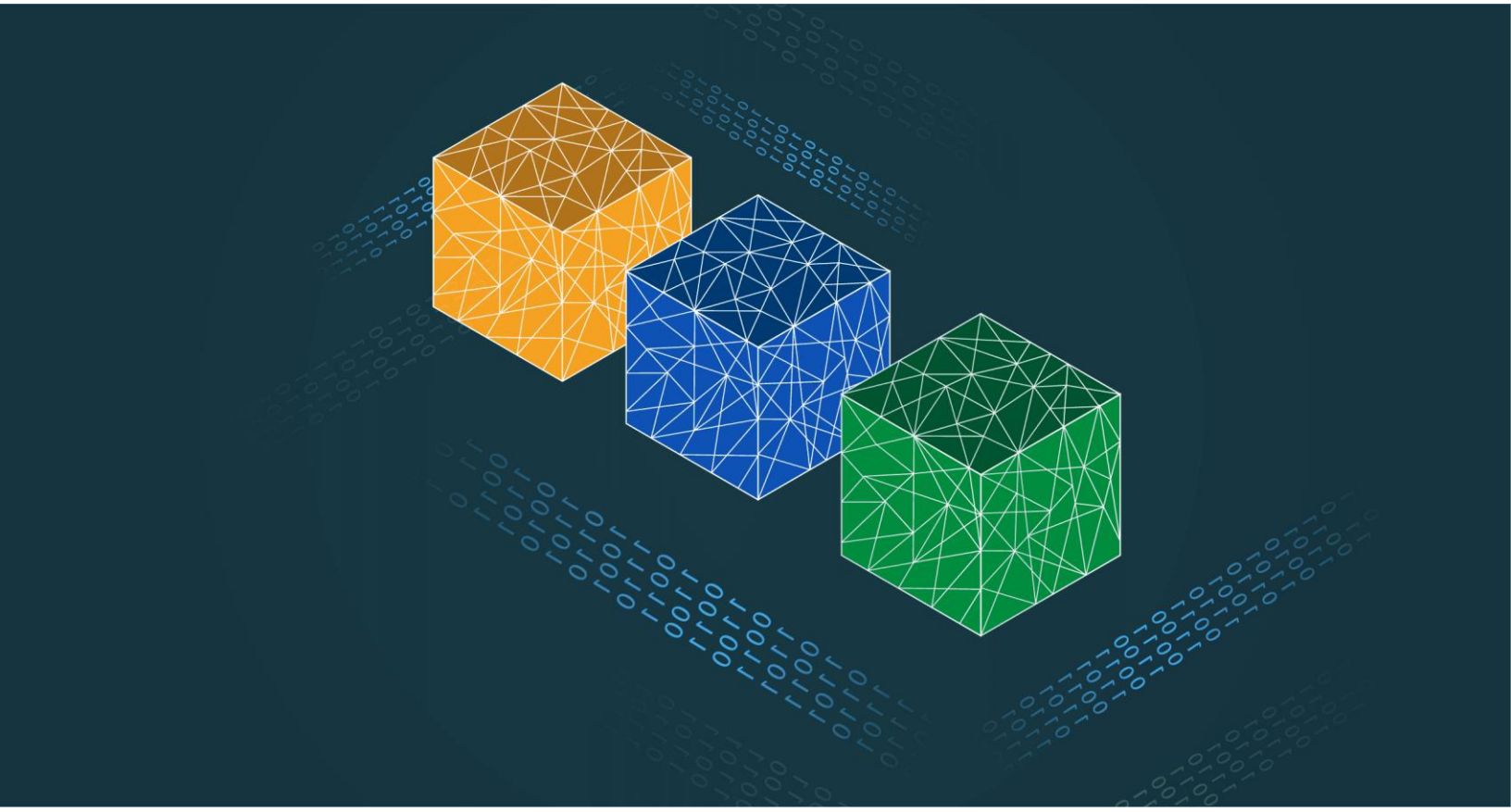
ProphetStor



Federator.ai GPU Booster Installation Guide

Federator.ai GPU Booster version 5.2

Installation Guide

ProphetStor Data Services, Inc.
830 Hillview Court, Suite 100
Milpitas, CA 95035 USA
Phone: 1.408.508.6255
Website: www.prophetstor.com

