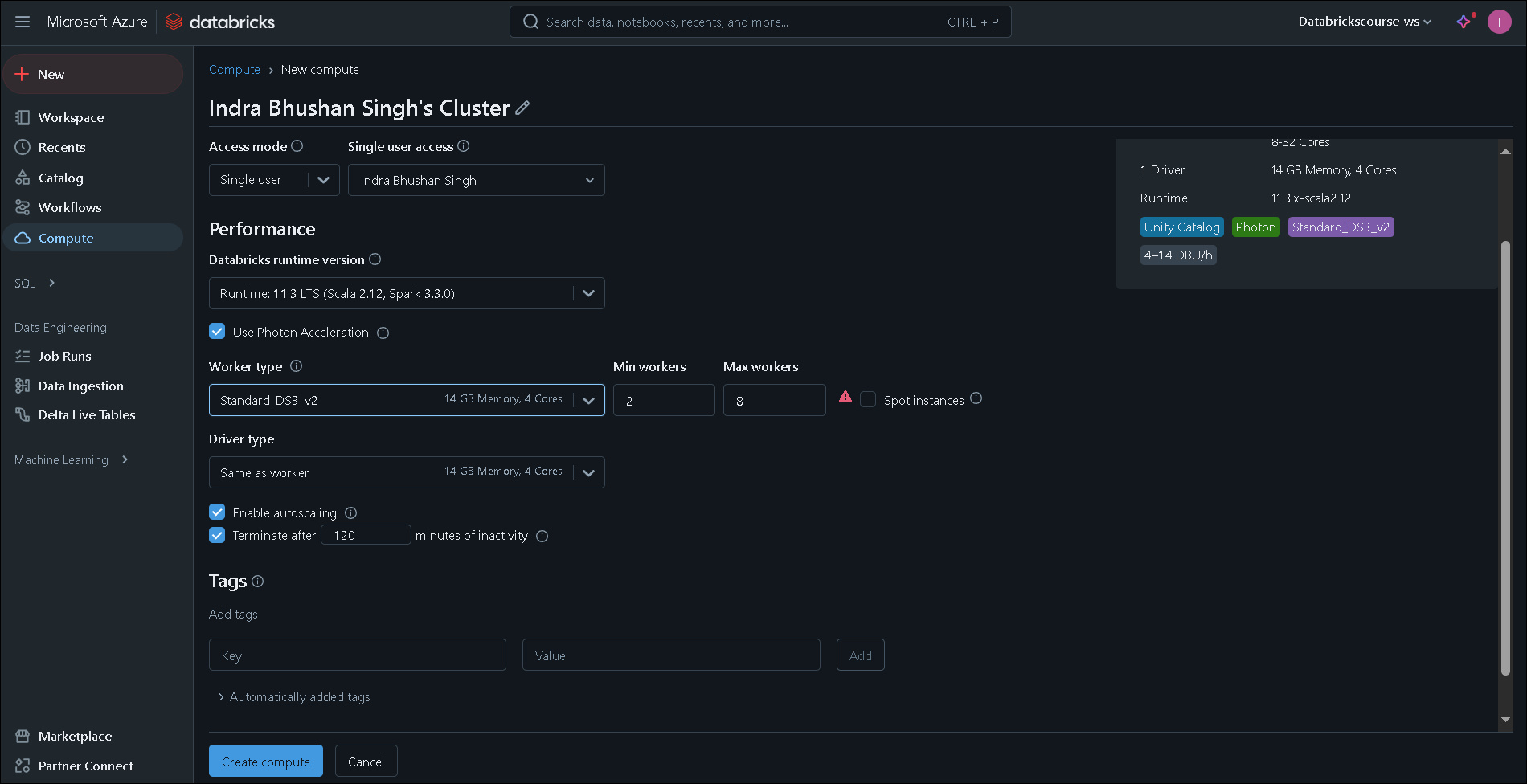


It's a great feature, but please note that it is in public preview, at the time of recording in December 2022.So it's lacking features and being developed, so please expect to see some fixes and improvements over time.

Also, most importantly, cluster policies are only available for workspaces created in premium tier.

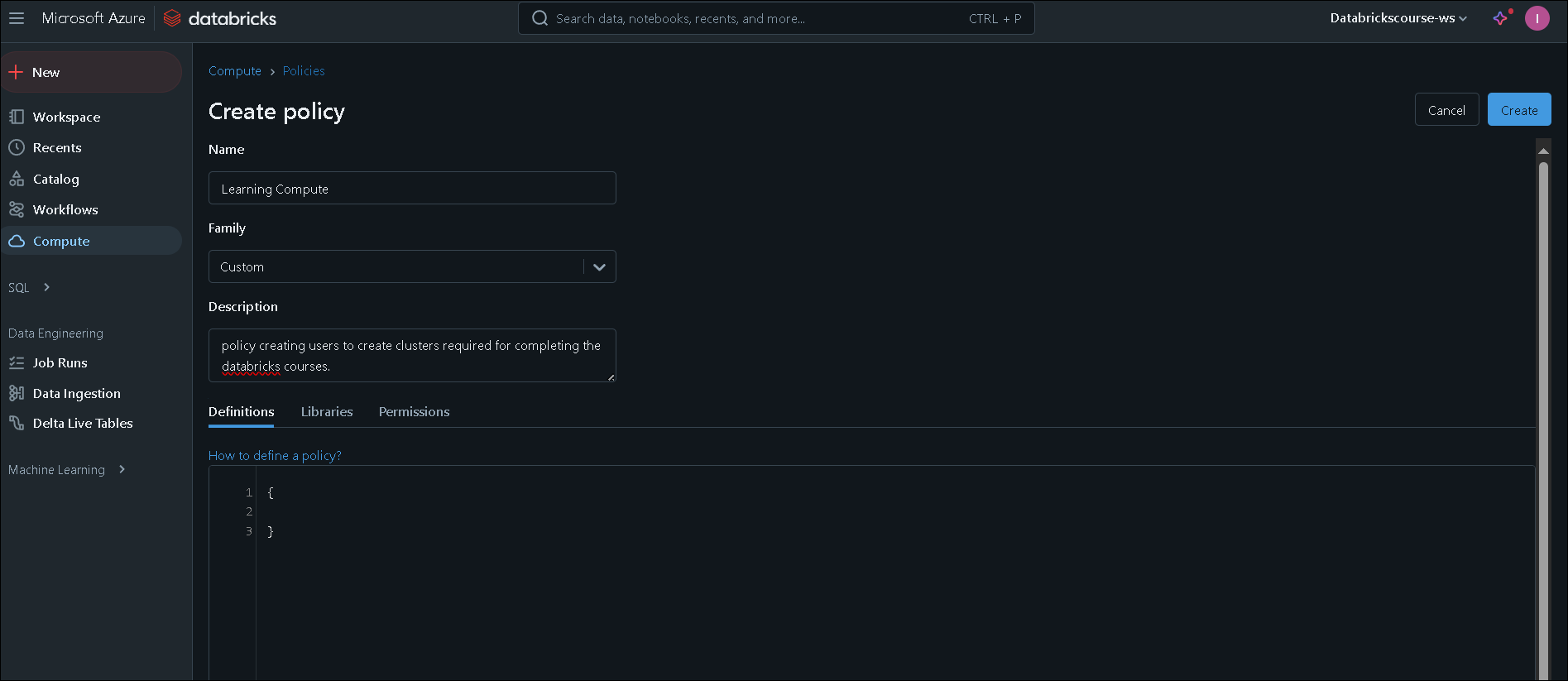
If you have a workspace created in standard tier in order to save some cost on the compute resources, please don't worry. You can simply create another workspace and try this lesson alone which will cost you nothing, as we are not going to create any compute resources as part of this lesson. You can then simply delete the workspace once you're done with this lesson, or else you can simply watch the video and carry on with the rest of the course as we don't need cluster policies for the rest of the course at all.

