**Requests and Limits** are part of **resource management** and help control the CPU and memory (resources) consumed by containers within a pod.

**1. Resource Requests**

* A **request** is the amount of CPU or memory guaranteed to a container.
* Kubernetes uses requests to determine which node can schedule a pod.
* If a node has enough resources to fulfill the request, the pod will be scheduled on that node.

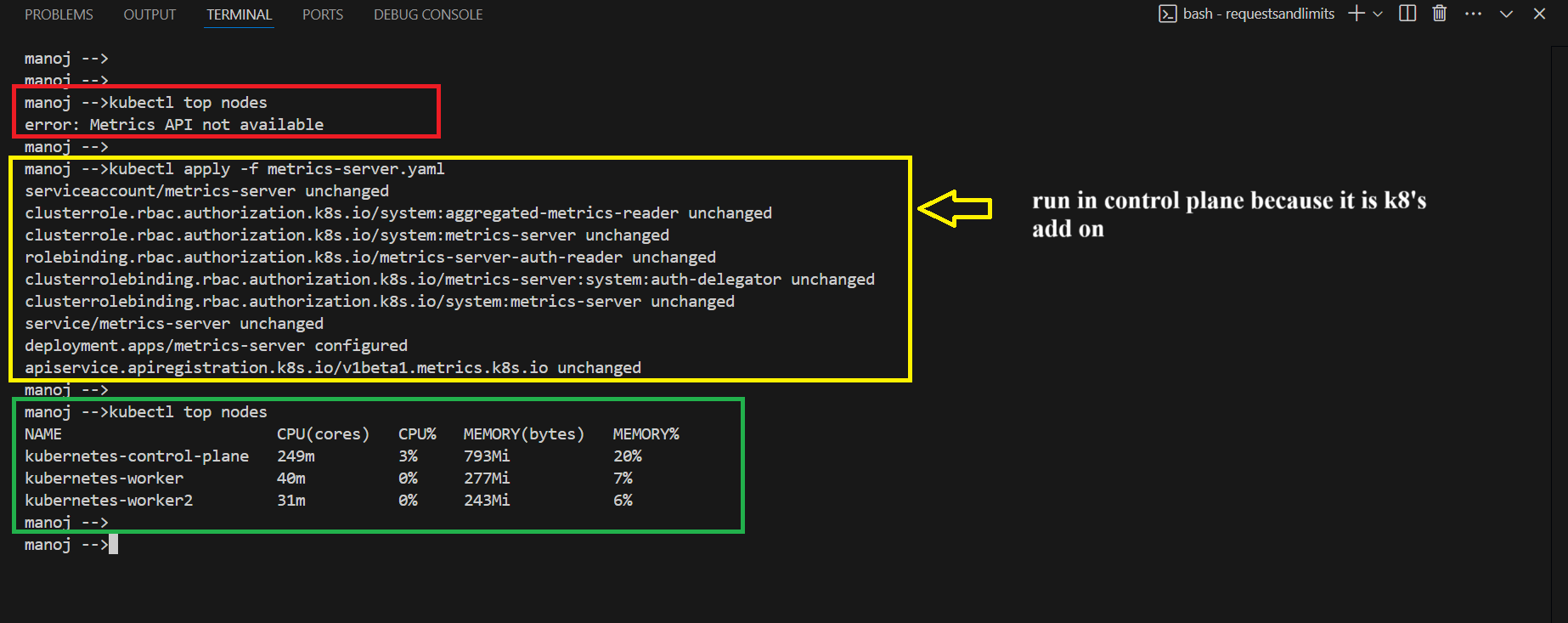
**2. Resource Limits**

* A **limit** is the maximum amount of CPU or memory a container can use.
* If a container exceeds its memory limit, it will be terminated, and if it exceeds its CPU limit, its usage will be throttled.
* **Some of the issues we can face in request and limits are:**
* **Insufficient Memory (or Insufficient CPU)**: The node doesn't have enough memory or CPU available to fulfill the pod's resource request.
* **OOMKilled**: The container was terminated because it exceeded its memory limit.
* **PodFitsResources**: The pod can't be scheduled because no node has sufficient resources to meet the pod's requests.
* **ResourceQuotaExceeded**: The pod exceeds the resource quota set for the namespace, preventing it from being created.
* **LimitRangeExceeded**: The pod's resource requests or limits exceed the constraints defined by the LimitRange policy in the namespace.
* **PodEvicted**: The pod was evicted due to resource pressure on the node, often because of insufficient memory.
* **ContainerCannotRun**: The container fails to start due to insufficient resources or incorrect configuration.
* **FailedScheduling**: Kubernetes was unable to find a suitable node to schedule the pod due to resource constraints.

