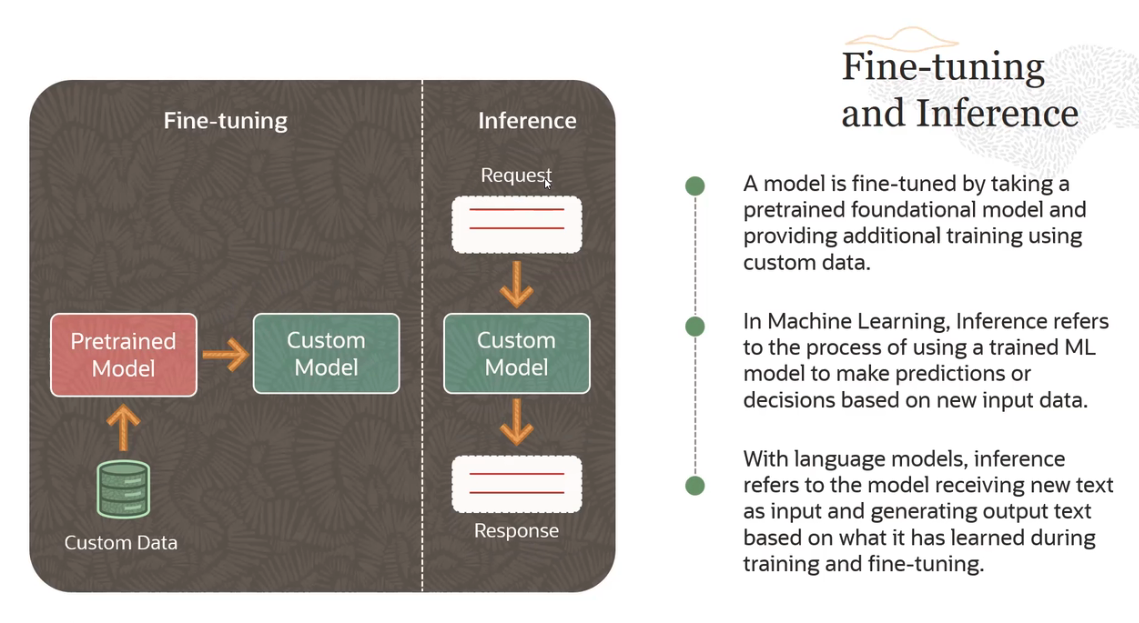
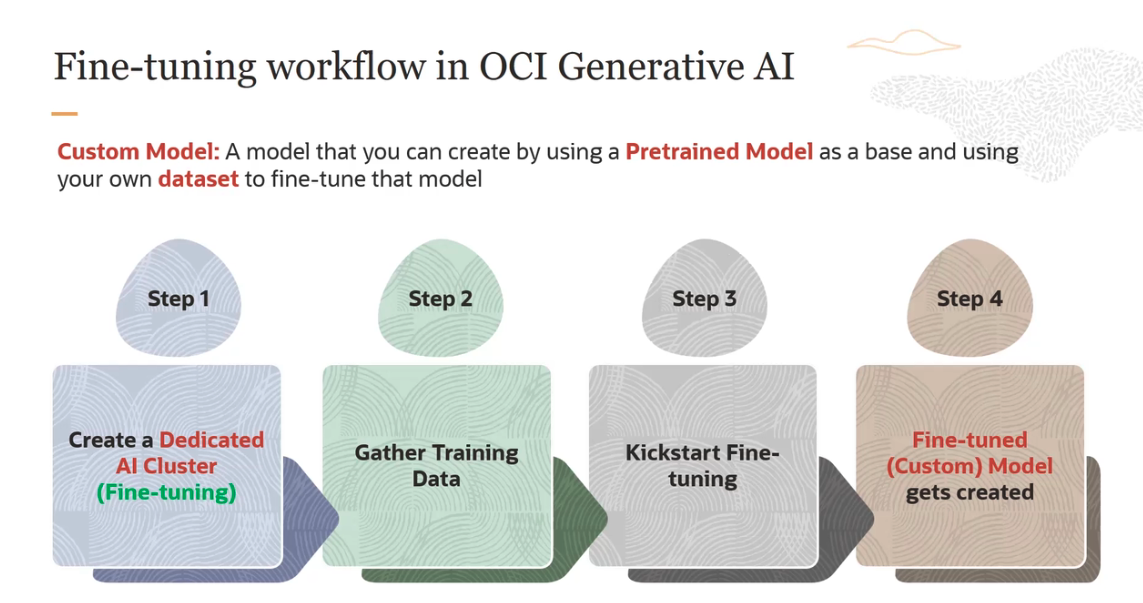
**Fine-tune and Inference in OCI Generative AI**



- In traditional machine learning terminology, inference refers to the process of using a trained machine learning model to make predictions or decisions based on new input data.