  
Let's now switch over to the Databricks workspace and see this in action to strengthen our understanding.Here we are in the Databricks workspace. Let's navigate to Compute and select Policies. As you can see, currently there are four Cluster policies available. They are default policies created by Databricks for us. If one of them suits our needs, we can simply use that. Also, we have the option to create another Policy, based on one of these policies available and override certain attributes. So that makes it easier to create a Policy. But in this lesson, we are going to look at creating a Cluster Policy from scratch.



Let's define our requirement first. Let me navigate to Compute and click Compute. When we select the Policy as Unrestricted, we are presented with the user interface with all the options available to create a Cluster. Let's say, I'm an administrator and I have a number of students taking the course that you are taking, and I want to restrict them to create Clusters, limiting to only the needs of this project. That means we only allow for a Single node cluster to be created. That means, we'll pre-select Single node and we don't let them choose Multi node at all. We can leave the Access mode as it is and they can choose what they want. And let's say we want them to select the latest runtime within the Standard group, not the ML group.

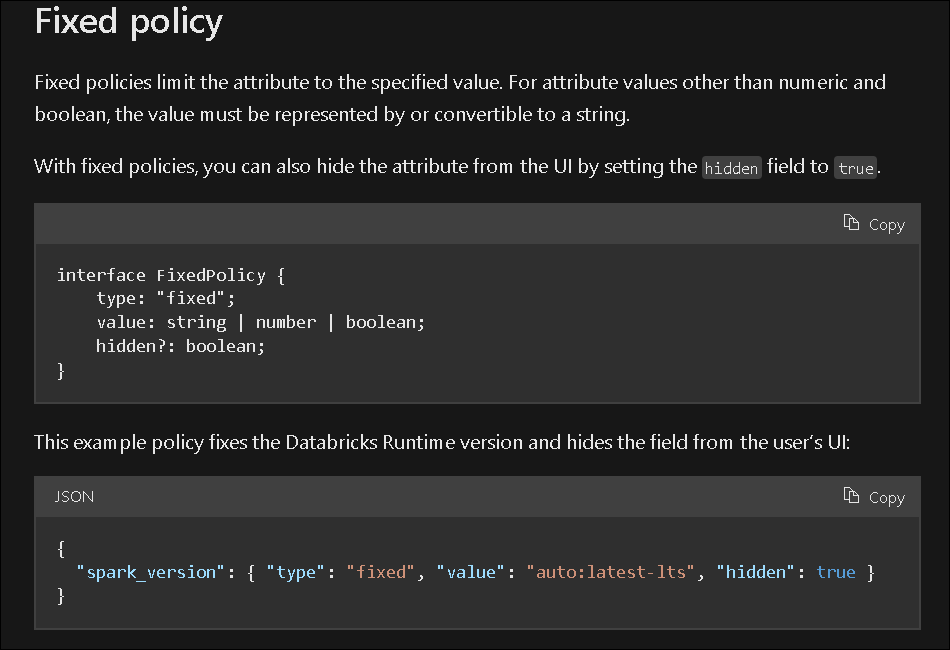
And we want to restrict the node type to be only the smaller one. So let's say we want them to be able to select only, Standard DS3\_v2, Standard D3\_v2 and let's say Standard D12\_v2 and DS12\_v2. And they're all requiring only for course. Finally, let's say we want to keep the Auto Termination to only 20 minutes and we don't want them to deselect the Auto Termination. So that means it should always be ticked with 20 minutes populated there and we don't want to give them the ability to change that at all. And that is our requirement. And that will vastly simplify this user interface and the users will have to simply come in, type in the Cluster name and click Create. And let's see how to do that. I'm going to simply duplicate this tab and work on that one so that we can switch between the two and test it while we're doing it.



So I'm going to go back into cluster policies again and let's click Create Cluster Policy. And as it's asking, we need to provide a name, I'm going to call mine as Learning Compute. And as I said previously, we can select a Family, we can choose one of the pre-existing cluster policies here and override certain parameters. We don't want to do that at this point in time. I'm going to go for Custom and create a brand new one. We want to give some description. Now we can start right in the definitions. Definitions are going to be in JSON format. I'll show you that in a minute. But once you've written the definitions, you can assign that to certain users or groups. And in order to do that, you come over to permissions and then you select the user or the group or the service principle, and then you give the permission. At this point in time, the only permission available is Can Use, and then you click Add. So let's go over to Definitions and start looking at how to write the definition.So there is a nice guide here. If you click on this link,

how to define a policy that opens up the documentation for that. [Compute policy reference - Azure Databricks | Microsoft Learn](https://learn.microsoft.com/en-us/azure/databricks/admin/clusters/policy-definition) Reading this document

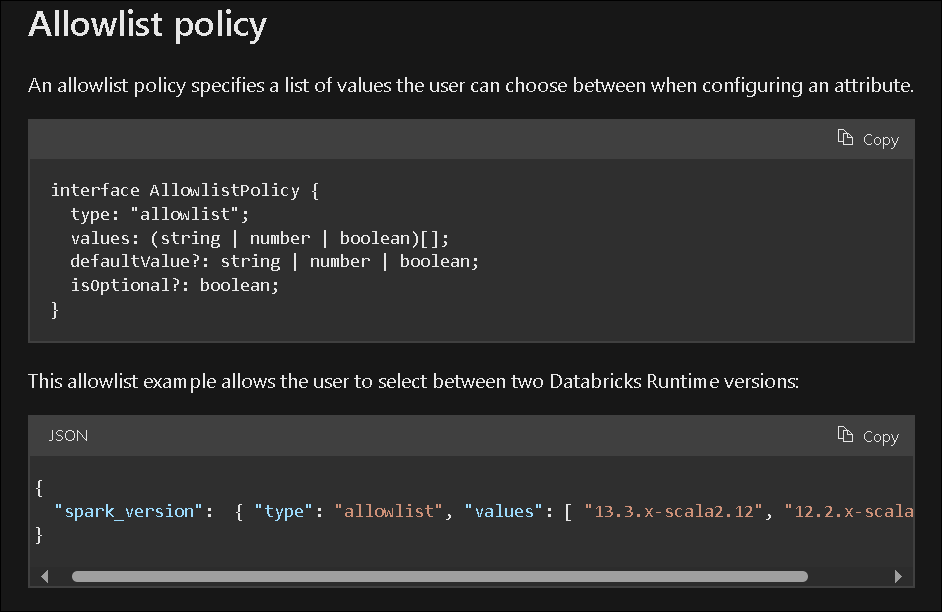
might not make a lot of sense to you at first, but seeing an example helps you better. So let me take you to an example.