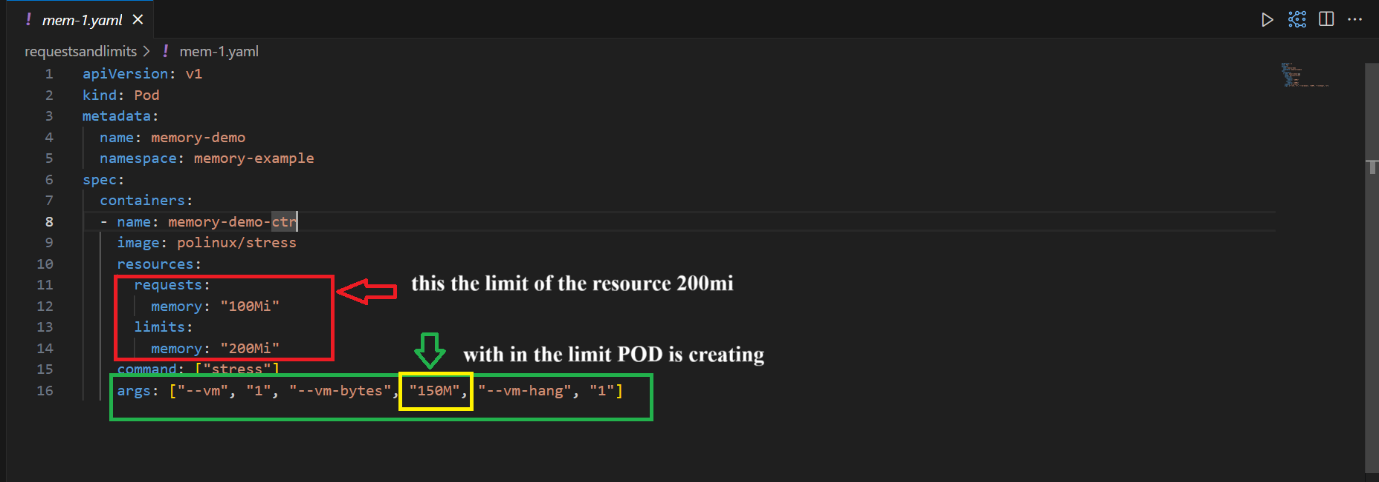
**For metric-server yaml use the link below to find the code**