- But in case of large language models, what inference refers to is the model receiving new text as input, and generating output text based on what it has learned during training and fine-tuning.

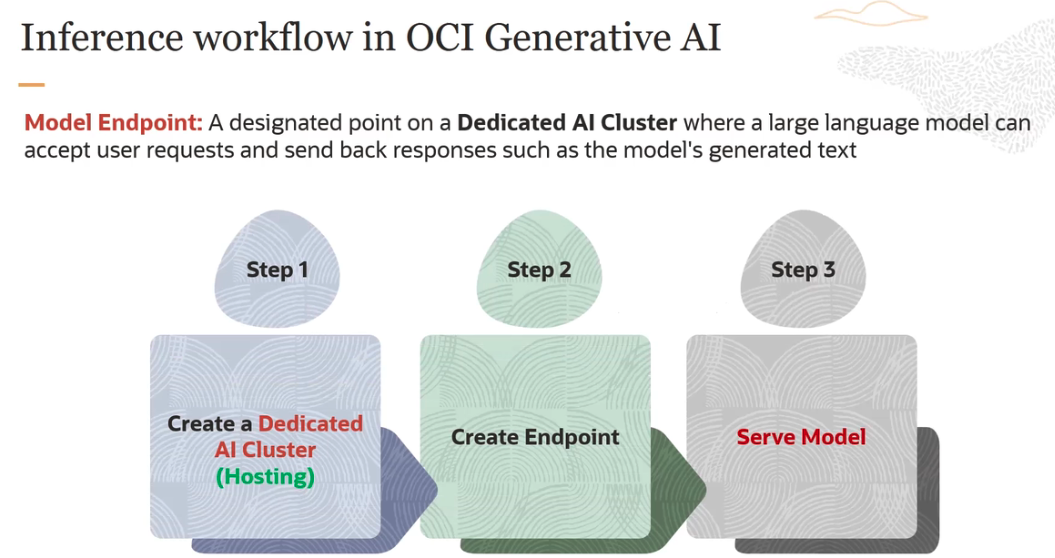


The process is quite straightforward. The first step is you create a dedicated AI cluster. And the type of the cluster is fine-tuning cluster. Then you gather training data.

Or you could gather training data, and then create a dedicated AI cluster. So you could interchange these steps.

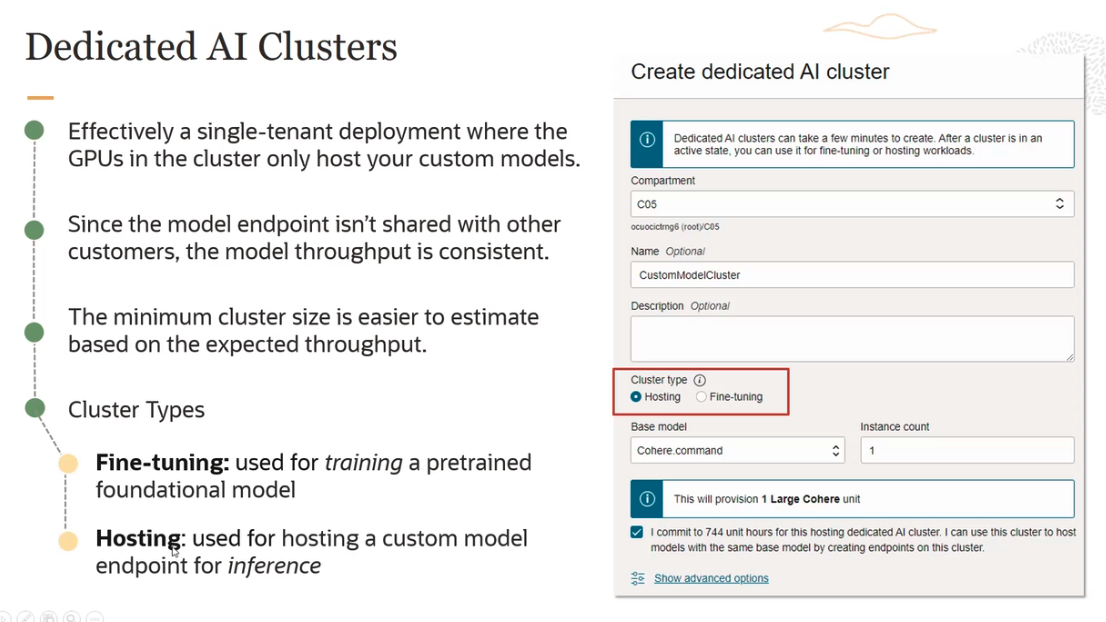
The third step is to kick start the fine-tuning process. And the fourth step is where you get the fine-tuned or the custom model created, and you have that model. So this is what the workflow looks like.