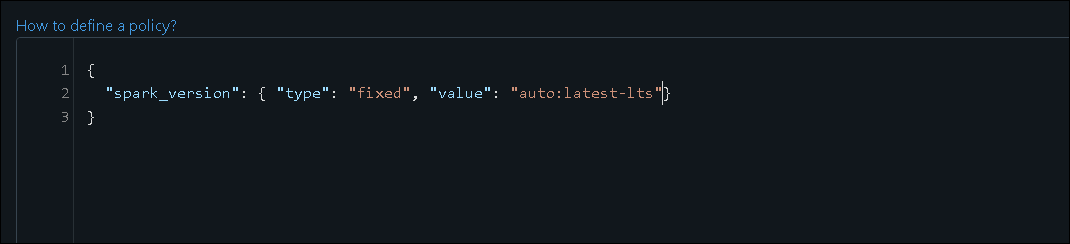
As you can see, we've got a number of policies and Fixed policy is one of them.So I'm going to take you to the example that belongs to the Fixed policy. Let's say we want to restrict our Clusters to have the Spark version to be a certain value. In order to do that, you would use this JSON object. So, as you can see, spark\_version is the attribute name. And within that, we're defining the policy with the type as fixed and we're giving the value as, 7.3scala2.12. And also, we are saying hidden equals true. So, what's going to happen is when you come to use this policy, the user won't be presented with this attribute at all actually,

they won't have to pick the runtime version because we've set the hidden as true. But if we didn't set that, what would happen is they'll see the value of the runtime version as 7.3, and they wouldn't see any other runtimes. So that's how you limit what the user sees on the user interface. Similarly, you can set other policies as well actually.

So for example, if you want to set some list of values, it's the same thing we're looking at here.



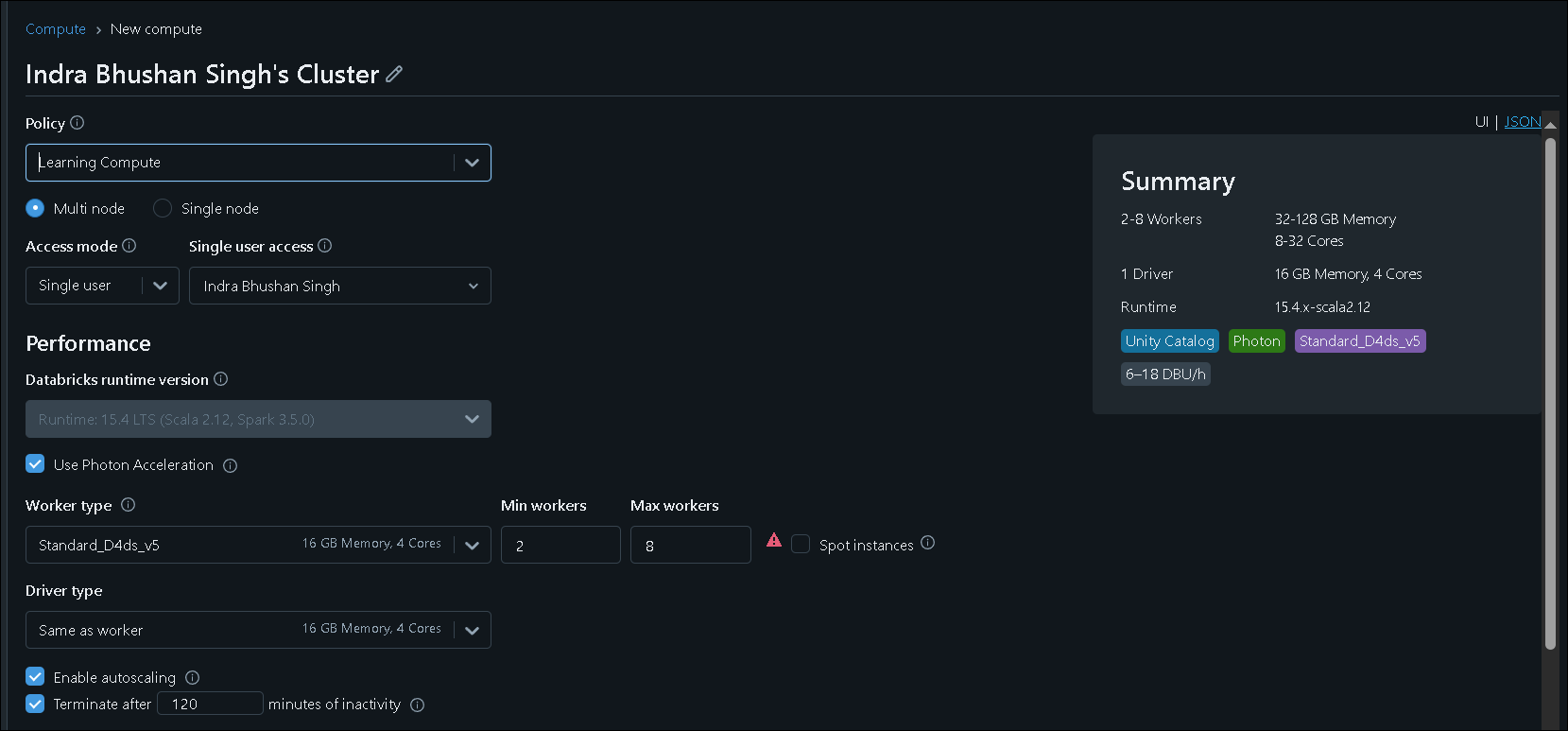
You would go for the type as allowlist and you set the values as 13.3 and 12.2 in this example. So they'll see both of them. You can also set the default value, so if you say, my default value is 12.2, they'll see both of them in the dropdown, but 12.2 will be the one which is a pre-selected for them. So that's how it works.



But let's put it into practice so that you can see in action. So I'm going to go back to my Cluster Policy and write that JSON object, which we saw just now. And as you know, the attribute we're interested in, is spark version. And let's set the attribute values now. And as we said, actually, the type is going to be fixed.