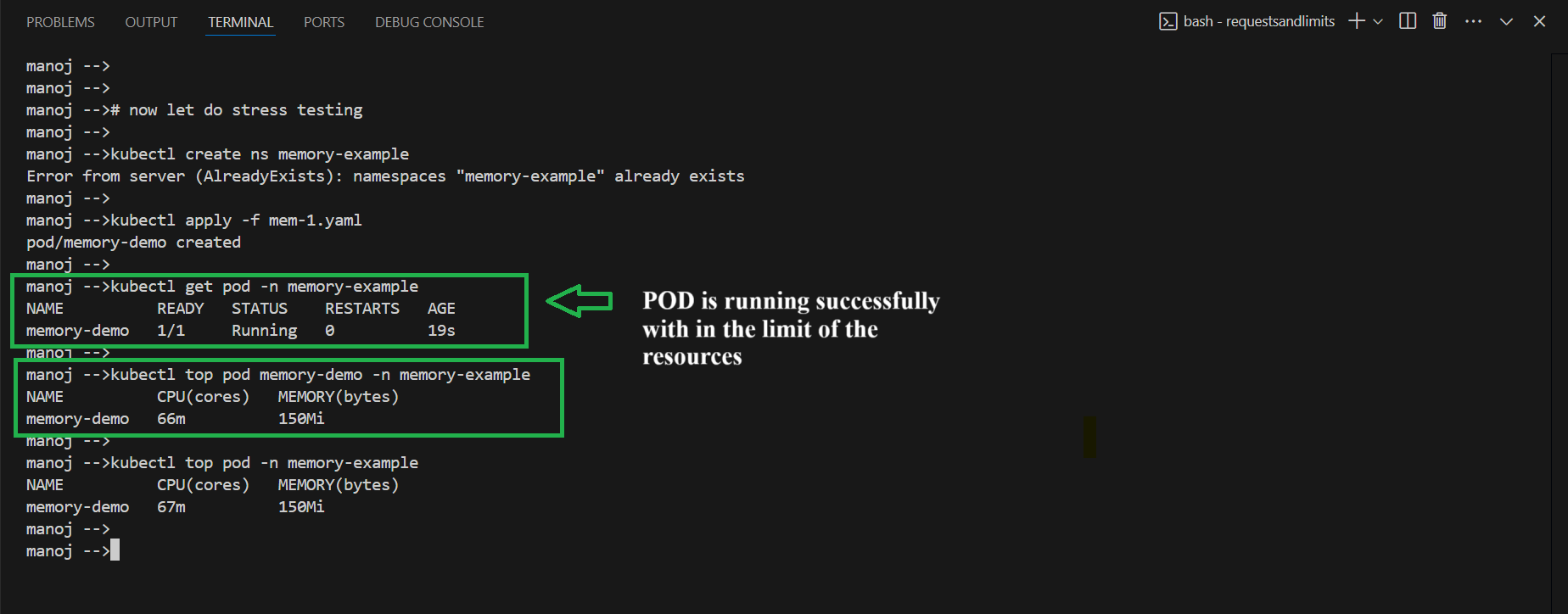
<https://drive.google.com/file/d/1CIbZkM_71xn4psUrGeIhfWTaKfD_5oZu/view?usp=drive_link>