7.12.2024

Contents

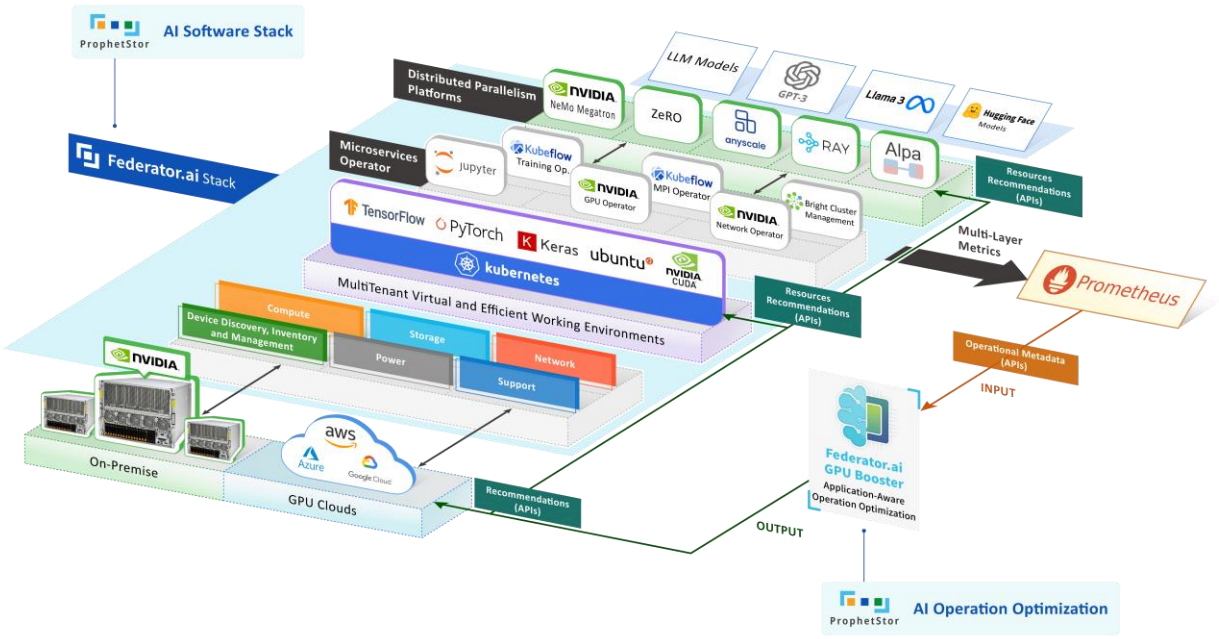
Overview

Federator.ai GPU Booster leverages advanced multi-layer correlation and machine learning technologies on Kubernetes to address the challenges of managing GPU resources in competitive AI and ML environments. Designed for MultiTenant settings, it efficiently orchestrates AI/ML workloads, particularly in large language model training.

By recommending GPU allocations to accommodate the varying demands of AI training workloads, Federator.ai GPU Booster optimizes resource usage and enhances training efficiency, enabling organizations to fully harness their AI/ML capabilities, thereby accelerating progress in the field.

Using advanced machine learning algorithms to predict application workloads, Federator.ai GPU Booster offers:

- Efficient GPU resource allocation by leveraging Multi-Instance GPU (MIG) technology to run MultiTenant training sessions in parallel, efficiently allocating the necessary resources based on ML algorithms
- Maximize total throughput by strategically reallocating GPU resources among concurrent GPU workloads
- AI-based workload prediction for containerized applications in Kubernetes
- Resource recommendations based on workload prediction, application, Kubernetes, and other related metrics
- Automatic provisioning of CPU/memory for generic Kubernetes application controllers/namespaces
- Actual cost and potential savings based on recommendations for clusters, Kubernetes applications, and Kubernetes namespaces



Federator.ai Stack Installation

Federator.ai GPU Booster runs on a Kubernetes cluster, managing GPU resources and workloads in Kubernetes clusters. To run Federator.ai GPU Booster on a GPU server and manage its Kubernetes cluster, it's essential to install a software stack on the GPU server in order to run GPU workloads and utilize Federator.ai GPU Booster for effective GPU system management. This software stack includes the Linux OS, Kubernetes, NVIDIA drivers/plugins/Operator, Prometheus, and Federator.ai GPU Booster.

The Federator.ai Stack

Federator.ai Stack is designed and developed to empower AI and ML innovations with a full suite of all-inclusive tools. Tailor-made for researchers, data scientists, and IT professionals, it takes away the pain of deploying and running your AI applications so that you can focus on breakthroughs. At the same time, we remove all the headaches of infra optimization.

If installing on a bare metal GPU server, an ISO image is provided to install the base OS. For a GPU server that has already been installed with a base OS, the Federator.ai Stack launcher script is provided for the installation process.

