

# Deploying and Installing SUSE AI in Air-Gapped Environments

## WHAT?

This document provides a comprehensive, step-by-step guide for the SUSE AI air-gapped deployment.

## WHY?

To help users successfully complete the air-gapped deployment process.

## GOAL

To learn enough information to deploy SUSE AI in both testing and production air-gapped environments.


## EFFORT

Less than one hour of reading and an advanced knowledge of Linux deployment.

SUSE AI is a versatile product consisting of multiple software layers and components. This document outlines the complete workflow for air-gapped deployment and installation of all SUSE AI dependencies, as well as SUSE AI itself. You can also find references to recommended hardware and software requirements, as well as steps to take after the product installation.



## Tip: Hardware and software requirements

For hardware, software and application-specific requirements, refer to [SUSE AI requirements \(https://documentation.suse.com/suse-ai/1.0/html/AI-requirements/index.html\)](https://documentation.suse.com/suse-ai/1.0/html/AI-requirements/index.html) .

Publication Date: 10 Oct 2025

## Contents

- 1 Air-gapped environments 3
- 2 Installation overview 5
- 3 Installing the Linux and Kubernetes distribution 7
- 4 Preparing the cluster for AI Library 44
- 5 Installing applications from AI Library 59
- 6 Steps after the installation is complete 92
- 7 Legal Notice 93
- Glossary 93
- A GNU Free Documentation License 98

# 1 Air-gapped environments

An air-gapped environment is a security measure where a single host or the whole network is isolated from all other networks, such as the public Internet. This “air gap” acts as a physical or logical barrier, preventing any direct connection that could be exploited by cyber threats.

## 1.1 Why you need an air-gapped environment?

The primary goal is to protect highly sensitive data and critical systems from unauthorized access, cyber attacks, malware and ransomware. Air-gapped environments are typically found in situations where security is of the utmost importance, such as:

- Military and government networks handling classified information.
- Industrial control systems (ICS) for critical infrastructure like power plants and water treatment facilities.
- Financial institutions and stock exchanges.
- Systems controlling nuclear power plants or other life-critical operations.

## 1.2 How do air-gapped environments work?

There are two types of air gaps:

### Physical air gaps

This is the most secure method, where the system is disconnected from any network. It might even involve placing the system in a shielded room.

### Logical air gaps

This type uses software controls such as firewall rules and network segmentation to create a highly restricted connection. While it offers more convenience, it is not as secure as a physical air gap because the air-gapped system is still technically connected to a network.

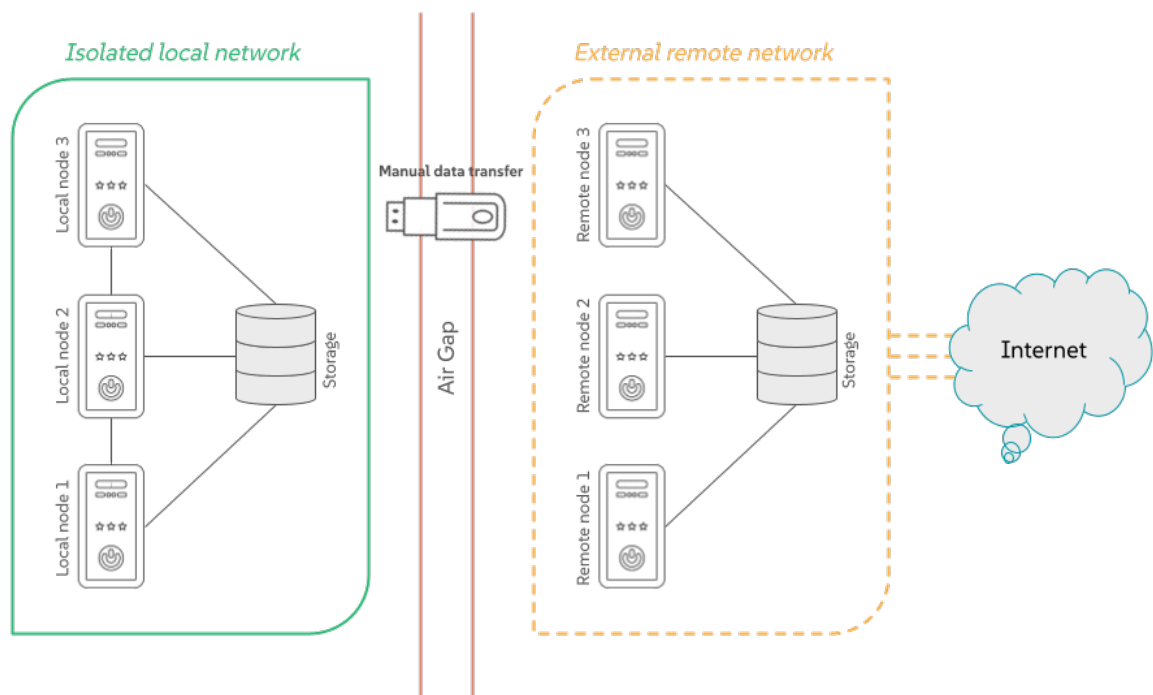


FIGURE 1: GENERAL SCHEMA OF AN AIR-GAPPED ENVIRONMENT

### 1.3 What challenges do air-gapped environments face?

When working in air-gapped systems, you usually face the following limitations:

#### Manual updates

Air-gapped systems cannot automatically receive software or security updates from external networks. You must manually download and install updates, which can be time-consuming and create vulnerabilities if not done regularly.

#### Insider threats and physical attacks

An air gap does not protect against threats that gain physical access to the system, such as a malicious insider with a compromised USB drive.

### Limited functionality

The lack of connectivity limits the system's ability to communicate with other devices or services, making it less efficient for many modern applications.

## 2 Installation overview

The following chart illustrates the installation process of SUSE AI. It outlines the following possible scenarios:

- You have clean cluster nodes prepared without a supported Linux operating system installed.
- You have a supported Linux operating system and Kubernetes distribution installed on cluster nodes.
- You have SUSE Rancher Prime and all supportive components installed on the Kubernetes cluster and are prepared to install the required applications from the AI Library.

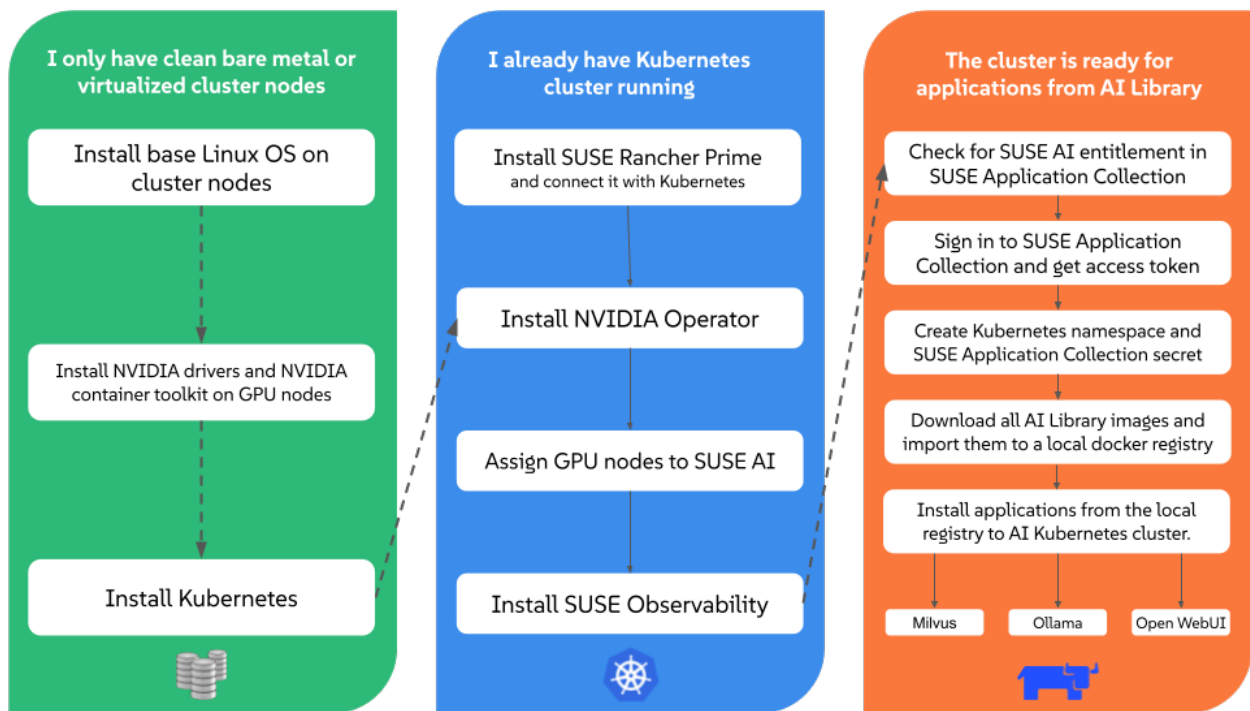


FIGURE 2: SUSE AI AIR-GAPPED INSTALLATION PROCESS

## 2.1 Air-gapped stack

The *air-gapped stack* is a set of scripts that ease the successful air-gap installation of certain SUSE AI components. To use them, you need to clone or download them from the stack's [GitHub repository \(https://github.com/SUSE/suse-ai-stack/tree/SUSEAI-79/air-gapped-install\)](https://github.com/SUSE/suse-ai-stack/tree/SUSEAI-79/air-gapped-install).

### 2.1.1 What scripts does the stack include?

The following scripts are included in the air-gapped stack:

SUSE-AI-mirror-nvidia.sh

Mirrors all RPM packages from a specified Web URLs.

SUSE-AI-get-images.sh

Downloads Docker images of SUSE AI applications from SUSE Application Collection.

SUSE-AI-load-images.sh

Loads downloaded Docker images into a custom Docker image registry.

### 2.1.2 Where do the scripts fit into the air-gap installation?

The scripts are required in several places during the SUSE AI air-gapped installation. The following simplified workflow outlines the intended usage:

1. Use `SUSE-AI-mirror-nvidia.sh` on a *remote* host to download the required NVIDIA RPM packages. Transfer the downloaded content to an air-gapped *local* host and add it as a Zypper repository to install NVIDIA drivers on *local* GPU nodes.
2. Use `SUSE-AI-get-images.sh` on a *remote* host to download Docker images of required SUSE AI components. Transfer them to an air-gapped *local* host.
3. Use `SUSE-AI-load-images.sh` to load the transferred Docker images of SUSE AI components into a custom *local* Docker image registry.
4. Install AI Library components on the *local* Kubernetes cluster from the *local* custom Docker registry.

## 3 Installing the Linux and Kubernetes distribution

This procedure includes the steps to install the base Linux operating system and a Kubernetes distribution for users who start deploying on cluster nodes from scratch. If you already have a Kubernetes cluster installed and running, you can skip this procedure and continue with [Section 5.1, “Installation procedure”](#).

1. Install and register a supported Linux operating system on each cluster node. We recommend using one of the following operating systems:
  - SUSE Linux Enterprise Server 15 SP6 for a traditional non-transactional operating system. For more information, see [Section 3.1, “Installing SUSE Linux Enterprise Server”](#).
  - SUSE Linux Micro 6.1 for an immutable transactional operating system. For more information, see [SUSE Linux Micro 6.1 documentation \(https://documentation.suse.com/sle-micro/6.1/\)](https://documentation.suse.com/sle-micro/6.1/).

For a list of supported operating systems, refer to <https://www.suse.com/suse-rancher/support-matrix/all-supported-versions/>.

2. Install the NVIDIA GPU driver on cluster nodes with GPUs. Refer to [Section 3.2, “Installing NVIDIA GPU drivers”](#) for details.