Run the metric-server on kube-system namespace present in the control plane

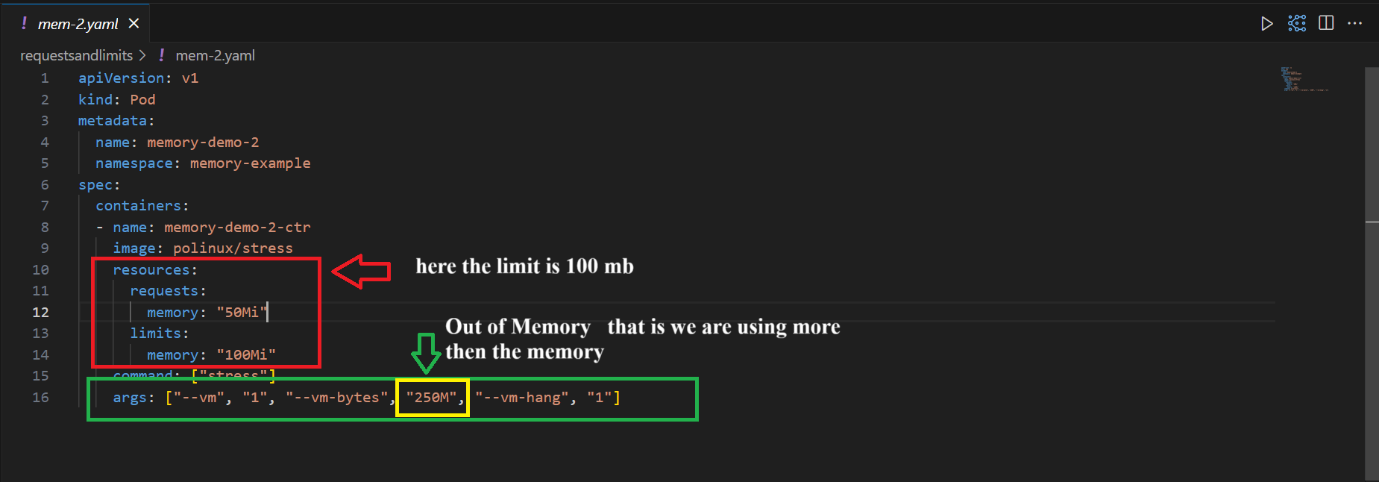


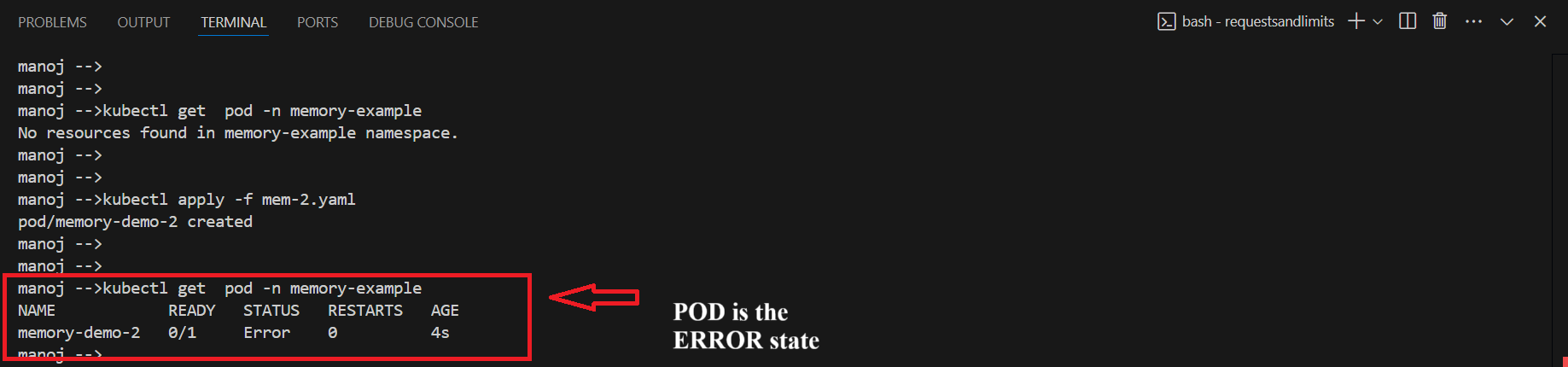
**Case 1: Request memory “100Mi” and limit the memory to “200Mi”. with in the limit POD is created.**