So let's do that. And in our case, we didn't want to have the version 12.2, we want to go for the latest LTS version available. So let's set the value to be the latest version. So either you can hard code it or you can use, Databricks to select it for you, which will be auto: and then you go for latest LTS. Then, Databricks knows I need to pick the latest LTS version, so that's what you would do. So now what will happen is when you use this policy, that's going to present you with only the latest LTS version, there will be nothing else available. We can also set the attribute to be hidden here, but I'm not going to do that. I'm going to show what comes up on the screen and then we can come back and change this again.

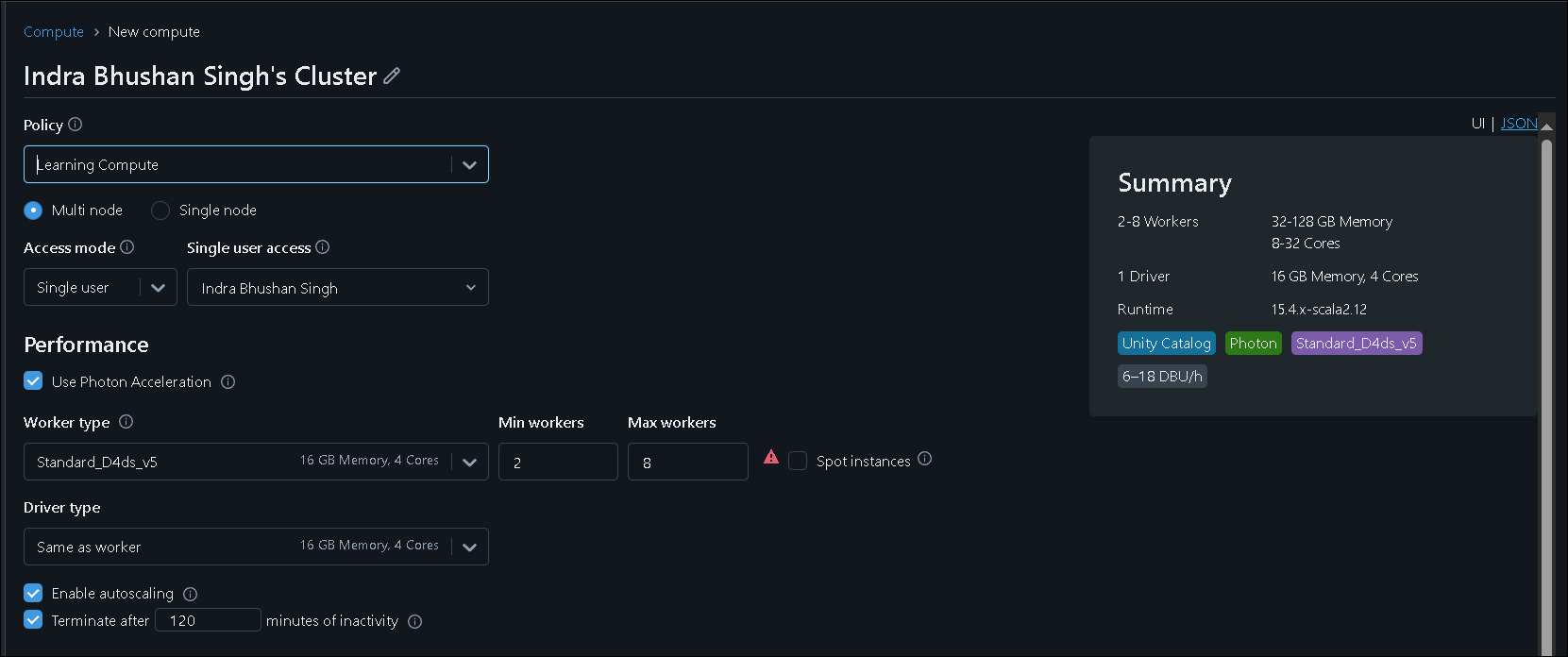
So let's click Create. And the Cluster Policy is now created.



So let's go back to this tab here, and I'm going to do a refresh here so that the policy takes effect and I'm going to select my policy, which is Learning Compute. So let's click Confirm. And as you can see here, the Databricks runtime version has been selected as 11.3 LTS, and that's been grayed out. So you can't change that as an user. So now that's one less problem for the user to think about, but they don't even need to see that on the screen.

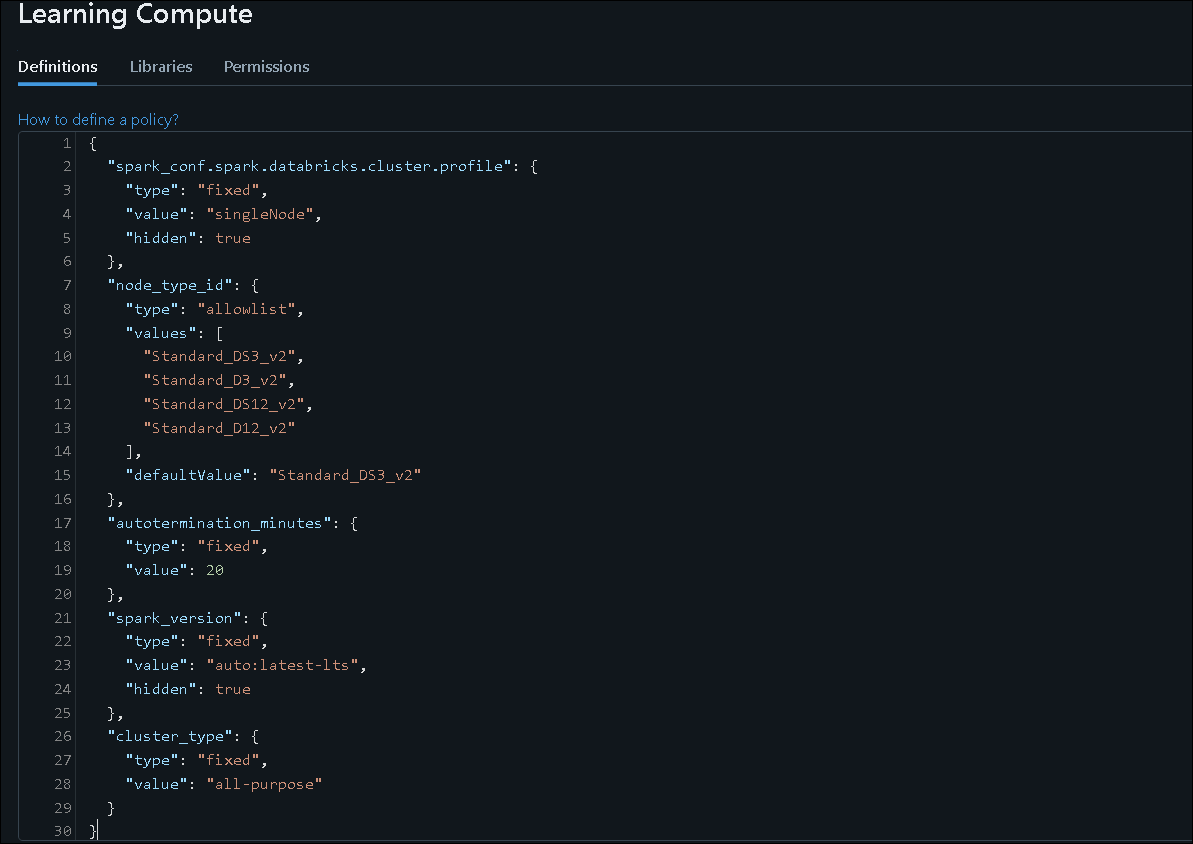


So in order to take that out, you just edit the Cluster Policy and set the attribute hidden to true. So let's do that.