3. Install Kubernetes on cluster nodes. We recommend using the supported SUSE Rancher Prime: RKE2 distribution. Refer to [SUSE Rancher Prime: RKE2 Air-Gapped Installation \(https://documentation.suse.com/cloudnative/rke2/latest/en/install/airgap.html\)](https://documentation.suse.com/cloudnative/rke2/latest/en/install/airgap.html) for details. For a list of supported Kubernetes platforms, refer to <https://www.suse.com/suse-rancher/support-matrix/all-supported-versions/>.

## 3.1 Installing SUSE Linux Enterprise Server

Use the following procedures to install SLES on all supported hardware platforms. They assume you have successfully booted into the installation system. For more detailed installation instructions and deployment strategies, refer to [SUSE Linux Enterprise Server Deployment Guide \(https://documentation.suse.com/sles/15-SP6/html/SLES-all/book-deployment.html\)](https://documentation.suse.com/sles/15-SP6/html/SLES-all/book-deployment.html).

### 3.1.1 The Unified Installer

Starting with SLES 15, the installation medium consists only of the Unified Installer, a minimal system for installing, updating and registering all SUSE Linux Enterprise base products. During the installation, you can add functionality by selecting modules and extensions to be installed on top of the Unified Installer.

### 3.1.2 Installing offline or without registration

The default installation medium `15 SP6-Online-ARCH-GM-media1.iso` is optimized for size and does not contain any modules and extensions. Therefore, the installation requires network access to register your product and retrieve repository data for the modules and extensions.

For installation without registering the system, use the `15 SP6-Full-ARCH-GM-media1.iso` image from <https://www.suse.com/download/sles/> and refer to [Installing without registration \(https://documentation.suse.com/sles/15-SP6/html/SLES-all/cha-install.html#sec-yast-install-scc-registration-none\)](https://documentation.suse.com/sles/15-SP6/html/SLES-all/cha-install.html#sec-yast-install-scc-registration-none).





## Tip: Copying the installation media image to a removable flash disk

Use the following command to copy the contents of the installation image to a removable flash disk.

```
> sudo dd if=IMAGE of=FLASH_DISK bs=4M && sync
```

*IMAGE* needs to be replaced with the path to the 15 SP6-Online-ARCH-GM-media1.iso or 15 SP6-Full-ARCH-GM-media1.iso image file. *FLASH\_DISK* needs to be replaced with the flash device. To identify the device, insert it and run:

```
# grep -Ff <(hwinfo --disk --short) <(hwinfo --usb --short)
disk:
  /dev/sdc                General USB Flash Disk
```

Make sure the size of the device is sufficient for the desired image. You can check the size of the device with:

```
# fdisk -l /dev/sdc | grep -e "^/dev"
/dev/sdc1 *      2048 31490047 31488000  15G 83 Linux
```

In this example, the device has a capacity of 15 GB. The command to use for the 15 SP6-Full-ARCH-GM-media1.iso would be:

```
dd if=15 SP6-Full-ARCH-GM-media1.iso of=/dev/sdc bs=4M && sync
```

The device must not be mounted when running the **dd** command. Note that all data on the partition will be erased.

### 3.1.3 The installation procedure

To install SLES, boot or IPL into the installer from the Unified Installer medium and start the installation.

### 3.1.3.1 Language, keyboard and product selection

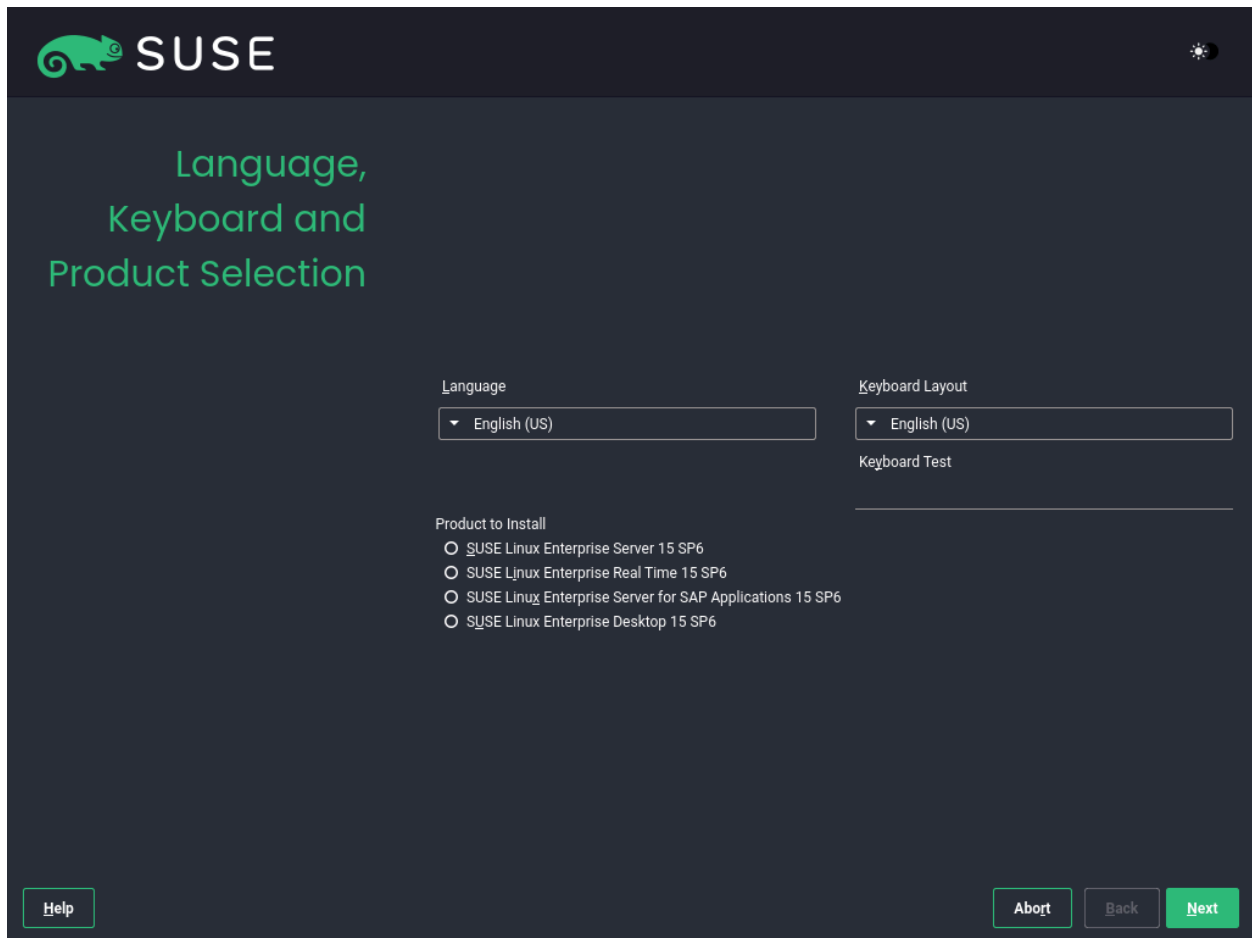


FIGURE 3: LANGUAGE, KEYBOARD AND PRODUCT SELECTION


The *Language* and *Keyboard Layout* settings are initialized with the language you chose on the boot screen. If you do not change the default, it remains English (US). Change the settings here, if necessary. Use the *Keyboard Test* text box to test the layout.

Select SUSE Linux Enterprise Server 15 SP6 for installation. You need to have a registration code for the product. Proceed with *Next*.



#### Tip: Light and high-contrast themes

If you have difficulty reading the labels in the installer, you can change the widget colors and theme.

Click the  button or press **Shift + F3** to open a theme selection dialog. Select a theme from the list and *Close* the dialog.

**Shift + F4** switches to the color scheme for vision-impaired users. Press the buttons again to switch back to the default scheme.

### 3.1.3.2 License agreement



FIGURE 4: LICENSE AGREEMENT

Read the License Agreement. It is presented in the language you have chosen on the boot screen. Translations are available via the *License Language* drop-down list. You need to accept the agreement by checking *I Agree to the License Terms* to install SLES. Proceed with *Next*.