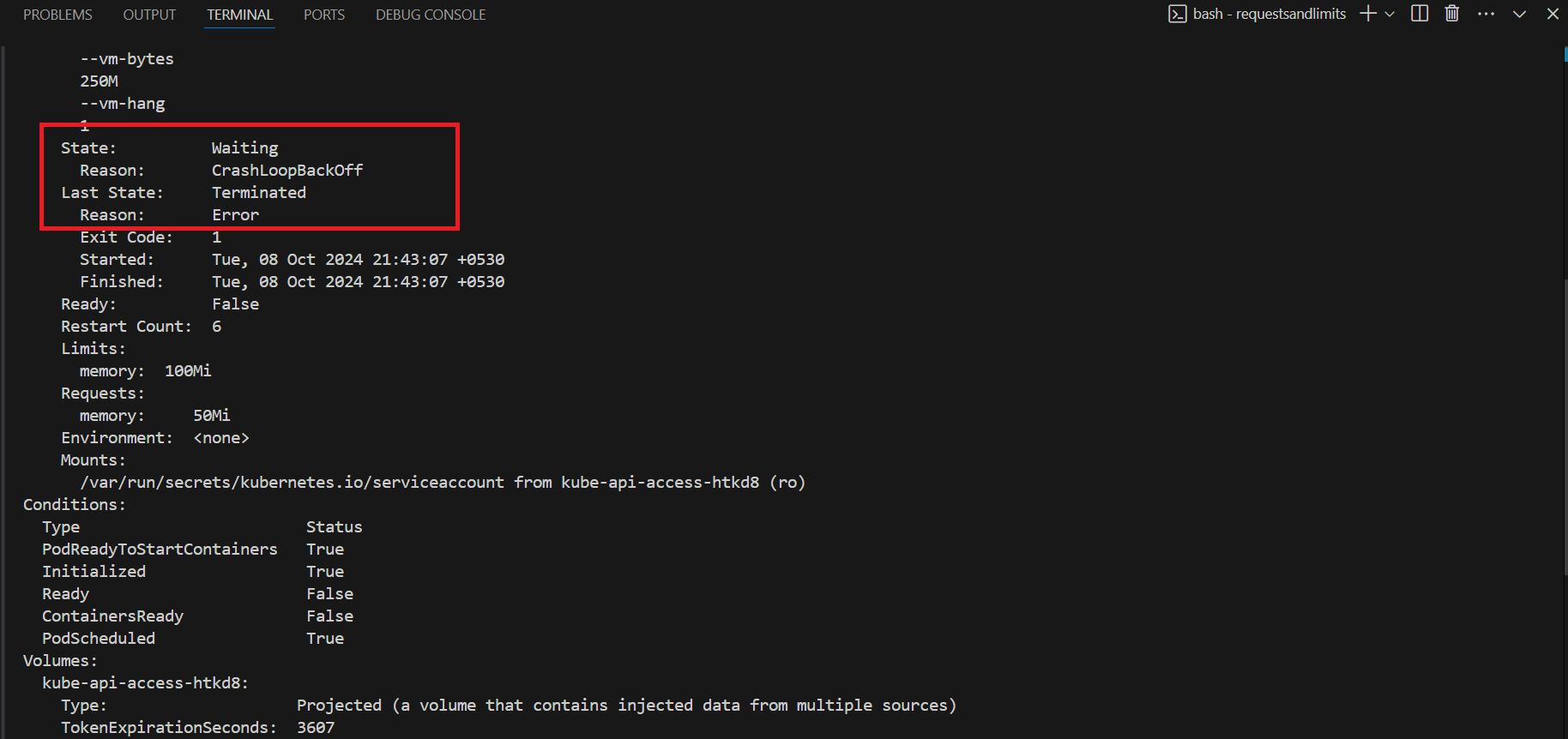


**Case 2: Request memory “50Mi” and limit the memory to “100Mi”. Although the container tries to allocate 250 MiB of memory, Kubernetes enforces a 100 MiB memory limit, so the container will likely be terminated due to exceeding the limit.**