The following software components will be installed:

- **Base OS:** For a bare metal GPU server, boot the GPU server from the Federator.ai Stack ISO image, and an Ubuntu base OS will be installed.
- **Nvidia GPU Driver:** Nvidia GPU drivers and CUDA Toolkits will be installed on Ubuntu OS.
- **Kubernetes:** A Kubernetes cluster will be created if the GPU server is not part of an existing Kubernetes cluster.
- **Nvidia GPU Operator:** The Nvidia GPU Operator for Kubernetes, which includes the Nvidia device plugin, will be installed in the Kubernetes cluster.
- **Prometheus:** An open-source monitoring and altering toolkit for monitoring microservices and containers for Kubernetes will be installed if it is not already installed.
- **Federator.ai GPU Booster:** Federator.ai GPU Booster will be installed in the Kubernetes cluster.
- **AI/ML Frameworks:** (Optional) AI/ML frameworks and development toolkits will be installed in the Kubernetes cluster.

Hardware Requirements

The hardware requirements for a GPU Server are,

- **CPU:** Intel Xeon ES or above or Similar AMD CPUs
- **Memory:** Minimum of 32 GB*

- **GPU:** Nvidia H100/A100 recommended
- **Network:** At least 1 NIC with Internet access
- **Local Storage:** 1 TB SSD recommended
- **Persistent Storage:** 500GB NFS Storage (optional)

* *Note that the memory requirement stated above is for Federator.ai GPU Booster only. To run additional AI/ML workloads, please increase memory accordingly.*

Network Accessibility

- **Internet Accessibility**: Installing a Kubernetes cluster, essential drivers/toolkits/tools, Federator.ai Stack, and GPU workloads for the PoC involves downloading software packages and workload datasets from the internet. Therefore, internet access is necessary during the installation process and the preparation of GPU workloads. However, internet access will no longer be required once the installation and preparation phases are completed.

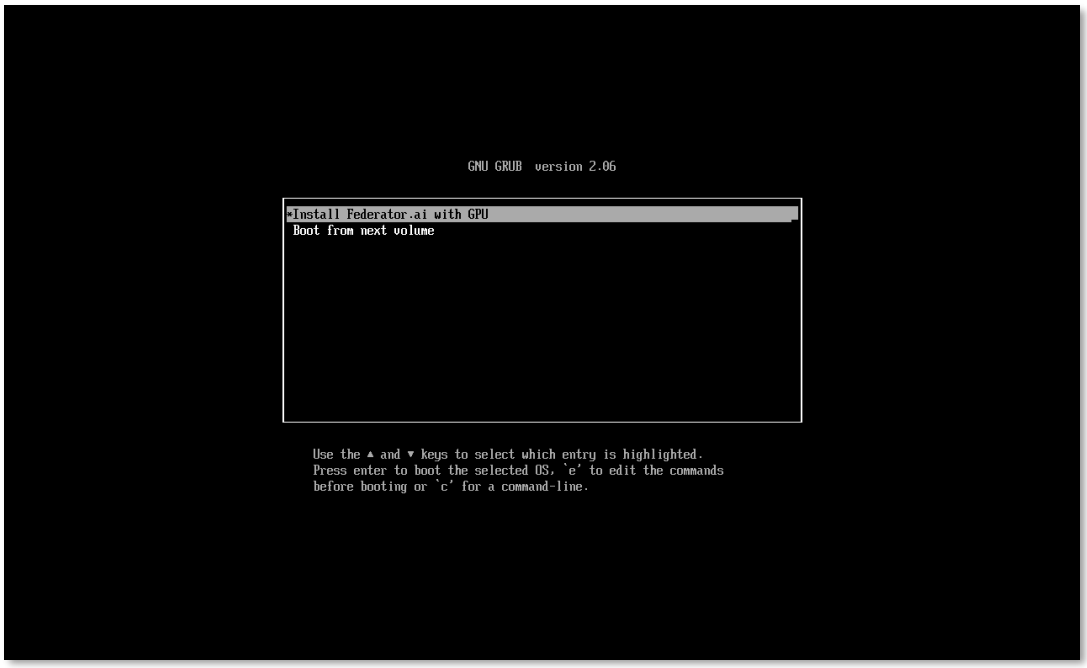
Installation Process

This section describes the procedure for installing the Federator.ai Stack on a GPU server. Following the script execution, the Federator.ai Stack (including Federator.ai GPU Booster) will be fully installed and operational.