### 3.1.3.3 Network settings

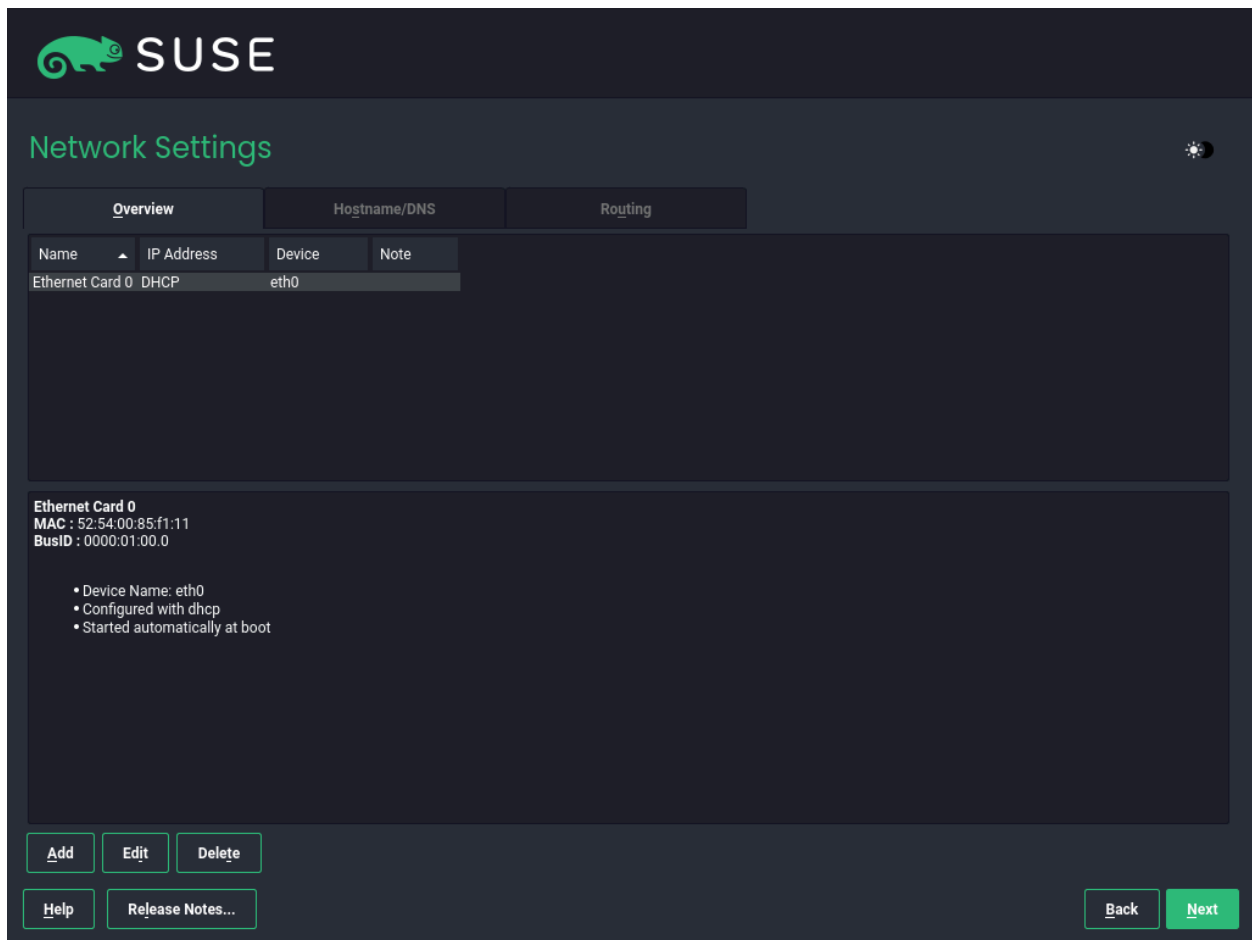
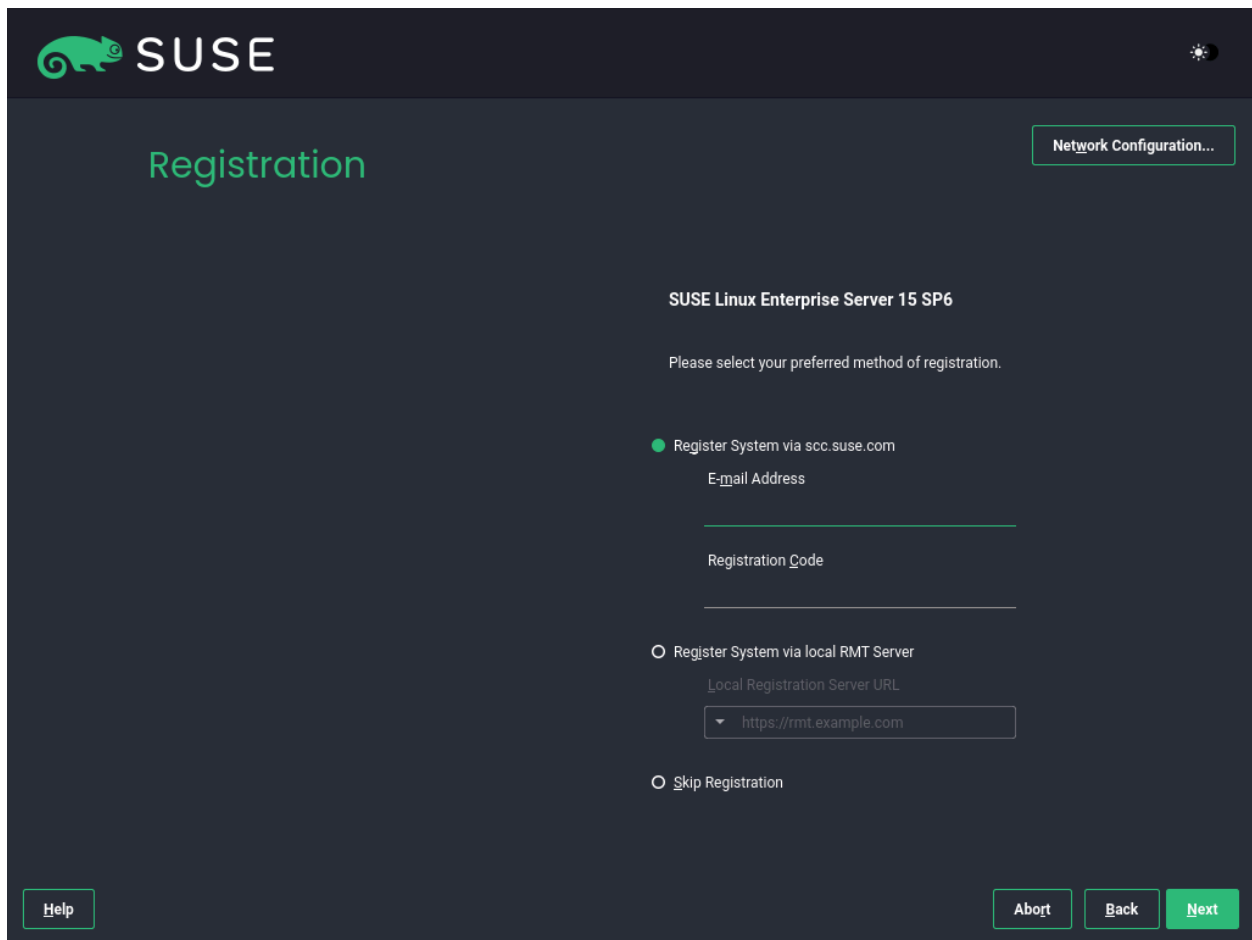


FIGURE 5: NETWORK SETTINGS

A system analysis is performed, where the installer probes for storage devices and tries to find other installed systems. If the network was automatically configured via DHCP during the start of the installation, you are presented the registration step.

If the network is not yet configured, the *Network Settings* dialog opens. Choose a network interface from the list and configure it with *Edit*. Alternatively, *Add* an interface manually. See the sections on [installer network settings \(https://documentation.suse.com/sles/15-SP6/html/SLES-all/cha-install.html#sec-yast-install-network\)](https://documentation.suse.com/sles/15-SP6/html/SLES-all/cha-install.html#sec-yast-install-network) and [configuring a network connection with YaST \(https://documentation.suse.com/sles/15-SP6/html/SLES-all/cha-network.html#sec-network-yast\)](https://documentation.suse.com/sles/15-SP6/html/SLES-all/cha-network.html#sec-network-yast) for more information. If you prefer to do an installation without network access, skip this step without making any changes and proceed with *Next*.

### 3.1.3.4 Registration



The image shows the SUSE Linux Enterprise Server 15 SP6 Registration window. The window has a dark blue header with the SUSE logo and a 'Network Configuration...' button. The main title is 'Registration'. Below the title, it says 'SUSE Linux Enterprise Server 15 SP6' and 'Please select your preferred method of registration.' There are three radio button options: 'Register System via scc.suse.com' (selected), 'Register System via local RMT Server', and 'Skip Registration'. The 'Register System via scc.suse.com' option has two input fields: 'E-mail Address' and 'Registration Code'. The 'Register System via local RMT Server' option has a dropdown menu for 'Local Registration Server URL' with the value 'https://rmt.example.com'. At the bottom, there are three buttons: 'Help', 'Abort', and 'Next'.

FIGURE 6: REGISTRATION

To get technical support and product updates, you need to register and activate SLES with the SUSE Customer Center or a local registration server. Registering your product at this stage also grants you immediate access to the update repository. This enables you to install the system with the latest updates and patches available.

When registering, repositories and dependencies for modules and extensions are loaded from the registration server.

#### *Register system at scc.suse.com*

To register at the SUSE Customer Center, enter the *E-mail Address* associated with your SUSE Customer Center account and the *Registration Code* for SLES. Proceed with *Next*.

### Register system via local RMT server

If your organization provides a local registration server, you may alternatively register to it. Activate *Register System via local RMT Server* and either choose a URL from the drop-down list or type in an address. Proceed with *Next*.

### Skip registration

If you are offline or want to skip registration, activate *Skip Registration*. Accept the warning with *OK* and proceed with *Next*.



### Important: Skipping the registration

Your system and extensions need to be registered to retrieve updates and to be eligible for support. Skipping the registration is only possible when installing from the 15 SP6-Full-ARCH-GM-medial.iso image.

If you do not register during the installation, you can do so at any time later from the running system. To do so, run *YaST > Product Registration* or the command-line tool **SUSE-Connect**.



### Tip: Installing product patches at installation time

After SLES has been successfully registered, you are asked whether to install the latest available online updates during the installation. If choosing *Yes*, the system will be installed with the most current packages without having to apply the updates after installation. Activating this option is recommended.



### Note: Firewall settings for receiving updates

By default, the firewall on SUSE AI only blocks incoming connections. If your system is behind another firewall that blocks outgoing traffic, make sure to allow connections to <https://scc.suse.com/> and <https://updates.suse.com> on ports 80 and 443 to receive updates.

### 3.1.3.5 Extension and module selection

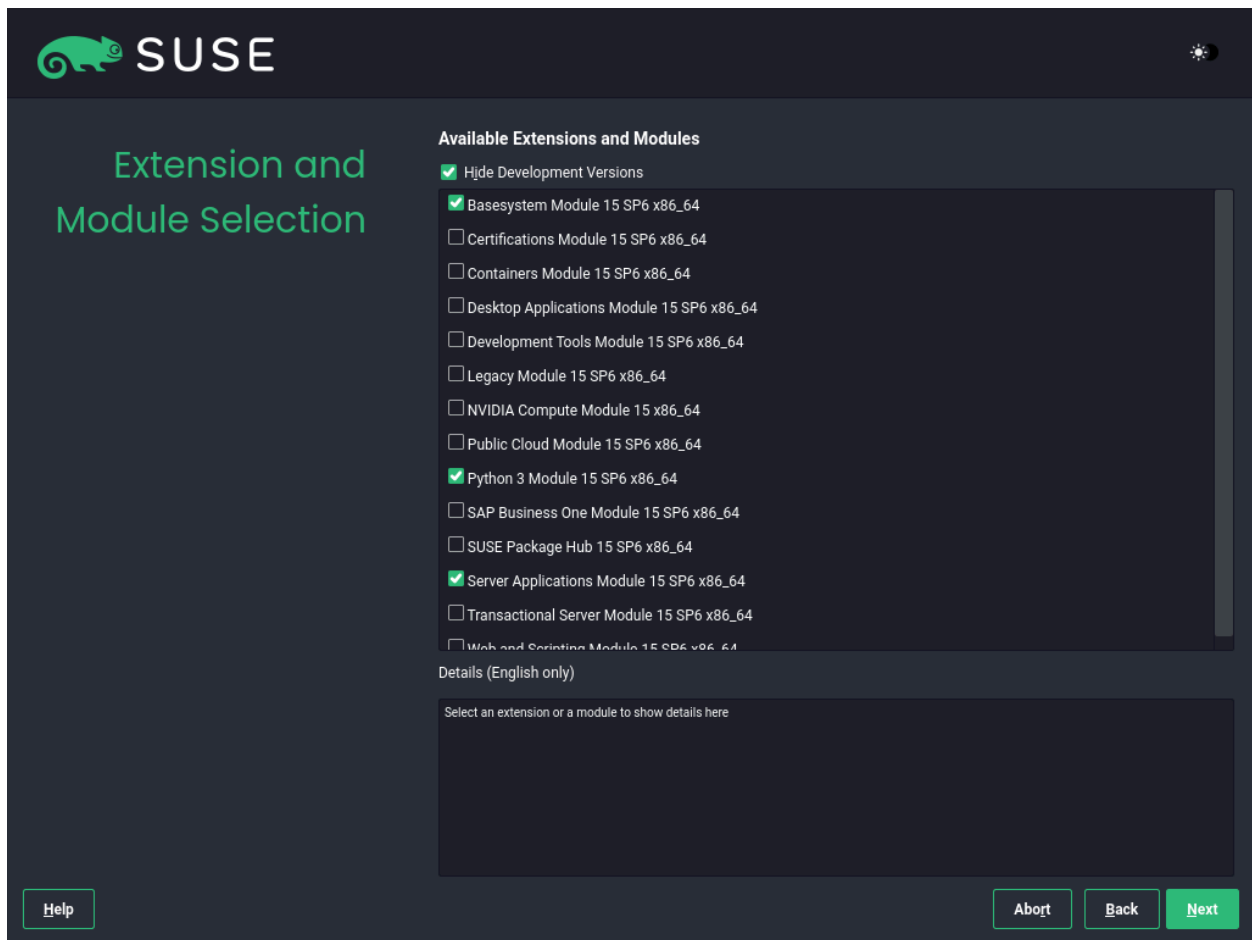


FIGURE 7: EXTENSION AND MODULE SELECTION

After the system is successfully registered, the installer lists modules and extensions that are available for SLES. Modules are components that allow you to customize the product according to your needs. They are included in your SLES subscription. Extensions add functionality to your product. They must be purchased separately.

The availability of certain modules or extensions depends on the product selected in the first step of the installation. For a description of the modules and their lifecycles, select a module to see the accompanying text. More detailed information is available in the [Modules and Extensions Quick Start \(https://documentation.suse.com/sles-15/html/SLES-all/article-modules.html\)](https://documentation.suse.com/sles-15/html/SLES-all/article-modules.html).