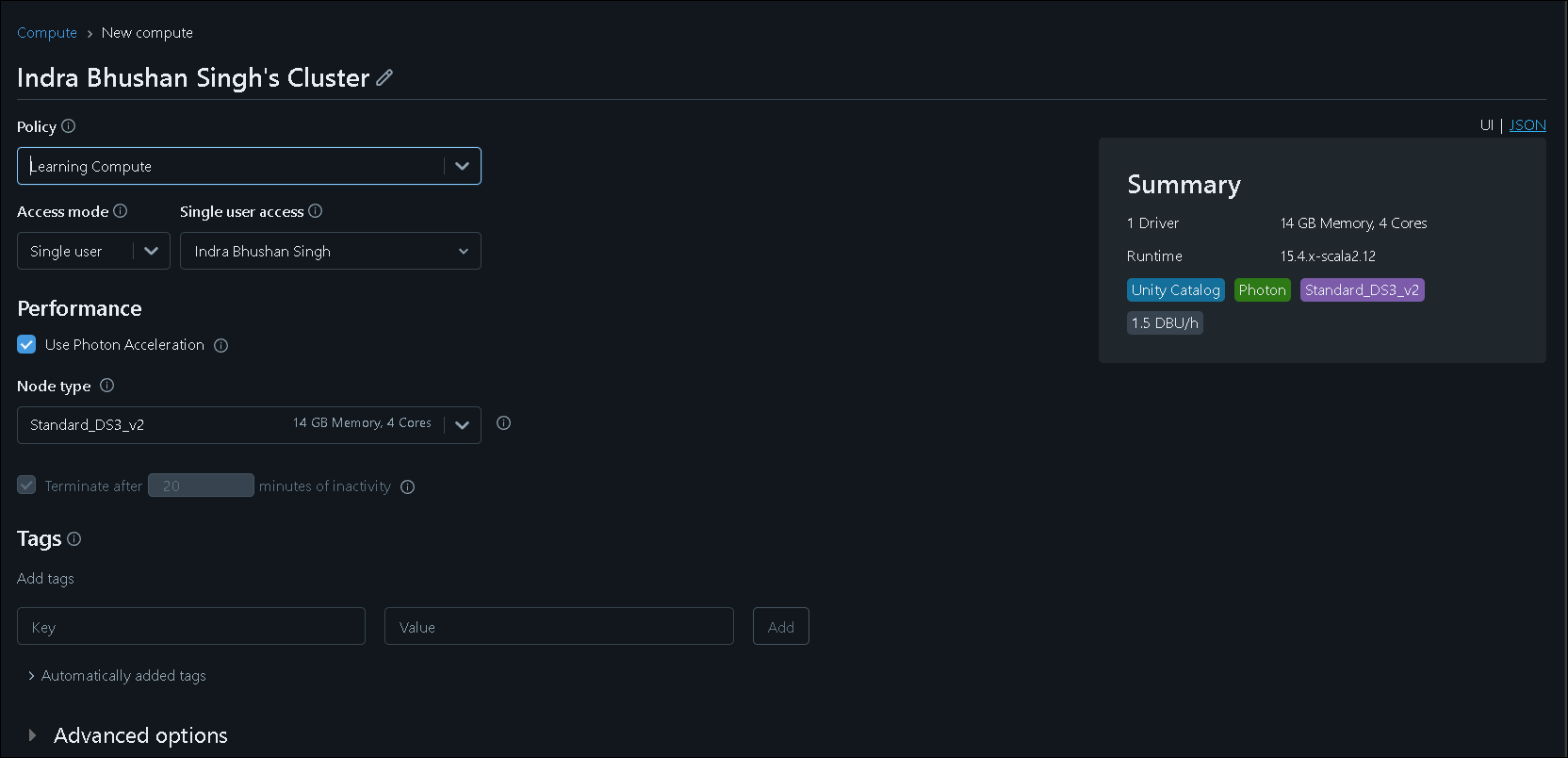
So again, I'm going to Update and let me come back and refresh. And this time when I select that policy, the runtime version itself should disappear from here. That's brilliant.

Now that you know how to create the policy and add definitions, let's run through the other requirements.



Just to speed up the lesson.I pre-prepared the JSON. Let me add that and walk you through. So I'm going to go over to Cluster Policies and click on Edit and I'm going to paste what I have. And as you can see, we've got a number of definitions now, not only the one we had previously, which is the Spark version, I've added others as well. So let me walk you through from the top. The first one here is the cluster profile. So we wanted to have the Single node cluster. So I've set the type as fixed and the value as Single node. So it's only going to have Single node clusters which are allowed on the policy and it's also going to be hidden so the user won't see it. And in terms of node type, we selected four different node types with four cores.

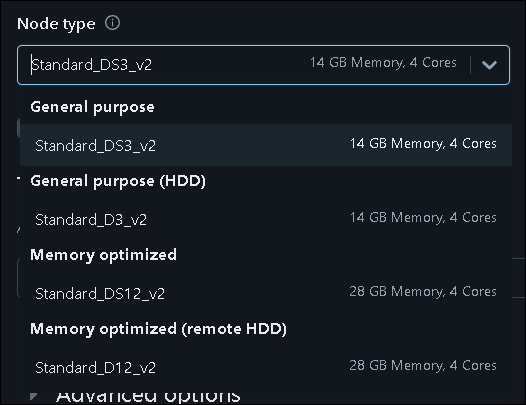
And also I've set the default value to be standard DS3\_v2. So when you look at the list, it's going to have these four and that's what the type allowlist does. And the default value is going to be the top one here. And the next one is going to limit the Auto Termination, minutes to 20. But this time we're not hiding it, I could hide it, but I just wanted to show it on the screen so I've set the hidden to be false or I didn't set the hidden here. And the Spark version we've already talked about. And also the last one is Cluster type. We want our Cluster Policy to be applied to only the Clusters, which are All Purpose Clusters not Job Clusters.So that's why, I've set the Cluster type to be All Purpose as well. So let's click Update.So that's now been created.