1. If the GPU server is a bare metal system without an operating system installed, download the Federator.ai Stack ISO image, mount the ISO image to the GPU server, and boot the server from this ISO image.

   ```
   ~# curl -u <user:password> -L -O https://public.prophetstor.com/federator.ai/gpu-booster/f8ai-stack.iso
   ```

   Upon booting, an Ubuntu installation wizard will guide you through the complete setup of the Linux OS.



2. If the installed GPU server is already a node in an existing Kubernetes cluster, please ensure that the "`kubectl`" command is configured and the "`kubectl get nodes`" command can run properly.

3. Download and run the Federator.ai Stack launcher script with credentials.

```
~# curl -u <user:password> -L -O https://public.prophetstor.com/f8ai-stack/f8ai-stack-launcher.sh | sh
```

The script supports installing the GPU server either as a Kubernetes control-plane node or as a worker node. It is necessary to install a control-plane node before installing and adding a worker node to the Kubernetes cluster.

To install the GPU server as a Kubernetes control-plane node (or as an all-in-one node), answer "n" to the question:

```
Configure as Kubernetes Worker Node (y/n) [n]: n
```

Otherwise, answer "y" and provide the IP address of the control-plane node to install it as a worker node.

```
Configure as Kubernetes Worker Node (y/n) [y]: y
Kubernetes control-plane IP Address []: 172.31.2.40
```

4. This script will prompt you to enter credentials, decide whether to enable custom GPU MIG configuration, and install NeMO.

```
Enable Customized MIG on GPU Instances (y/n) [y]: y
Install NVIDIA NeMo Framework (y/n) [y]: n
```

The script then automates the setup, deploys everything in the Federator.ai Stack. If the GPU server is not part of an existing Kubernetes cluster, a new Kubernetes cluster will be created, together with Prometheus.

```
root@k8s-1:~# curl -u xxx:xxx -L -O https://public.prophetstor.com/f8ai-stack/f8ai-stack-
launcher.sh | sh
Authentication is required to download setup files.
Username: xxx
Password:
 *************************
 ****  Running Setup   ****
 *************************

Using setup script version v1.0.0-b1001.
Existing physical network interfaces: enp0s2 enp0s3
renamed 'config_env.sh.tmp' -> 'config_env.sh'
Federator.ai Chart Version [5.2.0-3035]:

Configure as Kubernetes Worker Node (y/n) [n]: n
Enable Customized MIG on GPU Instances (y/n) [y]: y
Install NVIDIA NeMo Framework (y/n) [y]: n
renamed 'config_env.sh.tmp' -> 'config_env.sh'
   INFO: Start using existing kubernetes cluster
```

```
   =========================
   =   Setup successfully   =
   =========================

Press <enter> key to continue ...https://<cluster-node-ip>:31012
```

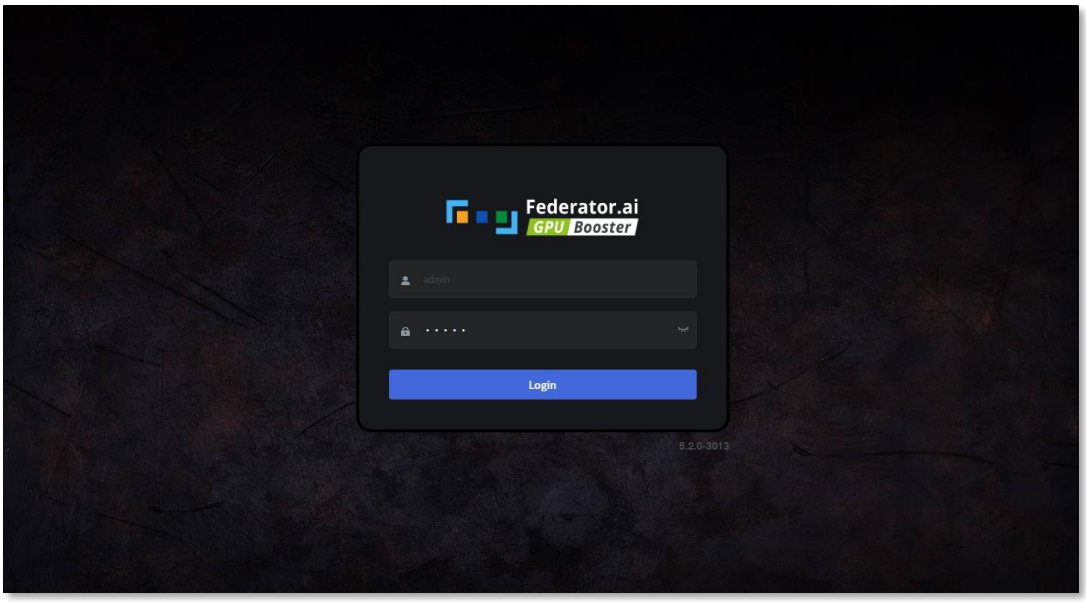
5. Once the installation is completed, a Kubernetes NodePort service (port 31012) is created to access the Federator.ai GPU Booster dashboard.