The selection of modules indirectly affects the scope of the installation, because it defines which software sources (repositories) are available for installation and in the running system.

The following modules and extensions are available for SUSE Linux Enterprise Server:

#### **Basesystem Module**

This module adds a basic system on top of the Unified Installer. It is required by all other modules and extensions. The scope of an installation that only contains the base system is comparable to the installation pattern *minimal system* of previous SLES versions. This module is selected for installation by default and should not be deselected.

*Dependencies:* None

#### **Certifications Module**

Contains the FIPS certification packages.

*Dependencies:* Server Applications

#### **Confidential Computing Technical Preview**

Contains packages related to confidential computing.

*Dependencies:* Basesystem

#### **Containers Module**

Contains support and tools for containers.

*Dependencies:* Basesystem

#### **Desktop Applications Module**

Adds a graphical user interface and essential desktop applications to the system.

*Dependencies:* Basesystem

#### **Development Tools Module**

Contains the compilers (including `gcc`) and libraries required for compiling and debugging applications. Replaces the former Software Development Kit (SDK).

*Dependencies:* Basesystem, Desktop Applications

#### **Legacy Module**

Helps you with migrating applications from earlier versions of SLES and other systems to SLES 15 SP6 by providing packages which are discontinued on SUSE Linux Enterprise. Packages in this module are selected based on the requirements for migration and the level of complexity of configuration.


This module is recommended when migrating from a previous product version.

*Dependencies:* Basesystem, Server Applications

#### **NVIDIA Compute Module**

Contains the NVIDIA CUDA (Compute Unified Device Architecture) drivers.



The software in this module is provided by NVIDIA under the [CUDA End User License Agreement](http://docs.nvidia.com/cuda/eula/) (<http://docs.nvidia.com/cuda/eula/>)  and is not supported by SUSE.

*Dependencies:* Basesystem

#### **Public Cloud Module**

Contains all tools required to create images for deploying SLES in cloud environments such as Amazon Web Services (AWS), Microsoft Azure, Google Compute Platform, or OpenStack.

*Dependencies:* Basesystem, Server Applications

#### **Python 3 Module**

This module contains the most recent versions of the selected Python 3 packages.

*Dependencies:* Basesystem

#### **SAP Business One Server**

This module contains packages and system configurations specific to SAP Business One Server. It is maintained and supported under the SUSE Linux Enterprise Server product subscription.

*Dependencies:* Basesystem, Server Applications, Desktop Applications, Development Tools

#### **Server Applications Module**

Adds server functionality by providing network services such as DHCP server, name server, or Web server.

*Dependencies:* Basesystem

#### **SUSE Linux Enterprise High Availability**

Adds clustering support for mission-critical setups to SLES. This extension requires a separate license key.

*Dependencies:* Basesystem, Server Applications

#### **SUSE Linux Enterprise Live Patching**

Adds support for performing critical patching without having to shut down the system. This extension requires a separate license key.


*Dependencies:* Basesystem, Server Applications

#### **SUSE Linux Enterprise Workstation Extension**

Extends the functionality of SLES with packages from SUSE Linux Enterprise Desktop, like additional desktop applications (office suite, e-mail client, graphical editor, etc.) and libraries. It allows combining both products to create a fully featured workstation. This extension requires a separate license key.

*Dependencies:* Basesystem, Desktop Applications

## SUSE Package Hub

Provides access to packages for SLES maintained by the openSUSE community. These packages are delivered without L3 support and do not interfere with the supportability of SLES. For more information, refer to <https://packagehub.suse.com/> .

*Dependencies:* Basesystem

## Transactional Server Module

Adds support for transactional updates. Updates are either applied to the system as a single transaction or not applied at all. This happens without influencing the running system. If an update fails, or if the successful update is deemed to be incompatible or otherwise incorrect, it can be discarded to immediately return the system to its previous functioning state.

*Dependencies:* Basesystem

## Web and Scripting Module

Contains packages intended for a running Web server.

*Dependencies:* Basesystem, Server Applications

Certain modules depend on the installation of other modules. Therefore, when selecting a module, other modules may be selected automatically to fulfill dependencies.

Depending on the product, the registration server can mark modules and extensions as recommended. Recommended modules and extensions are preselected for registration and installation. To avoid installing these recommendations, deselect them manually.

Select the modules and extensions you want to install and proceed with *Next*. In case you have chosen one or more extensions, you will be prompted to provide the respective registration codes. Depending on your choice, it may also be necessary to accept additional license agreements.



## Important: Default modules for offline installation

When performing an offline installation from the 15 SP6-Full-*ARCH*-GM-media1.iso, only the *Basesystem Module* is selected by default. To install the complete default package set of SUSE Linux Enterprise Server, additionally select the *Server Applications Module* and the *Python 3 Module*.

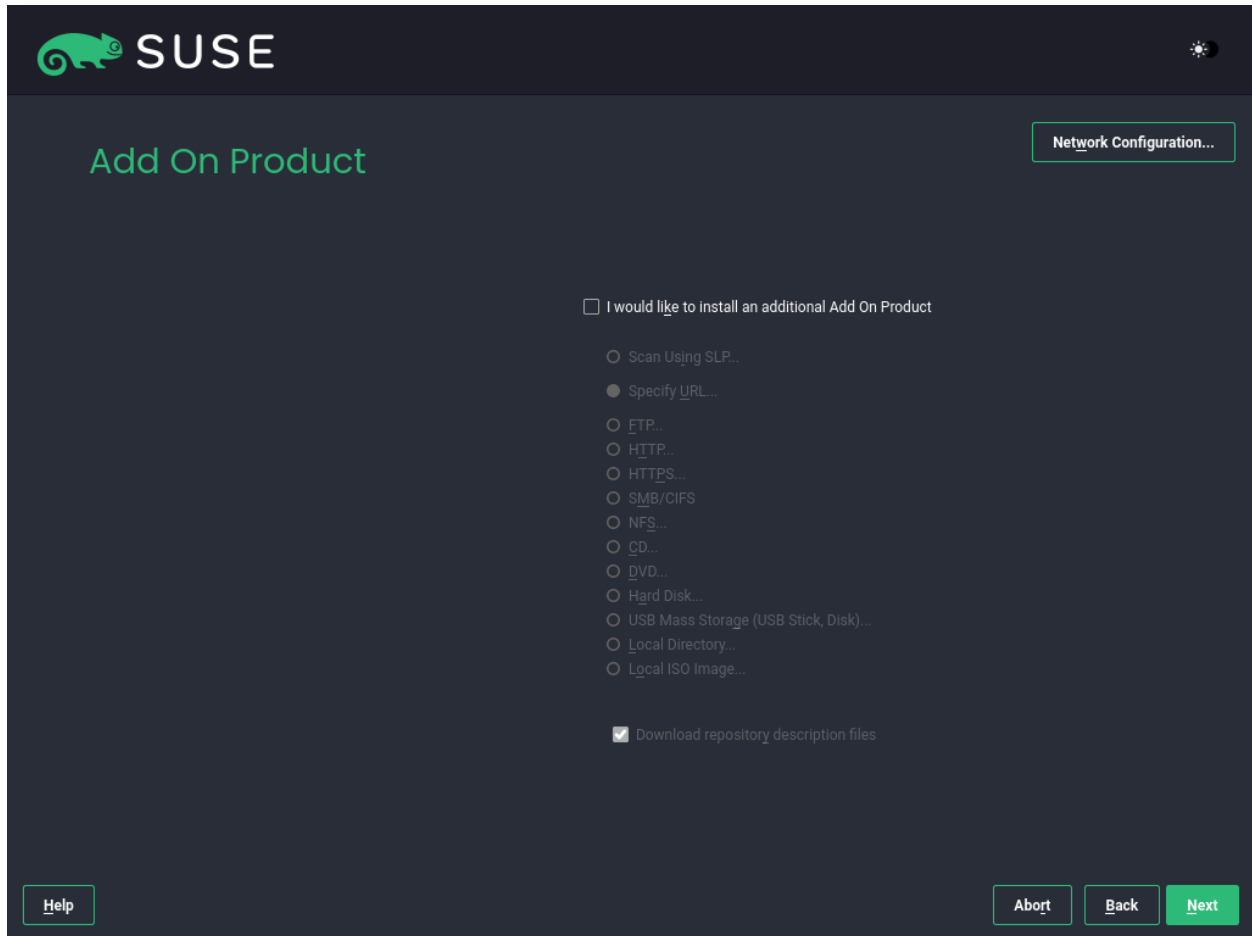



FIGURE 8: ADD-ON PRODUCT

The *Add-On Product* dialog allows you to add additional software sources (called “repositories”) to SLES that are not provided by the SUSE Customer Center. Add-on products may include third-party products and drivers as well as additional software for your system.



#### Tip: Adding drivers during the installation

You can also add driver update repositories via the *Add-On Product* dialog. Driver updates for SUSE Linux Enterprise are provided at <https://drivers.suse.com/> . These drivers have been created through the SUSE SolidDriver Program.

To skip this step, proceed with *Next*. Otherwise, activate *I would like to install an additional Add On Product*. Specify a media type, a local path, or a network resource hosting the repository and follow the on-screen instructions.

Check *Download Repository Description Files* to download the files describing the repository now. If deactivated, they will be downloaded after the installation has started. Proceed with *Next* and insert a medium if required. Depending on the content of the product, it may be necessary to accept additional license agreements. Proceed with *Next*. If you have chosen an add-on product requiring a registration key, you will be asked to enter it before proceeding to the next step.