I'm going to go over to the other tab and I'm going to refresh the screen here and let's select Learning Compute and click Confirm. And as you can see, we've got a lot simpler user interface here and the policy is Learning Compute and access mode,

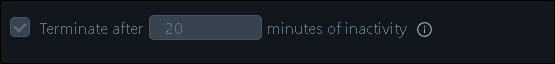


as we said, we're not limiting to anything, so the user can select what they want to select.



And the node type here is only the four node types that we've selected are given in the list and the

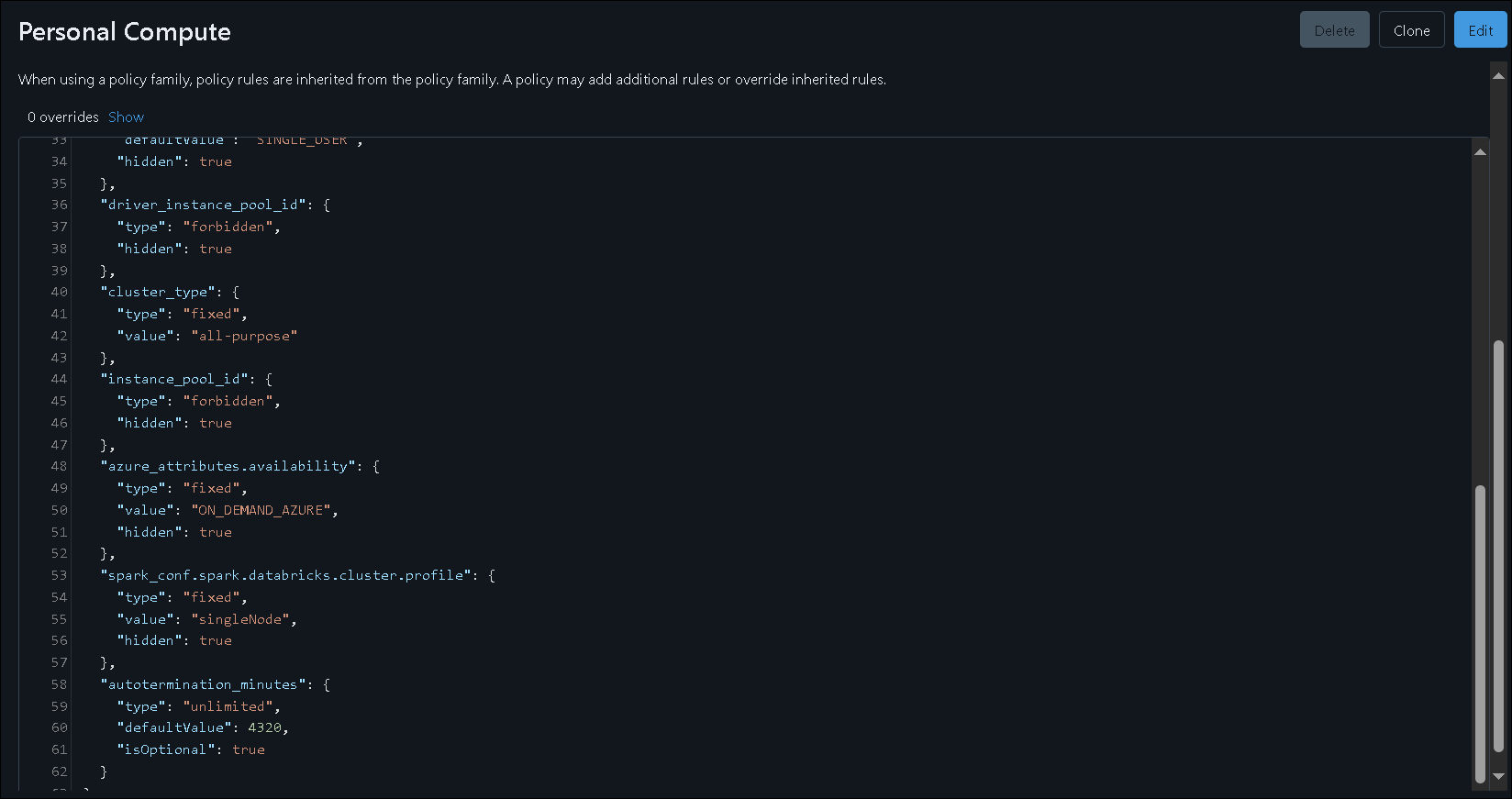
default is Standard DS3\_v2 which is brilliant,