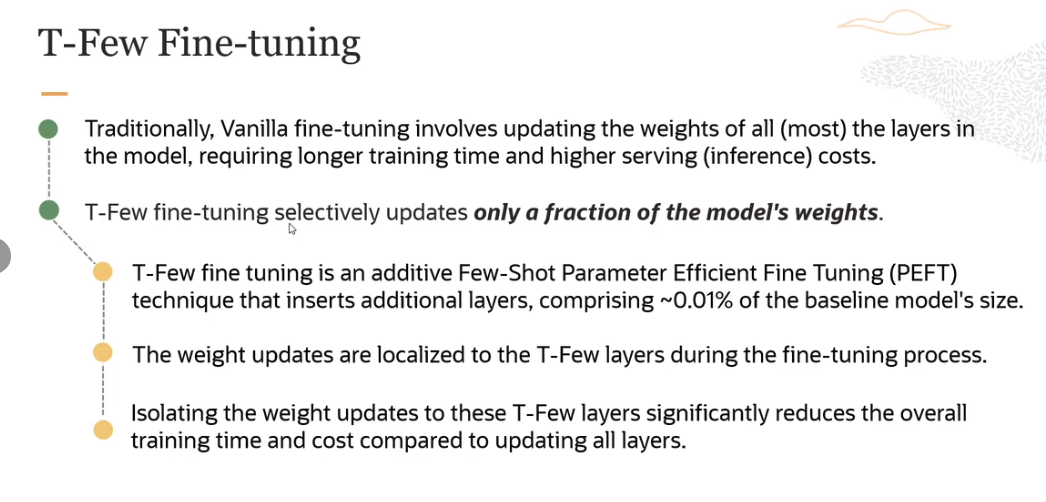
**Custom Model** - Custom model is a model that you create by using a pre-trained model as a base, and using your own data set to fine tune that model. It's called a custom model.