### 3.1.3.7 System role

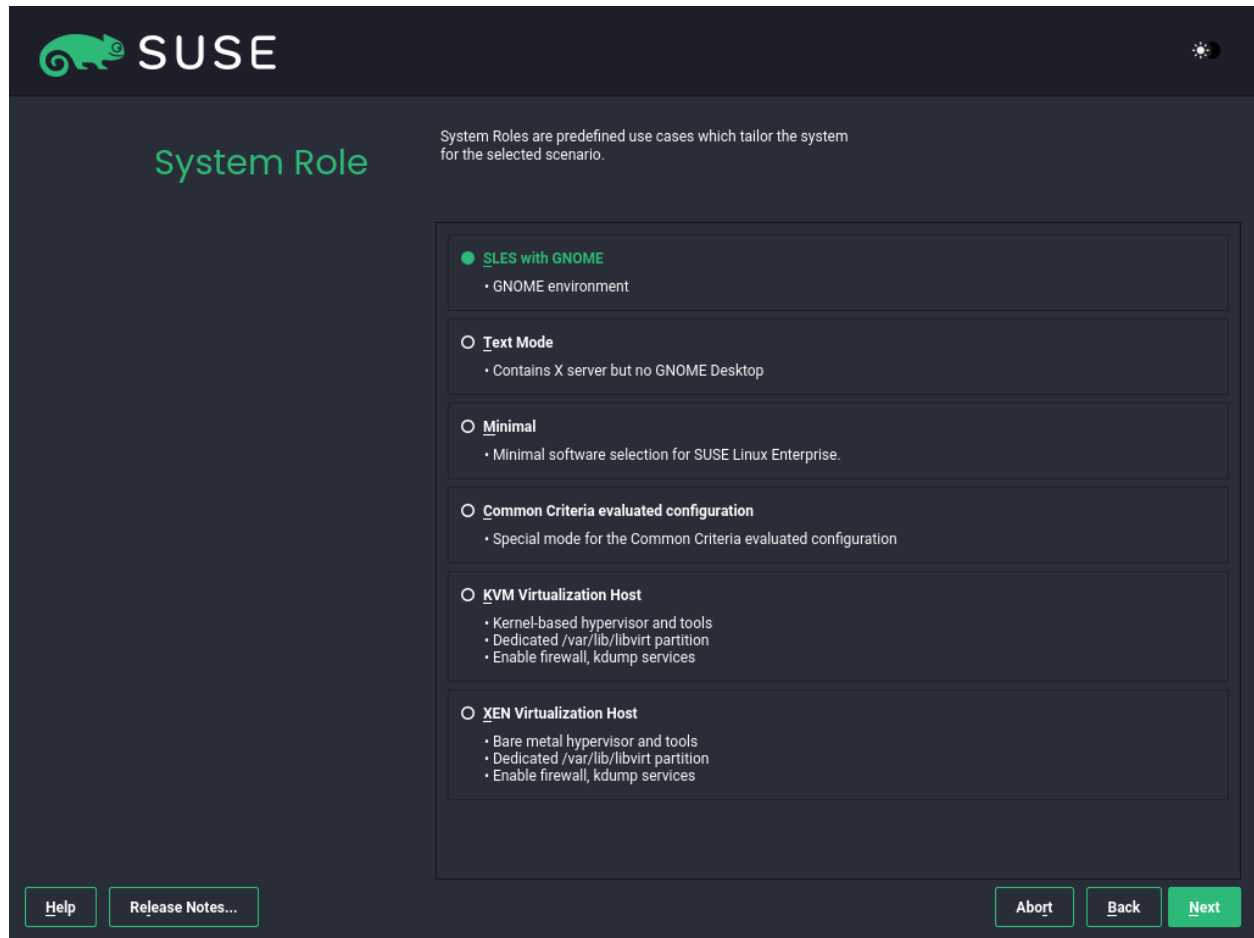


FIGURE 9: SYSTEM ROLE

The availability of system roles depends on your selection of modules and extensions. System roles define, for example, the set of software patterns that are preselected for the installation. Refer to the description on the screen to make your choice. Select a role and proceed with *Next*. If from the enabled modules only one role or no role is suitable for the respective base product, the *System Role* dialog is omitted.



## Tip: Release notes

From this point on, the Release Notes can be viewed from any screen during the installation process by selecting *Release Notes*.

### 3.1.3.8 Suggested partitioning

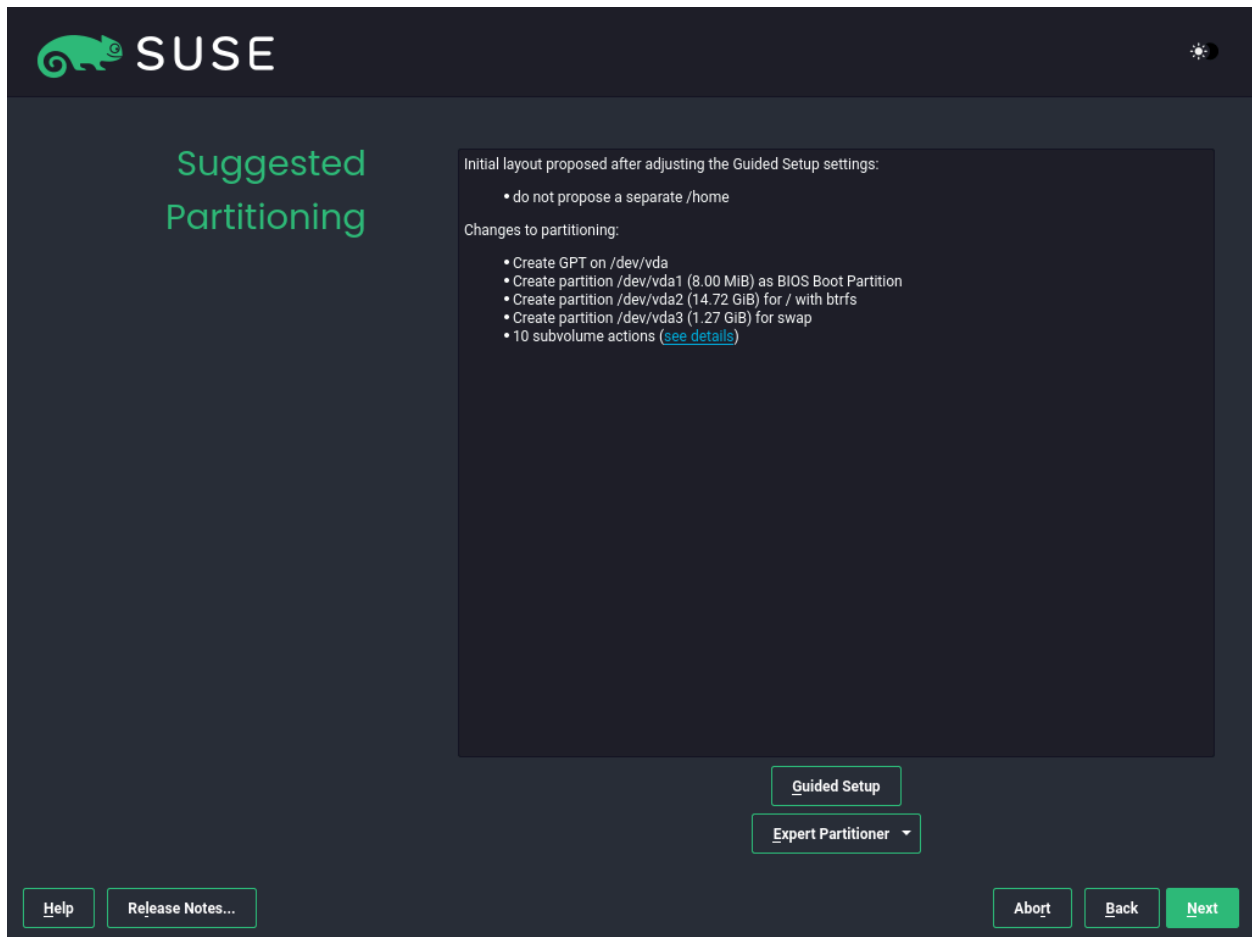


FIGURE 10: SUGGESTED PARTITIONING


Review the partition setup proposed by the system. If necessary, change it. You have the following options:

#### *Guided setup*

Starts a wizard that lets you refine the partitioning proposal. The options available here depend on your system setup. If it contains more than a single hard disk, you can choose which disk or disks to use and where to place the root partition. If the disks already contain partitions, decide whether to remove or resize them.

In subsequent steps, you may also add LVM support and disk encryption. You can change the file system for the root partition and decide whether or not to have a separate home partition.

### *Expert partitioner*

Opens the *Expert Partitioner*. This gives you full control over the partitioning setup and lets you create a custom setup. This option is intended for experts. For details, see the [Expert Partitioner \(https://documentation.suse.com/sles/15-SP6/html/SLES-all/cha-expert-partitioner.html#sec-expert-partitioner\)](https://documentation.suse.com/sles/15-SP6/html/SLES-all/cha-expert-partitioner.html#sec-expert-partitioner)  chapter.



### Warning: Disk space units

For partitioning purposes, disk space is measured in binary units rather than in decimal units. For example, if you enter sizes of 1GB, 1GiB or 1G, they all signify 1 GiB (Gibibyte), as opposed to 1 GB (Gigabyte).

#### Binary

1 GiB = 1 073 741 824 bytes.

#### Decimal

1 GB = 1 000 000 000 bytes.

#### Difference

1 GiB  $\approx$  1.07 GB.

To accept the proposed setup without any changes, choose *Next* to proceed.

### 3.1.3.9 Clock and time zone

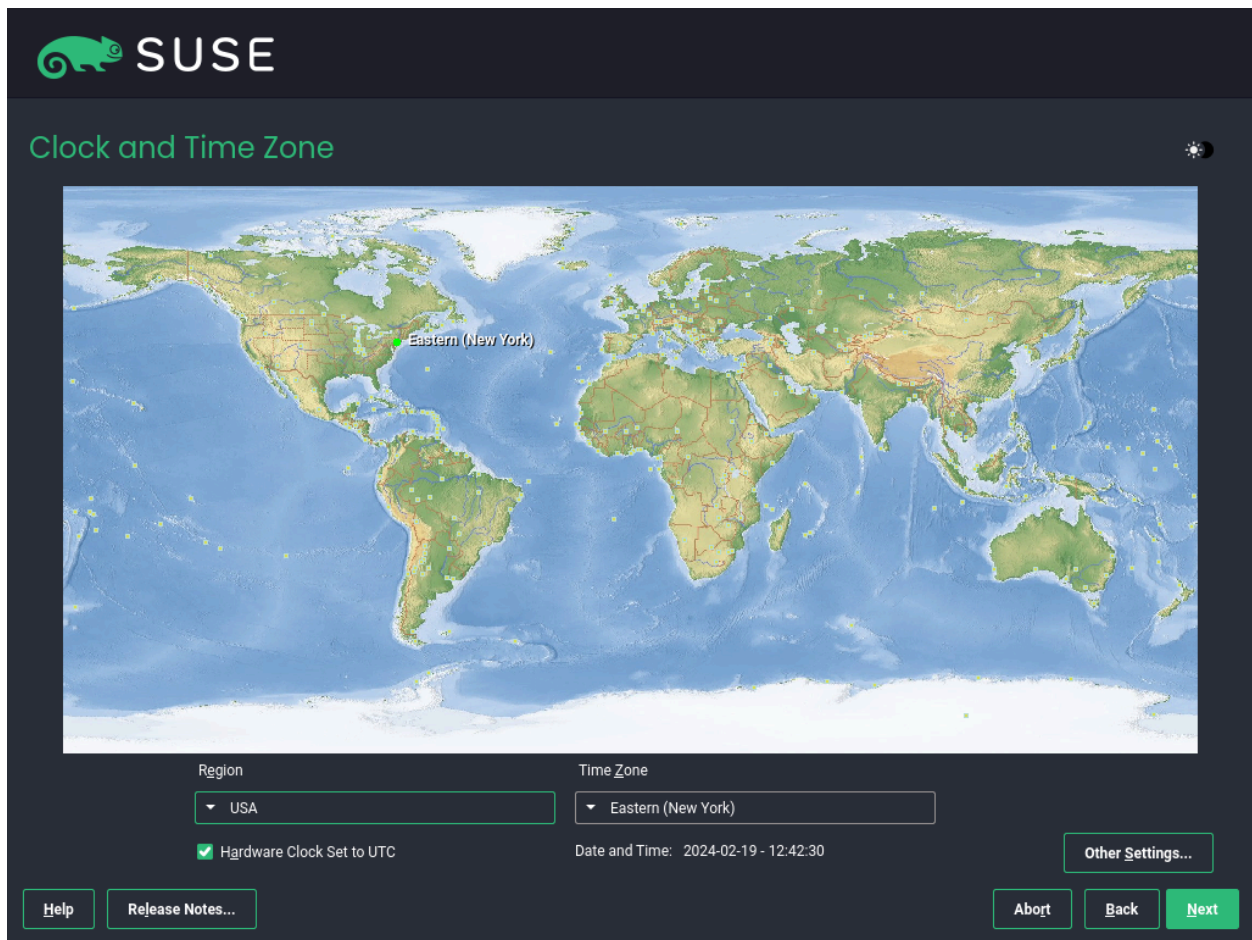


FIGURE 11: CLOCK AND TIME ZONE

Select the clock and time zone to use in your system. To manually adjust the time or to configure an NTP server for time synchronization, choose *Other Settings*. See the section on [Clock and Time Zone](https://documentation.suse.com/sles/15-SP6/html/SLES-all/cha-install.html#sec-yast-install-date-time) (<https://documentation.suse.com/sles/15-SP6/html/SLES-all/cha-install.html#sec-yast-install-date-time>) for detailed information. Proceed with *Next*.