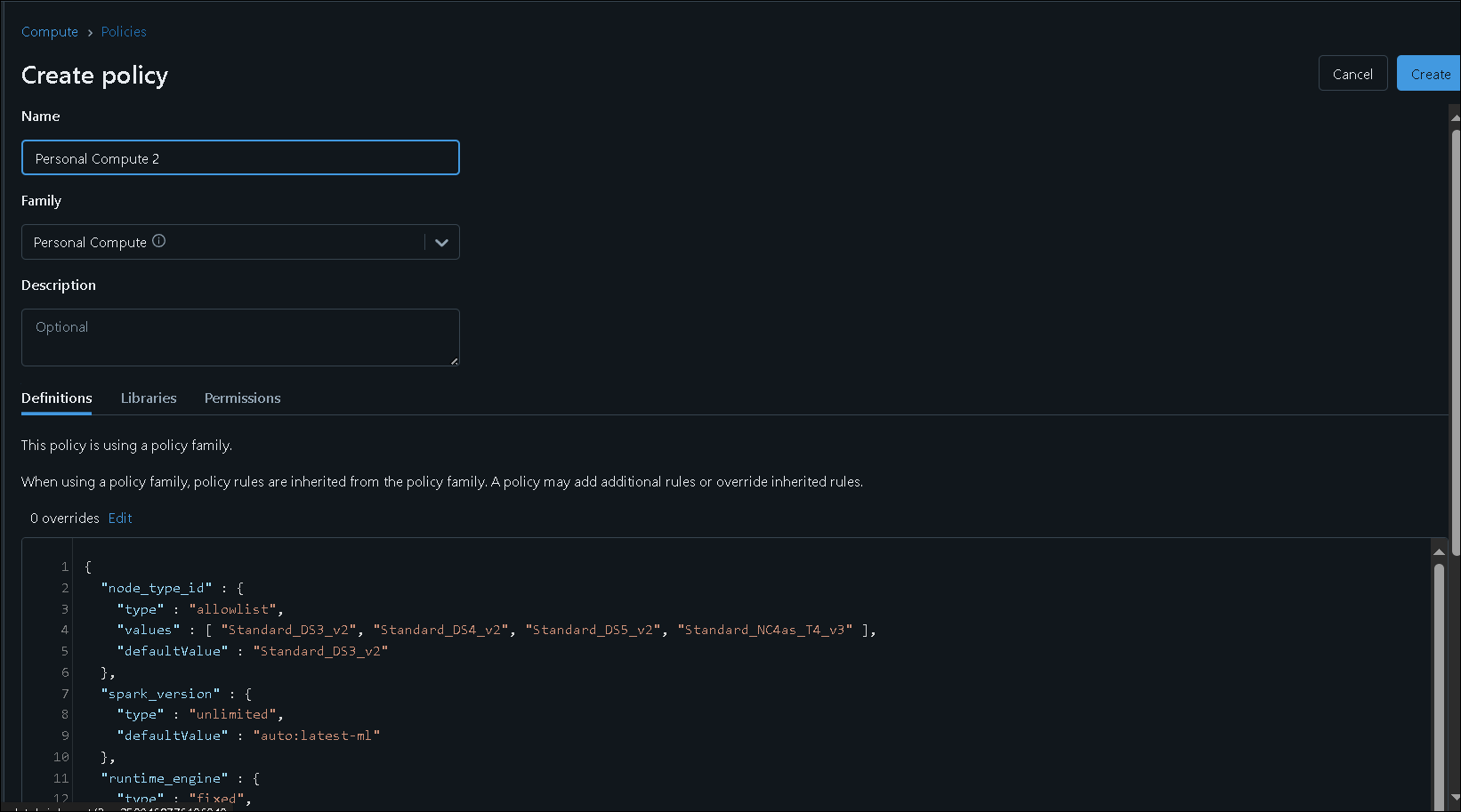


and now it's going to terminate after 20 minutes of inactivity. And as a user, I can't change it, as you can see, I'm not allowed to change it or I'm not allowed to untick this box as well. So whoever is creating the Cluster is going to have the Auto Termination set to 20 minutes and there are no going to be any surprises in terms of cost.

As you can see, Cluster Policies are really powerful. They help us keep the user interface simple as well as keep the cost down. Before we finish this lesson, I'm going to quickly show you how to use one of the existing cluster policy templates or families and just override one of the attributes,

instead of creating a whole new policy like we did.

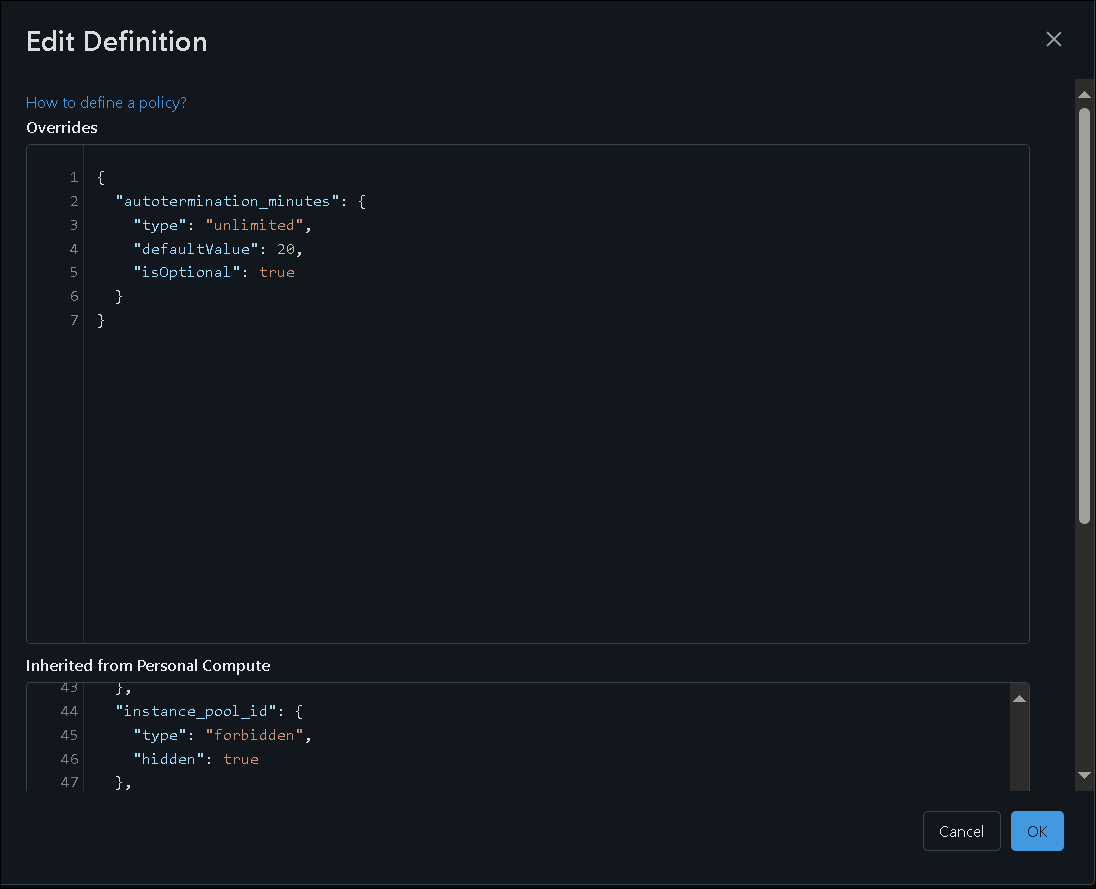
 Let's say I'm happy with everything on a Personal Compute for my purposes, but I don't like one of these attributes. So for example, in this policy, I don't like the fact that the default value of Auto Termination is set to 4320. I would like that to be 20 minutes. So I want to change that, but I want to use everything else as it is.



So what I can simply do is go to cluster policies, create a Cluster Policy, let's call this one as Personal Compute 2.And I'm going to select the Policy. I'm going to select the Personal Compute Policy.

As you can see, it's populated everything from the personal compute policy into my policy here now.