Federator.ai GPU Booster Dashboard Access

After a successful installation, the Fedeartor.ai GPU Booster can be accessed through the following

- **Dashboard URL**: **https://*<cluster-node-ip>*:31012**
- **Default Login**: Username: **admin** | Password: **admin**

This interface allows for comprehensive management and monitoring of your AI workloads.



Federator.ai GPU Booster Installation

Federator.ai GPU Booster runs on a Kubernetes cluster and can remotely manage GPU resources and workloads across multiple Kubernetes clusters. It supports installation in an existing Kubernetes cluster, whether or not it has GPU resources, by using helm chart.

Supported Platforms

- Kubernetes (K8S): 1.16 ~ 1.29
- Rancher v2.4.8, v2.5.8, v2.5.9
- OpenShift: 4.6 ~ 4.9
- VMware Tanzu

Resource Requirements

- Total resource requirements:
  - CPU: 10 cores
  - Memory: 32 GB Memory
  - Storage: 200 GB (ReadWriteOnce access mode)
- Resource requirements for the time-series database:

- It is required to have at least one worker node with more than 3 CPU cores and 8 GB of memory available.

- Persistent volumes
  - It is strongly recommended to use persistent volumes instead of ephemeral storage to store the data in the production environment. Any data on ephemeral storage will be lost after Federator.ai GPU Booster pods are restarted.

* *If your environment for testing does not have enough CPU/memory resources to install Federator.ai GPU Booster, you can disable resource requests for Federator.ai GPU Booster by setting the installation parameter '`--set global.resourcesRequestsEnabled="false"`'.*

Installation Process

1. Add Federator.ai GPU Booster helm chart repository

```
~# helm repo add prophetstor https://prophetstor-ai.github.io/federatorai-operator-helm/
```

2. Test the helm chart repository

```
~# helm search repo federatorai-gb
```

3. Install Federator.ai GPU Booster helm chart with a release name (e.g., 'federatorai-gb') in the 'federatorai' namespace.

   ⓘ *The default persistent storage class is 'default', use '--set global.storageClassName="<name>"' to configure an alternative storage class.*

```
~# $ helm install federatorai-gb prophetstor/federatorai-gb --namespace=federatorai --
create-namespace
NAME: federatorai-gb
LAST DEPLOYED: Wed Apr  3 09:06:28 2024
NAMESPACE: federatorai
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
CHART NAME: federatorai-gb
CHART VERSION: 5.2.0
APP VERSION: 5.2.0-3035

** Please be patient while the chart is being deployed **

Watch the Federator.ai GPU Booster pods status using the command:
    kubectl get pods -w --namespace federatorai -l app.kubernetes.io/part-of=federatorai-gb
```

4. Confirm Federator.ai GPU Booster pods are running and ready

```
~# kubectl get pods -n federatorai
NAME                                                READY   STATUS    RESTARTS      AGE
alameda-ai-6f979dff95-l894g                         1/1     Running   0             19m
alameda-ai-dispatcher-5c89b97764-zhj8k              1/1     Running   0             19m
alameda-datahub-6c554b9c66-9tfj2                    1/1     Running   0             19m
alameda-executor-5c65485d58-vqhhq                   1/1     Running   0             19m
alameda-influxdb-0                                  1/1     Running   0             19m
alameda-notifier-79889989bc-958sg                   1/1     Running   0             19m
alameda-rabbitmq-7f645b5b88-767dl                   1/1     Running   0             19m
fedemeter-api-7cf9f567b5-mw4kt                      1/1     Running   0             19m
fedemeter-influxdb-0                                1/1     Running   0             19m
federatorai-agent-586859c486-hs9cz                  1/1     Running   0             19m
federatorai-alert-detector-55f8dfb559-d6c4q         1/1     Running   0             19m
federatorai-alert-manager-57c68c9bf8-wzpll          1/1     Running   0             19m
federatorai-dashboard-backend-769d9b66f9-v9tfx      1/1     Running   0             19m
federatorai-dashboard-frontend-ff744768f-m4s7d      1/1     Running   0             19m
federatorai-data-adapter-5f6dfc7549-vzdsm           1/1     Running   0             19m
federatorai-gpu-device-plugin-dj9mt                 2/2     Running   0             19m
federatorai-postgresql-0                            1/1     Running   0             19m
federatorai-recommender-dispatcher-5684d96585-whzs9 1/1     Running   0             19m
federatorai-recommender-gpu-7d849f7b65-hwhdx        1/1     Running   0             19m
federatorai-recommender-worker-d5457d744-9kqjw      1/1     Running   0             19m
federatorai-rest-869f555c79-4jkw2                   1/1     Running   0             19m
federatorai-sys-exporter-6htxq                      1/1     Running   0             19m
```