**SUSE**

## Local User

☒ Create New User

User's Full Name

Username

Password

Confirm Password

☐ Use this password for system administrator

☐ Automatic Login

☐ Skip User Creation

Help Release Notes... Abort Back Next

FIGURE 12: LOCAL USER CREATION

To create a local user, type the first and last name in the *User's Full Name* field, the login name in the *Username* field, and the password in the *Password* field.

The password should be at least eight characters long and should contain both uppercase and lowercase letters and numbers. The maximum length for passwords is 72 characters, and passwords are case-sensitive.

For security reasons, it is also strongly recommended *not* to enable *Automatic Login*. You should also *not* *Use this Password for the System Administrator* but provide a separate root password in the next installation step.

If you install on a system where a previous Linux installation was found, you may *Import User Data from a Previous Installation*. Click *Choose User* for a list of available user accounts. Select one or more users.



In an environment where users are centrally managed (for example, by NIS or LDAP), you can skip the creation of local users. Select *Skip User Creation* in this case.

Proceed with *Next*.

### 3.1.3.11 Authentication for the system administrator “root”

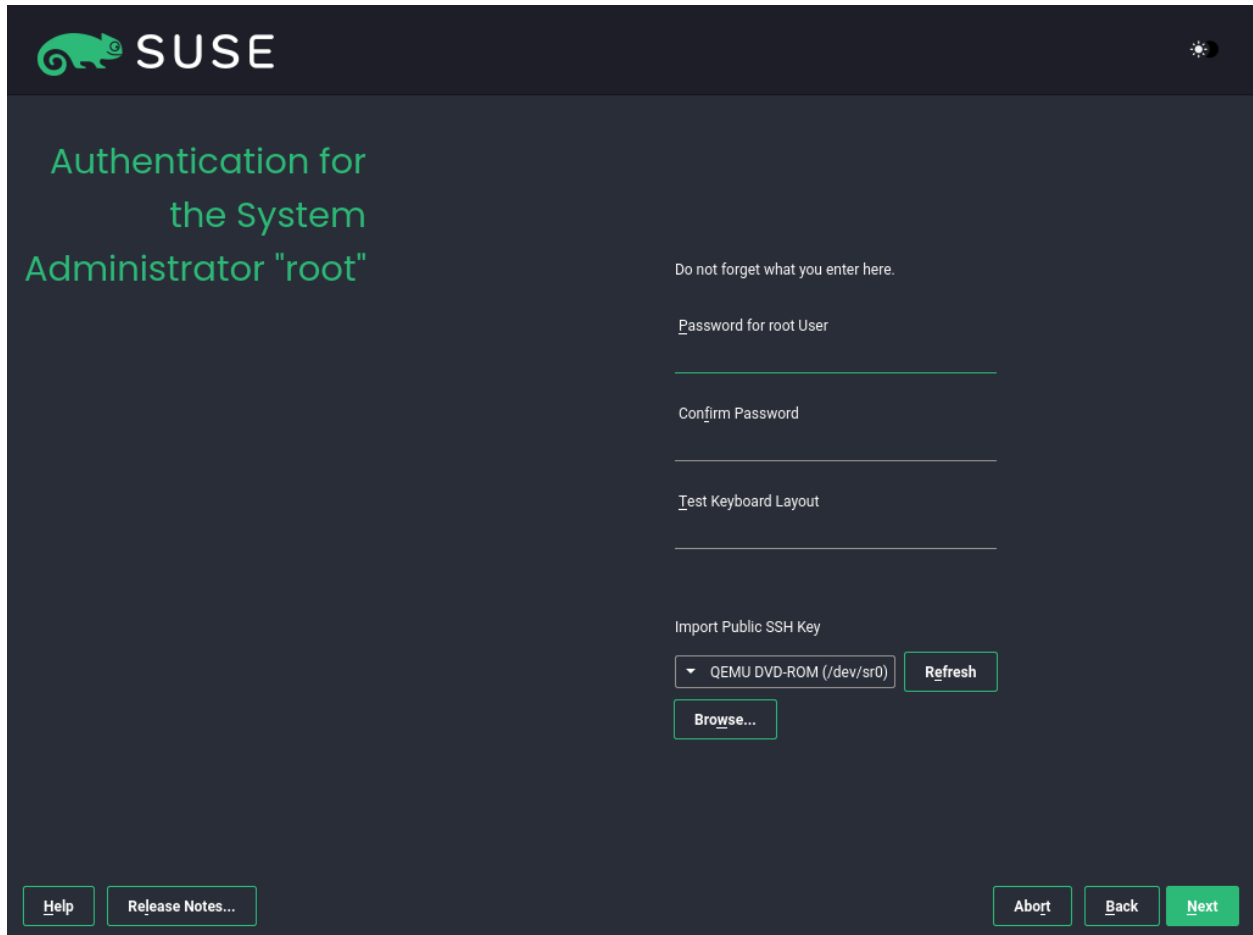
The screenshot shows the SUSE installation interface for setting the root user's password. The title is "Authentication for the System Administrator 'root'". It includes a warning "Do not forget what you enter here." and three input fields: "Password for root User", "Confirm Password", and "Test Keyboard Layout". Below these is an "Import Public SSH Key" section with a dropdown menu showing "QEMU DVD-ROM (/dev/sr0)", a "Refresh" button, and a "Browse..." button. At the bottom, there are "Help" and "Release Notes..." buttons on the left, and "Abort", "Back", and "Next" buttons on the right.

FIGURE 13: PASSWORD FOR THE SYSTEM ADMINISTRATOR “ROOT”

Type a password for the system administrator (called the root user) or provide a public SSH key. If you want, you can use both.

Because the root user is equipped with extensive permissions, the password should be chosen carefully. You should never forget the root password. After you entered it here, the password cannot be retrieved.



## Tip: Passwords and keyboard layout

It is recommended to use only US ASCII characters. In the event of a system error or when you need to start your system in rescue mode, the keyboard may not be localized.

To access the system remotely via SSH using a public key, import a key from removable media or an existing partition. See the section on [Authentication for the system administrator root](https://documentation.suse.com/sles/15-SP6/html/SLES-all/cha-install.html#sec-yast-install-user-root) (<https://documentation.suse.com/sles/15-SP6/html/SLES-all/cha-install.html#sec-yast-install-user-root>) for more information.

Proceed with *Next*.

### 3.1.3.12 Installation settings

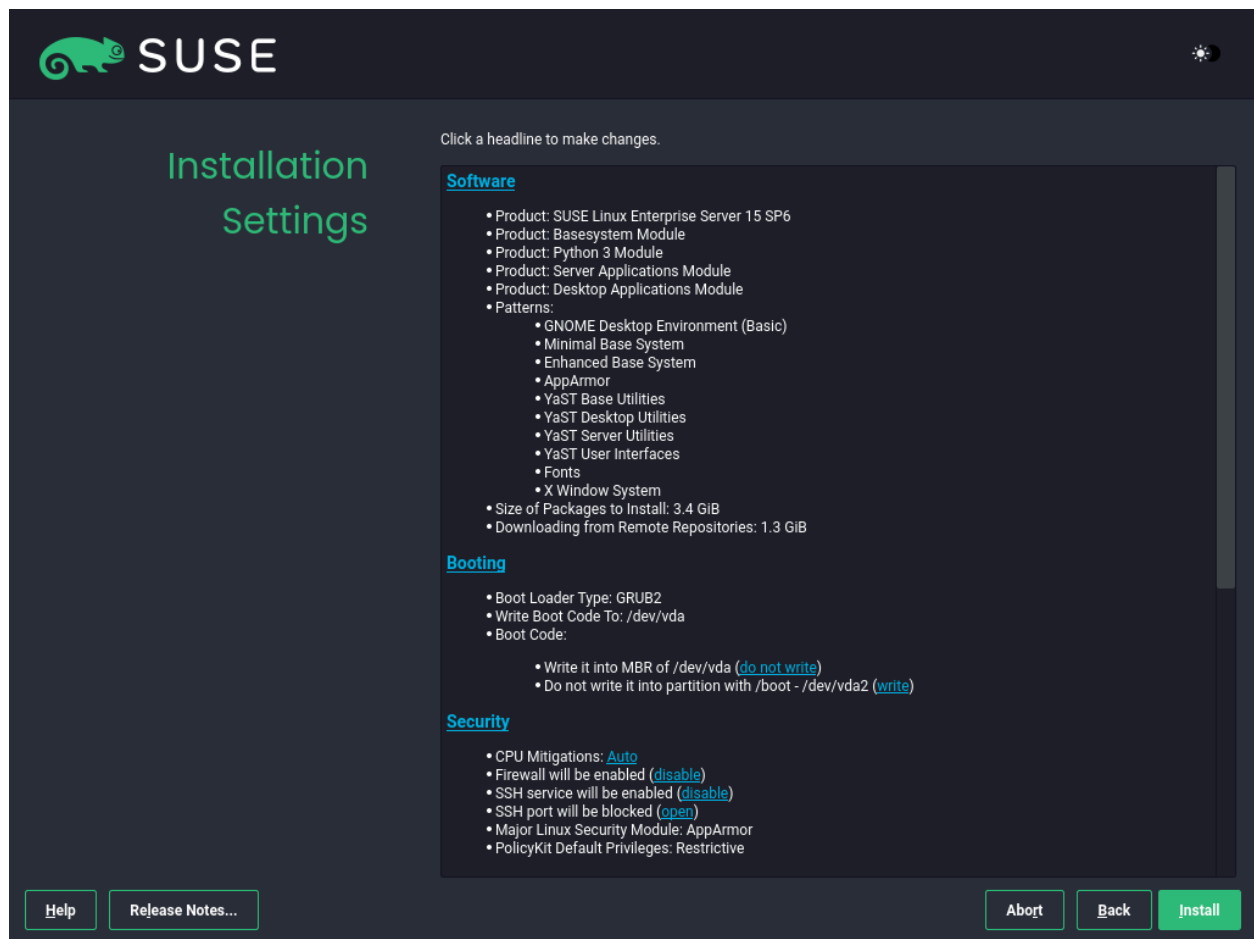


FIGURE 14: INSTALLATION SETTINGS


Use the *Installation Settings* screen to review and—if necessary—change several proposed installation settings. The current configuration is listed for each setting. To change it, click the headline. Certain settings, such as firewall or SSH, can be changed directly by clicking the respective links.

## Important: Remote access


Changes you can make here can also be made later at any time from the installed system. However, if you need remote access right after the installation, you may need to open the SSH port in the *Security* settings.

### *Software*



The scope of the installation is defined by the modules and extensions you have chosen for this installation. However, depending on your selection, not all packages available in a module are selected for installation.

Clicking *Software* opens the *Software Selection and System Tasks* screen, where you can change the software selection by selecting or deselecting patterns. Each pattern contains several software packages needed for specific functions (for example, *KVM Host Server*). For a more detailed selection based on software packages to install, select *Details* to switch to the YaST *Software Manager*. See [Installing or removing software \(https://documentation.suse.com/sles/15-SP6/html/SLES-all/cha-yast-software.html\)](https://documentation.suse.com/sles/15-SP6/html/SLES-all/cha-yast-software.html)  for more information.

### *Booting*

This section shows the boot loader configuration. Changing the defaults is recommended only if really needed. Refer to [The boot loader GRUB 2 \(https://documentation.suse.com/sles/15-SP6/html/SLES-all/cha-grub2.html\)](https://documentation.suse.com/sles/15-SP6/html/SLES-all/cha-grub2.html)  for details.