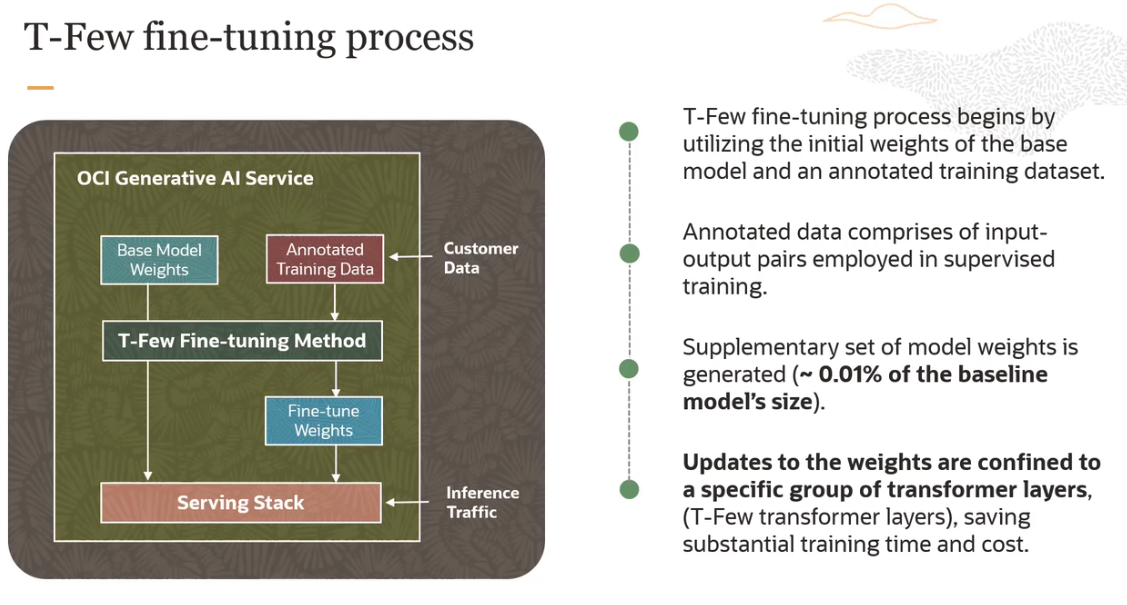
The first step is, you create a dedicated AI cluster, like we did with fine-tuning. Now the cluster is called a hosting cluster. And then you create the endpoint. And then, basically, you serve the traffic, the production load, or you serve the model.

**Model Endpoint:** Model Endpoint is a designated point on the dedicated AI cluster, where the large language model can accept user request, and send back responses, such as models generated text.



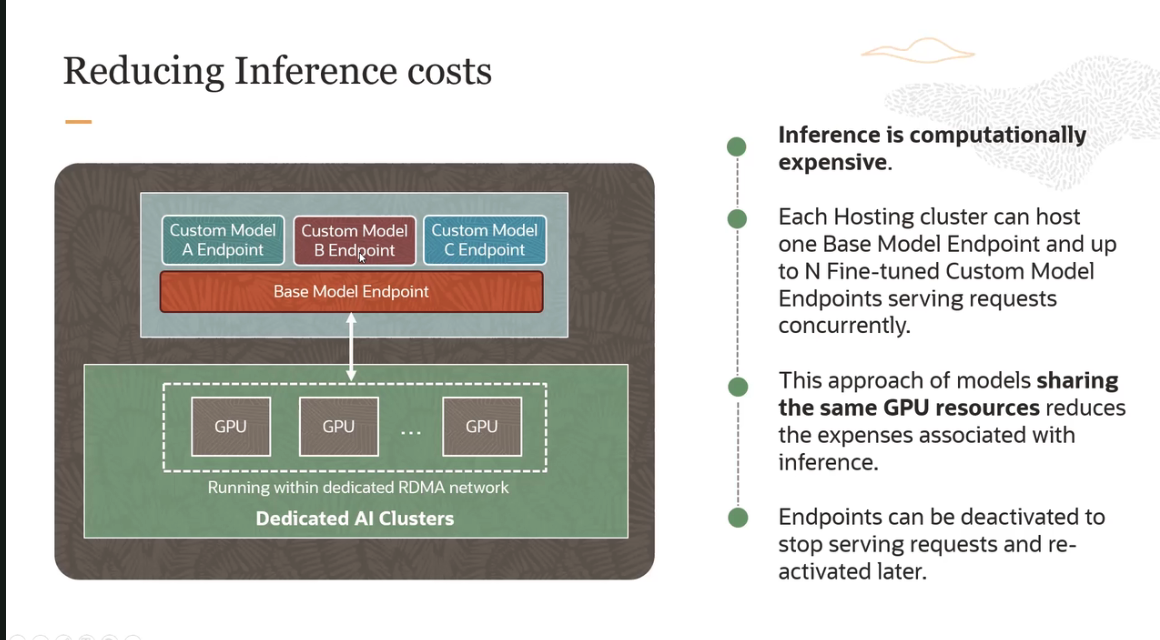


It is a fine-une technique. it's much efficient. It's much cheaper, much faster.



The process begins by utilizing the initial weights of the base model, and an annotated training data set. Now, this annotated training data set is the labeled data set, which is basically input/output pairs employed in supervised training.