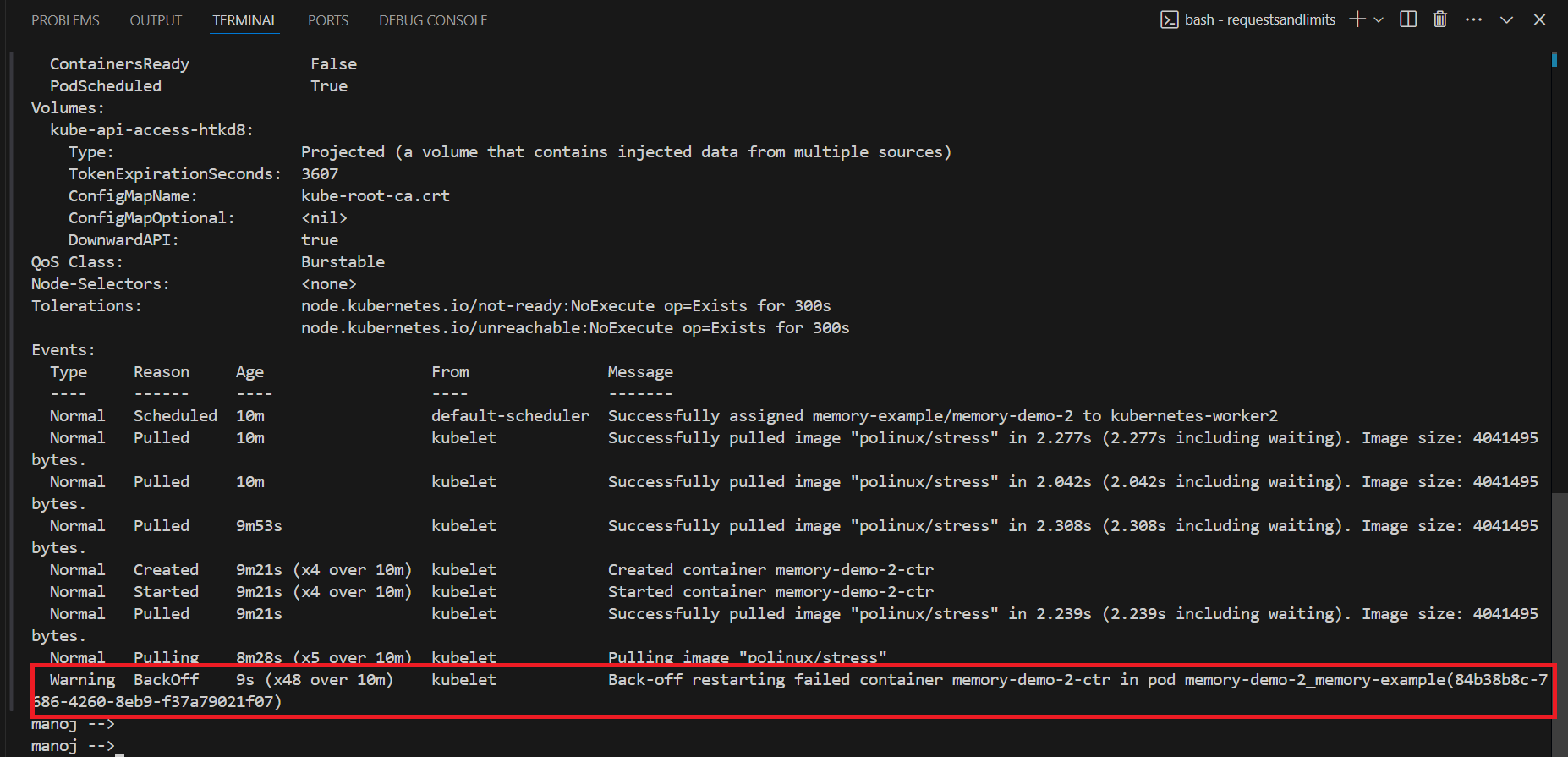




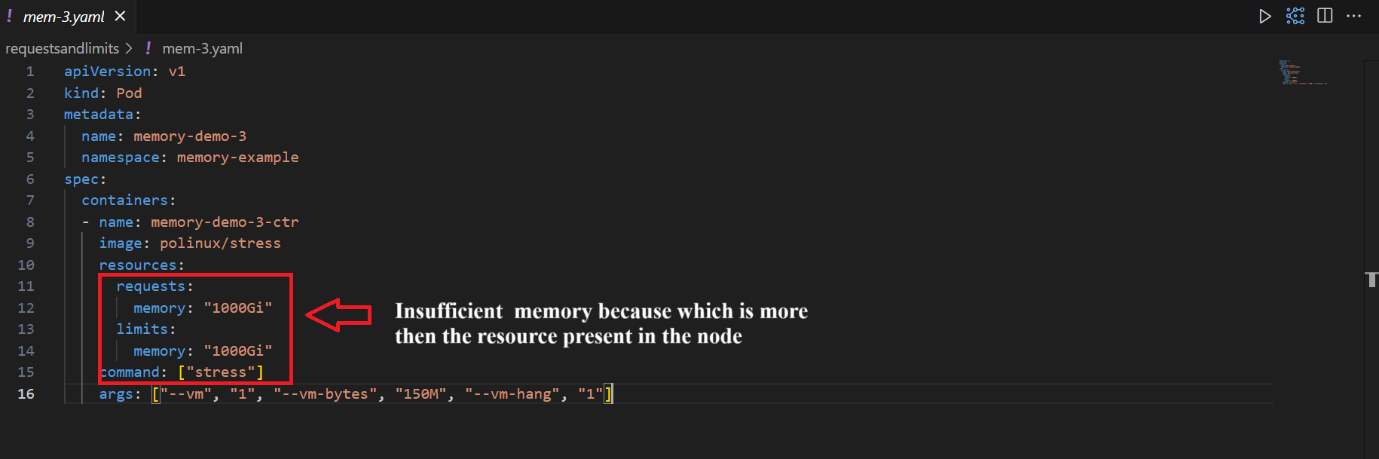




If the container exceeds the **100 MiB memory limit**, it will be terminated by Kubernetes (likely with an OOMKilled error).