### *Security*

The *CPU Mitigations* refer to kernel boot command-line parameters for software mitigations that have been deployed to prevent CPU side-channel attacks. Click the selected entry to choose a different option. For details, see the section on [CPU Mitigations \(https://documentation.suse.com/sles/15-SP6/html/SLES-all/cha-grub2.html#vle-grub2-yast2-cpu-mitigations\)](https://documentation.suse.com/sles/15-SP6/html/SLES-all/cha-grub2.html#vle-grub2-yast2-cpu-mitigations) . By default, the *Firewall* is enabled on all configured network interfaces. To disable `firewalld`, click *disable* (not recommended). Refer to the [Masquerading and Firewalls \(https://documentation.suse.com/sles/15-SP6/html/SLES-all/cha-security-firewall.html\)](https://documentation.suse.com/sles/15-SP6/html/SLES-all/cha-security-firewall.html)  chapter for configuration details.



## Note: Firewall settings for receiving updates

By default, the firewall on SUSE AI only blocks incoming connections. If your system is behind another firewall that blocks outgoing traffic, make sure to allow connections to <https://scc.suse.com/> and <https://updates.suse.com> on ports 80 and 443 to receive updates.

The *SSH service* is enabled by default, but its port (22) is closed in the firewall. Click *open* to open the port or *disable* to disable the service. If SSH is disabled, remote logins will not be possible. Refer to [Securing network operations with OpenSSH \(https://documentation.suse.com/sles/15-SP6/html/SLES-all/cha-ssh.html\)](https://documentation.suse.com/sles/15-SP6/html/SLES-all/cha-ssh.html) for more information.

The default *Major Linux Security Module* is *AppArmor*. To disable it, select *None* as the module in the *Security* settings.

### Security Policies

Click to *enable* the [Defense Information Systems Agency STIG](https://documentation.suse.com/sles/15-SP6/html/SLES-all/cha-install.html#sec-yast-install-proposal-security-profile) security policy. If any installation settings are incompatible with the policy, you will be prompted to modify them accordingly. Certain settings can be adjusted automatically while others require user input. Enabling a security profile enables a full SCAP remediation on first boot. You can also perform a *scan only* or *do nothing* and manually remediate the system later with OpenSCAP. For more information, refer to the section on [Security Profiles \(https://documentation.suse.com/sles/15-SP6/html/SLES-all/cha-install.html#sec-yast-install-proposal-security-profile\)](https://documentation.suse.com/sles/15-SP6/html/SLES-all/cha-install.html#sec-yast-install-proposal-security-profile).

### Network configuration

Displays the current network configuration. By default, **wicked** is used for server installations and NetworkManager for desktop workloads. Click *Network Configuration* to change the settings. For details, see the section on [Configuring a network connection with YaST \(https://documentation.suse.com/sles/15-SP6/html/SLES-all/cha-network.html#sec-network-yast\)](https://documentation.suse.com/sles/15-SP6/html/SLES-all/cha-network.html#sec-network-yast).



## Important: Support for NetworkManager

SUSE only supports NetworkManager for desktop workloads with SLED or the Workstation extension. All server certifications are done with **wicked** as the network configuration tool, and using NetworkManager may invalidate them. NetworkManager is not supported by SUSE for server workloads.

### ***Kdump***

Kdump saves the memory image (“core dump”) to the file system in case the kernel crashes. This enables you to find the cause of the crash by debugging the dump file. Kdump is preconfigured and enabled by default. See the [Basic Kdump configuration \(https://documentation.suse.com/sles/15-SP6/html/SLES-all/cha-tuning-kexec.html#cha-tuning-kdump-basic\)](https://documentation.suse.com/sles/15-SP6/html/SLES-all/cha-tuning-kexec.html#cha-tuning-kdump-basic) for more information.

### ***Default systemd target***

If you have installed the desktop applications module, the system boots into the *graphical* target, with network, multi-user and display manager support. Switch to *multi-user* if you do not need to log in via a display manager.

### ***System***

View detailed hardware information by clicking *System*. In the resulting screen, you can also change *Kernel Settings*—see the section on [System Information \(https://documentation.suse.com/sles/15-SP6/html/SLES-all/cha-install.html#sec-yast-install-proposal-system\)](https://documentation.suse.com/sles/15-SP6/html/SLES-all/cha-install.html#sec-yast-install-proposal-system) for more information.

### 3.1.3.13 Start the installation

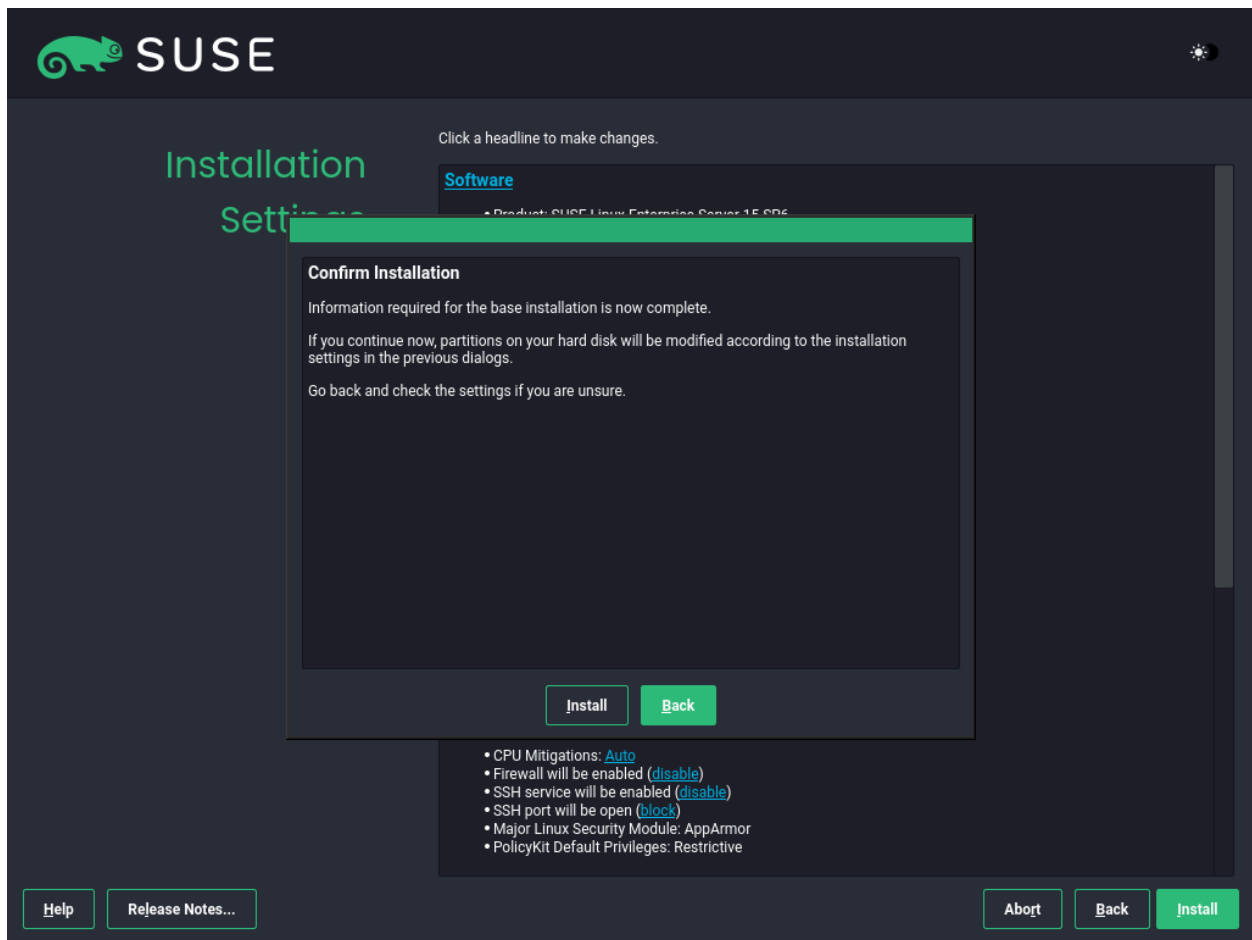


FIGURE 15: CONFIRM INSTALLATION

After you have finalized the system configuration on the *Installation Settings* screen, click *Install*. Depending on your software selection, you may need to agree to license agreements before the installation confirmation screen pops up. Up to this point, no changes have been made to your system. After you click *Install* a second time, the installation process starts.

### 3.1.3.14 The installation process

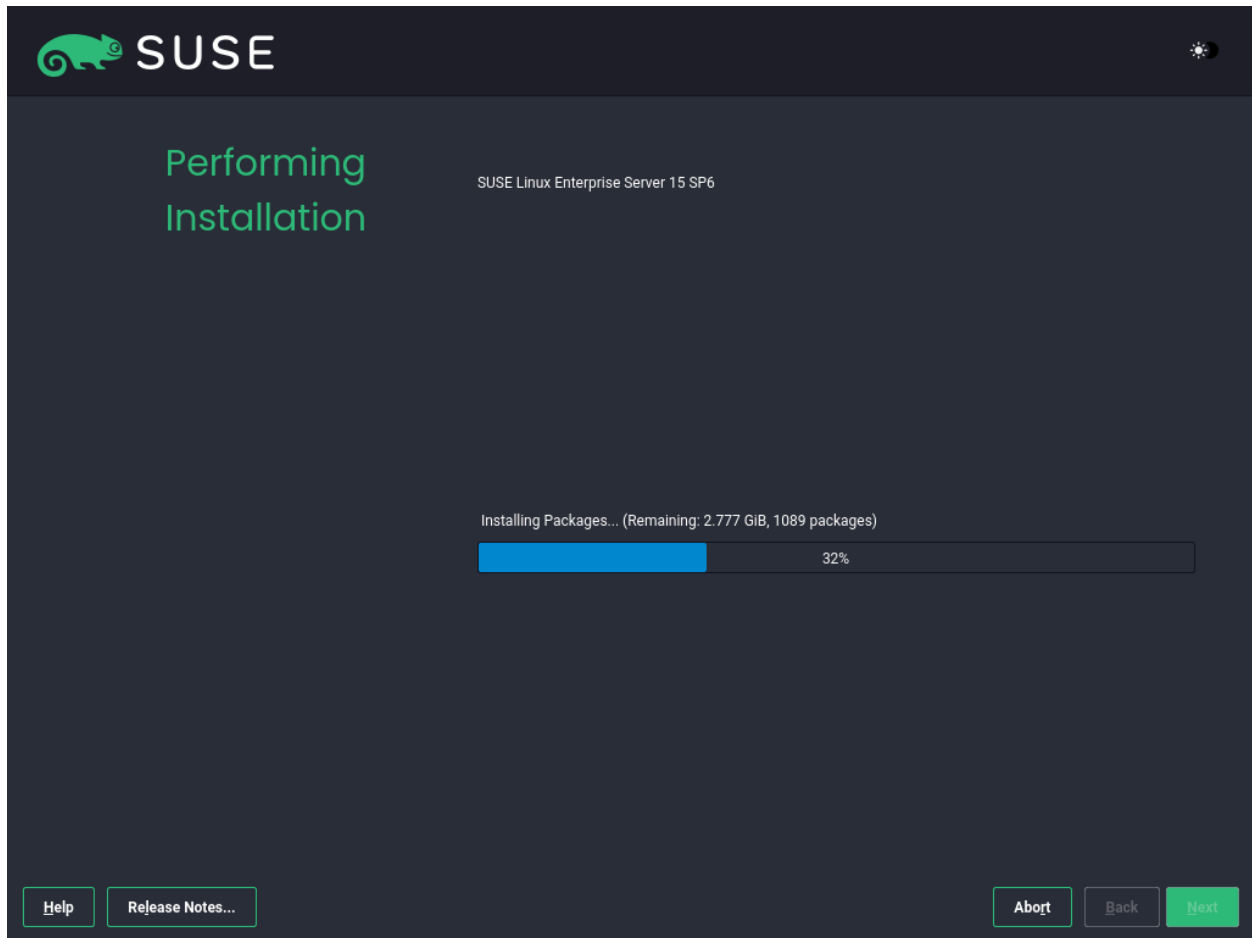


FIGURE 16: PERFORMING THE INSTALLATION

During the installation, the progress is shown. After the installation routine has finished, the computer is rebooted into the installed system.