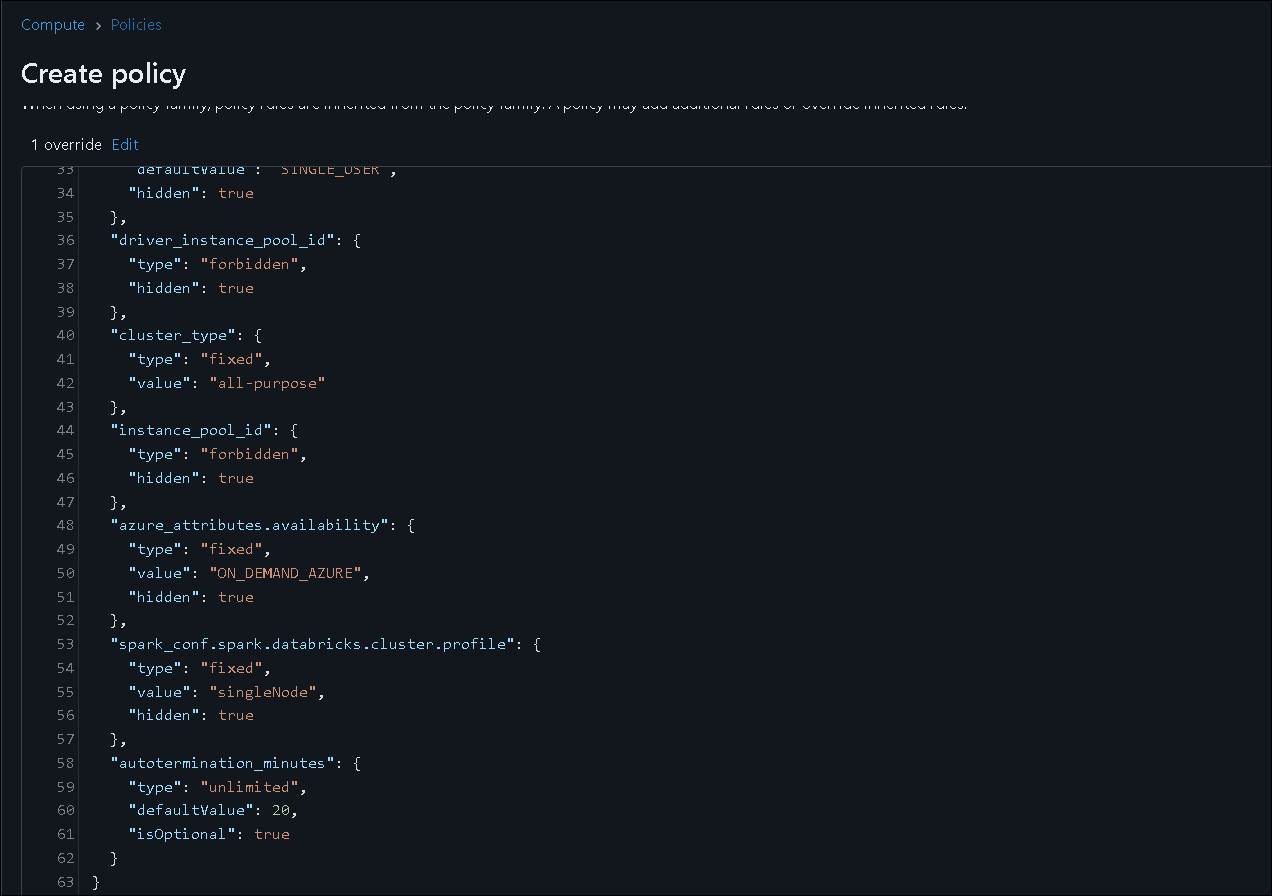
But I want to override the value of Auto Termination.



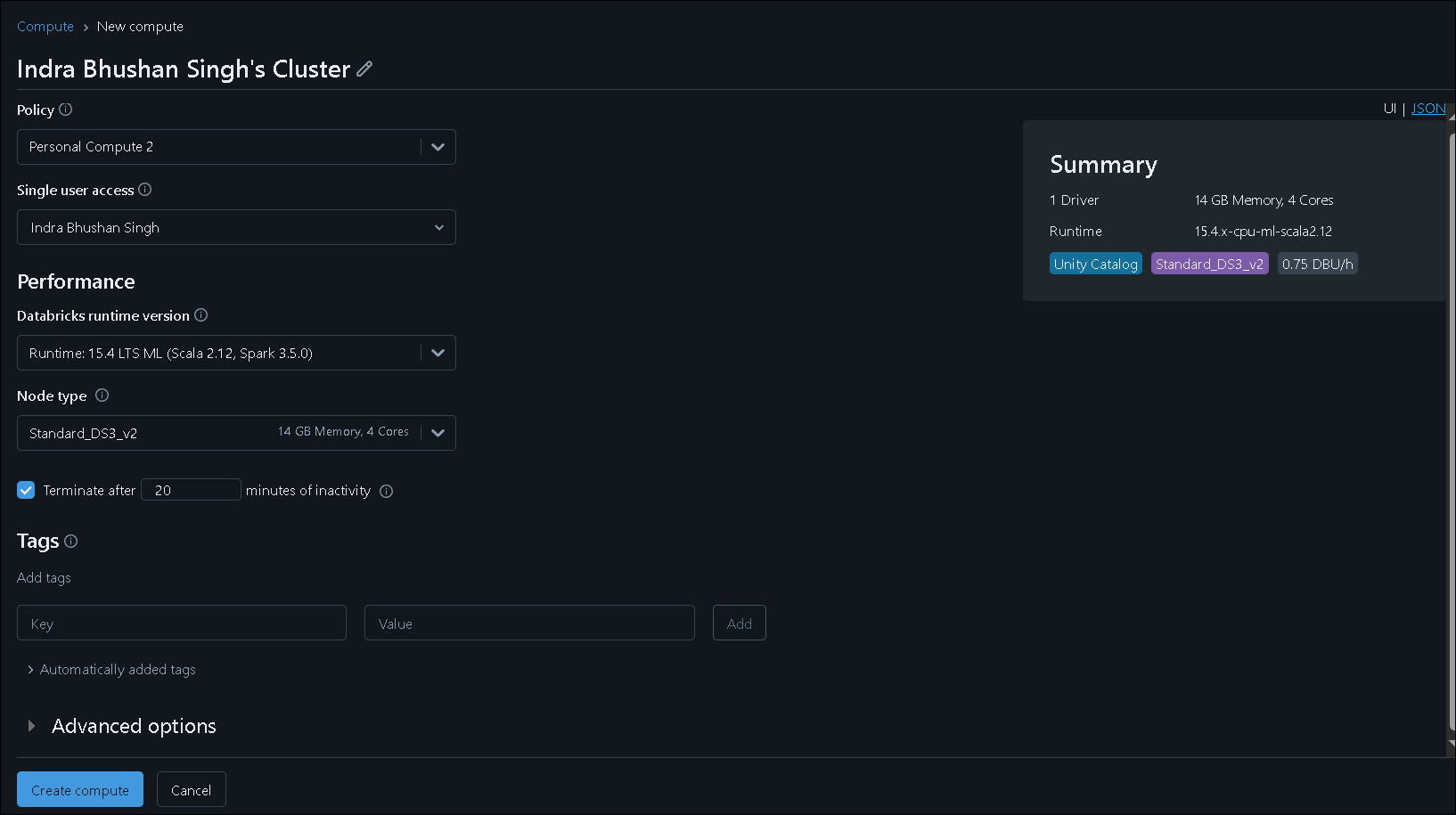
In order to do that, you click on Edit here and then you put your values or you put your definition here.

I'm going to simply copy the definition, which is already there. And let's change the value to 20 minutes.

So that's what I'm doing and let's clickOk.



Now we have a policy. If you scroll down, you'll see that the Auto Termination is set to 20 and you can see override here.



If I try to use it, I'll get everything which was on the Personal Compute, but I will also get the Auto Termination set to 20 minutes rather than 4320. And that's what I'm seeing here. So that's another option for you. So if you want to use one of the existing families and override only certain values, you don't have to create a whole new policy. Just adopt it and override the values you want to override.