5. By default, Federator.ai GPU Booster helm chart creates a NodePort service for Federator.ai GPU Booster dashboard. If your environment does not allow creating NodePort services, you'll need to manually configure Kubernetes to expose the Federator.ai GPU Booster dashboard service ("`federatorai-dashboard-frontend`") for external access.

```
~# kubectl get svc federatorai-dashboard-frontend-public -n federatorai
NAME                                 TYPE     CLUSTER-IP    EXTERNAL-IP PORT(S)         AGE
federatorai-dashboard-frontend-public NodePort 10.110.55.157 <none>      9001:31012/TCP  9d
```

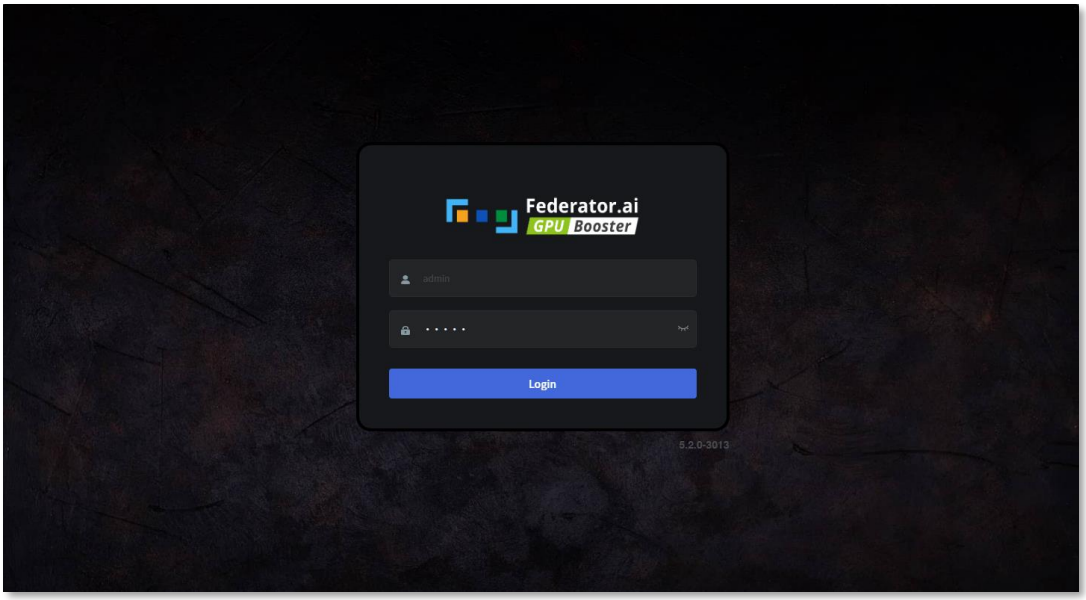
```
~# kubectl get svc federatorai-dashboard-frontend -n federatorai
NAME                            TYPE       CLUSTER-IP    EXTERNAL-IP  PORT(S)            AGE
federatorai-dashboard-frontend  ClusterIP  10.104.56.149 <none>       9000/TCP,9001/TCP  9d
```

Federator.ai GPU Booster Dashboard Access

After a successful installation, the Fedeartor.ai GPU Booster can be accessed through the following

- **Dashboard URL**: **https://*<cluster-node-ip>*:31012**
- **Default Login**: Username: **admin** | Password: **admin**

This interface allows for comprehensive management and monitoring of your AI workloads.



Uninstallation

- Uninstall/Delete the Federator.ai GPU Booster release (e.g., ‘federatorai-gb‘)

```
~# helm ls --all-namespaces
~# helm delete federatorai-gb --namespace=federatorai
```