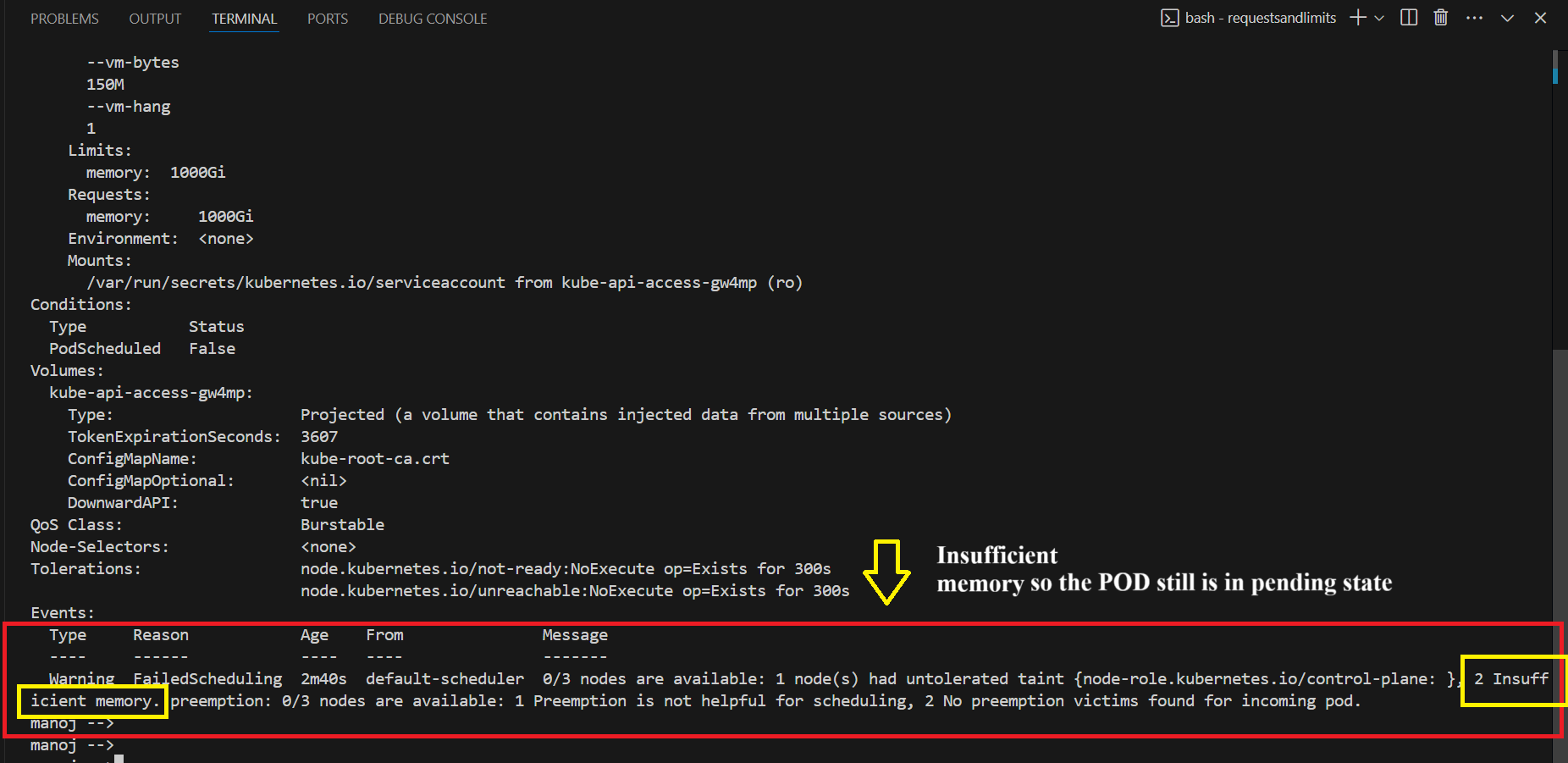


**Case 3: Request and Limit resources are specified here are more then the resource[memory] present in node**





In most clusters, such a pod would likely fail to be scheduled unless a node with very high memory capacity exists.



**Key Points:**

* **Requests**: Minimum resources guaranteed for a container.
* **Limits**: Maximum resources a container is allowed to consume.
* If you only set a **request**, Kubernetes will not limit resource usage beyond that.
* If you only set a **limit**, the pod might not be scheduled if the node doesn’t have sufficient resources.