So you take this annotated training data set. You have the base model weights. And then, basically, you generate a supplementary set of model weights, which is, again, roughly around 0.01% of the baseline model's size. So these fine-tune weights are generated. And now you propagate these weights to a specific group of transformer layers, which are called the T-Few transformer layers, rather than updating all the layers in the model. And doing so, basically, you reduce the training time, and also the cost.

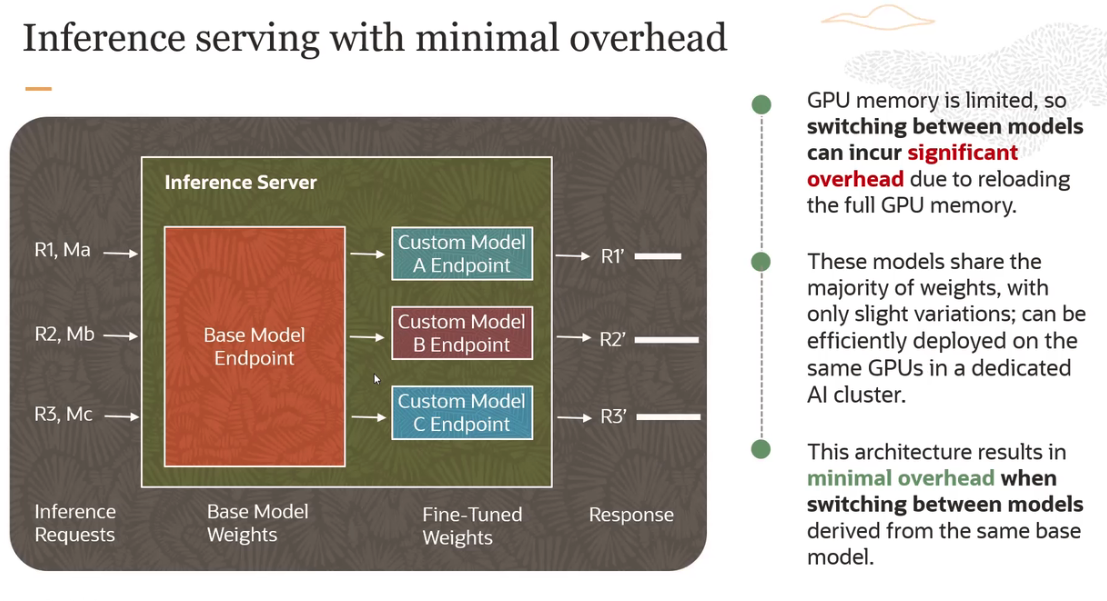


How do we handle the GPU memory?

- In a typical scenario, GPU memory is limited. So when you switch between models, it can incur significant **overhead** due to reloading the full GPU memory.

- Means, GPUs are particularly well suited for deep learning tasks, due to their parallel processing capabilities. But they offer a finite amount of memory. When you switch between models, the GPU typically needs to load the entire model into its memory space before it can start processing data. This is what we refer to as **overhead** here.

- This **overhead** includes the time and computational resources required to transfer the model data from the system memory to the GPU memory, as well as any initialization or setup tasks needed to prepare the GPU for processing with the new model.



In case of OCI Generative AI service, basically what we are doing is, these models are sharing the majority of weights, in case of a T-Few fine-tuning process. And with only slight variations between them or among them. So they can be efficiently deployed on the same GPUs in a dedicated AI cluster.

- So you have a base model here. And then you have several custom models. And the weight difference between the base model and model A, B, C, is minimal. Because we are updating only roughly, approximately 0.01% of the base model weights.

- So what this does is, because of this architecture, this results in minimal overhead when you have to switch between models derived from the same base model. Because what is happening is, you are deploying the base model alongside its fine-tuned versions.

- We are using the T-Few fine-tuning here. And this allows for something which is referred to as parameter sharing, where the common parts of the models are loaded into memory once, and shared across different tasks or processes.

- This significantly reduces the total amount of memory required, compared to a scenario where each model with its entire set of parameters is loaded separately. So this is how we reduce the memory overhead.

- And as you can see here, we are getting several inference requests. And then we are returning these responses here. So it's much lower overhead because of the inherent architecture of the OCI Generative AI service and the dedicated clusters we have.