## 3.2 Installing NVIDIA GPU drivers

This article demonstrates how to implement host-level NVIDIA GPU support via the open-driver. The open-driver is part of the core package repositories. Therefore, there is no need to compile it or download executable packages. This driver is built into the operating system rather than dynamically loaded by the NVIDIA GPU Operator. This configuration is desirable for customers who want to pre-build all artifacts required for deployment into the image, and where the dynamic selection of the driver version via Kubernetes is not a requirement.

## 3.2.1 Installing NVIDIA GPU drivers on SUSE Linux Enterprise Server

### 3.2.1.1 Requirements

If you are following this guide, it assumes that you have the following already available:

- At least one host with SLES 15 SP6 installed, physical or virtual.
- Your hosts are attached to a subscription as this is required for package access.
- A compatible NVIDIA GPU (<https://github.com/NVIDIA/open-gpu-kernel-modules#compatible-gpus>) installed or fully passed through to the virtual machine in which SLES is running.
- Access to the `root` user—these instructions assume you are the `root` user, and not escalating your privileges via `sudo`.

### 3.2.1.2 Considerations before the installation

#### 3.2.1.2.1 Select the driver generation

You must verify the driver generation for the NVIDIA GPU that your system has. For modern GPUs, the `G06` driver is the most common choice. Find more details in [the support database \(https://en.opensuse.org/SDB:NVIDIA\\_drivers#Install\)](https://en.opensuse.org/SDB:NVIDIA_drivers#Install).

This section details the installation of the `G06` generation of the driver.

#### 3.2.1.2.2 Additional NVIDIA components

Besides the NVIDIA open-driver provided by SUSE as part of SLES, you might also need additional NVIDIA components. These could include OpenGL libraries, CUDA toolkits, command-line utilities such as `nvidia-smi`, and container-integration components such as `nvidia-container-toolkit`. Many of these components are not shipped by SUSE as they are proprietary NVIDIA software. This section describes how to configure additional repositories that give you access to these components and provides examples of using these tools to achieve a fully functional system.



### 3.2.1.3 The installation procedure

1. On the *remote* host, run the script **SUSE-AI-mirror-nvidia.sh** from the air-gapped stack (see [Section 2.1, “Air-gapped stack”](#)) to download all required NVIDIA RPM packages to a local directory, for example:

```
> SUSE-AI-mirror-nvidia.sh \  
-p /LOCAL_MIRROR_DIRECTORY \  
-l https://nvidia.github.io/libnvidia-container/stable/rpm/x86_64 \  
https://developer.download.nvidia.com/compute/cuda/repos/sles15/x86_64/
```

After the download is complete, transfer the downloaded directory with all its content to each GPU-enabled *local* host.

2. On each GPU-enabled *local* host in its **transactional-update shell** session, add a package repository from the safely transferred NVIDIA RPM packages directory. This allows pulling in additional utilities, for example, `nvidia-smi`.

```
# zypper ar --no-gpgcheck \  
file:///LOCAL_MIRROR_DIRECTORY \  
nvidia-local-mirror  
transactional update # zypper --gpg-auto-import-keys refresh
```

3. Install the Open Kernel driver KMP and detect the driver version.

```
# zypper install -y --auto-agree-with-licenses \  
nv-prefer-signed-open-driver  
# version=$(rpm -qa --queryformat '%{VERSION}\n' \  
nv-prefer-signed-open-driver | cut -d "_" -f1 | sort -u | tail -n 1)
```

4. You can then install the appropriate packages for additional utilities that are useful for testing purposes.

```
# zypper install -y --auto-agree-with-licenses \  
nvidia-compute-utils-G06=${version} \  
nvidia-persistenced=${version}
```

5. Reboot the host to make the changes effective.

```
# reboot
```

6. Log back in and use the **nvidia-smi** tool to verify that the driver is loaded successfully and that it can both access and enumerate your GPUs.

```
# nvidia-smi
```

The output of this command should show you something similar to the following output. In the example below, the system has one GPU.

```
Fri Aug  1 15:32:10 2025
+-----+
+
| NVIDIA-SMI 580.82.07              Driver Version: 580.82.07      CUDA Version:
13.0   |
|-----+-----+
+-----+
| GPU  Name                Persistence-M | Bus-Id        Disp.A | Volatile
Uncorr. ECC |
| Fan  Temp   Perf         Pwr:Usage/Cap |      Memory-Usage | GPU-Util
Compute M.  |
|                                           |                   |
| MIG M. |
|=====+=====|
+=====+
|   0   Tesla T4                       On   | 00000000:00:1E.0 Off |
0   |
| N/A    33C    P8              13W / 70W |      0MiB / 15360MiB |      0%
Default |
|                                           |                   |
|   N/A   |
+-----+-----+
+-----+
+
| Processes:
|
| GPU  GI    CI             PID    Type    Process name                        GPU
Memory |
|      ID    ID                                     Usage
|
|
|=====+=====+
| No running processes found
|
+-----+
+
```

### 3.2.1.4 Validation of the driver installation

Running the **nvidia-smi** command has verified that, at the host level, the NVIDIA device can be accessed and that the drivers are loading successfully. To validate that it is functioning, you need to validate that the GPU can take instructions from a user-space application, ideally via a container and through the CUDA library, as that is typically what a real workload would use. For this, we can make a further modification to the host OS by installing **nvidia-container-toolkit**.

1. Install the **nvidia-container-toolkit** package from the NVIDIA Container Toolkit repository.

```
# zypper ar \
"https://nvidia.github.io/libnvidia-container/stable/rpm/"\
nvidia-container-toolkit.repo
# zypper --gpg-auto-import-keys install \
-y nvidia-container-toolkit
```

The **nvidia-container-toolkit.repo** file contains a stable repository **nvidia-container-toolkit** and an experimental repository **nvidia-container-toolkit-experimental**. Use the stable repository for production use. The experimental repository is disabled by default.

2. Verify that the system can successfully enumerate the devices using the NVIDIA Container Toolkit. The output should be verbose, with INFO and WARN messages, but no ERROR messages.

```
# nvidia-ctl cdi generate --output=/etc/cdi/nvidia.yaml
```

This ensures that any container started on the machine can employ discovered NVIDIA GPU devices.

3. You can then run a Podman-based container. Doing this via **podman** gives you a good way of validating access to the NVIDIA device from within a container, which should give confidence for doing the same with Kubernetes at a later stage.

Give Podman access to the labeled NVIDIA devices that were taken care of by the previous command and simply run the **bash** command.

```
# podman run --rm --device nvidia.com/gpu=all \
--security-opt=label=disable \
-it registry.suse.com/bci/bci-base:latest bash
```

You can now execute commands from within a temporary Podman container. It does not have access to your underlying system and is *ephemeral*—whatever you change in the container does not persist. Also, you cannot break anything on the underlying host.

4. Inside the container, install the required CUDA libraries. Identify their version from the output of the **nvidia-smi** command. From the above example, we are installing CUDA version 13.0 with many examples, demos and development kits to fully validate the GPU.

```
# zypper ar \
  http://developer.download.nvidia.com/compute/cuda/repos/sles15/x86_64/ \
  cuda-sle15-sp6
# zypper --gpg-auto-import-keys refresh
# zypper install -y cuda-libraries-13-0 cuda-demo-suite-12-9
```

5. Inside the container, run the **deviceQuery** CUDA example of the same version, which comprehensively validates GPU access via CUDA and from within the container itself.

```
# /usr/local/cuda-12.9/extras/demo_suite/deviceQuery Starting...

CUDA Device Query (Runtime API)

Detected 1 CUDA Capable device(s)

Device 0: "Tesla T4"
  CUDA Driver Version / Runtime Version      13.0/ 13.0
  CUDA Capability Major/Minor version number: 7.5
  Total amount of global memory:             14913 MBytes (15637086208 bytes)
  (40) Multiprocessors, ( 64) CUDA Cores/MP: 2560 CUDA Cores
  GPU Max Clock rate:                        1590 MHz (1.59 GHz)
  Memory Clock rate:                         5001 Mhz
  Memory Bus Width:                          256-bit
  L2 Cache Size:                             4194304 bytes
  Maximum Texture Dimension Size (x,y,z)     1D=(131072), 2D=(131072, 65536),
  3D=(16384, 16384, 16384)
  Maximum Layered 1D Texture Size, (num) layers 1D=(32768), 2048 layers
  Maximum Layered 2D Texture Size, (num) layers 2D=(32768, 32768), 2048 layers
  Total amount of constant memory:            65536 bytes
  Total amount of shared memory per block:    49152 bytes
  Total number of registers available per block: 65536
  Warp size:                                 32
  Maximum number of threads per multiprocessor: 1024
  Maximum number of threads per block:        1024
  Max dimension size of a thread block (x,y,z): (1024, 1024, 64)
  Max dimension size of a grid size (x,y,z): (2147483647, 65535, 65535)
  Maximum memory pitch:                      2147483647 bytes
```

```

Texture alignment:                    512 bytes
Concurrent copy and kernel execution: Yes with 3 copy engine(s)
Run time limit on kernels:            No
Integrated GPU sharing Host Memory:   No
Support host page-locked memory mapping: Yes
Alignment requirement for Surfaces:   Yes
Device has ECC support:                Enabled
Device supports Unified Addressing (UVA): Yes
Device supports Compute Preemption:   Yes
Supports Cooperative Kernel Launch:   Yes
Supports MultiDevice Co-op Kernel Launch: Yes
Device PCI Domain ID / Bus ID / location ID: 0 / 0 / 30
Compute Mode:
    < Default (multiple host threads can use ::cudaSetDevice() with device
    simultaneously) >

deviceQuery, CUDA Driver = CUDART, CUDA Driver Version = 13.0, CUDA Runtime Version
= 13.0, NumDevs = 1, Device0 = Tesla T4
Result = PASS

```

From inside the container, you can continue to run any other CUDA workload—such as compilers—to run further tests. When finished, you can exit the container.

```
# exit
```



## Important

Changes you have made in the container and packages you have installed inside will be lost and will not impact the underlying operating system.

## 3.2.2 Installing NVIDIA GPU drivers on SUSE Linux Micro

### 3.2.2.1 Requirements

If you are following this guide, it assumes that you have the following already available:

- At least one host with SUSE Linux Micro 6.1 installed, physical or virtual.
- Your hosts are attached to a subscription as this is required for package access.

- A compatible NVIDIA GPU (<https://github.com/NVIDIA/open-gpu-kernel-modules#compatible-gpus>) installed or fully passed through to the virtual machine in which SUSE Linux Micro is running.
- Access to the root user—these instructions assume you are the root user, and not escalating your privileges via sudo.

### 3.2.2.2 Considerations before the installation

#### 3.2.2.2.1 Select the driver generation

You must verify the driver generation for the NVIDIA GPU that your system has. For modern GPUs, the G06 driver is the most common choice. Find more details in [the support database \(https://en.opensuse.org/SDB:NVIDIA\\_drivers#Install\)](https://en.opensuse.org/SDB:NVIDIA_drivers#Install).

This section details the installation of the G06 generation of the driver.

#### 3.2.2.2.2 Additional NVIDIA components

Besides the NVIDIA open-driver provided by SUSE as part of SUSE Linux Micro, you might also need additional NVIDIA components. These could include OpenGL libraries, CUDA toolkits, command-line utilities such as nvidia-smi, and container-integration components such as nvidia-container-toolkit. Many of these components are not shipped by SUSE as they are proprietary NVIDIA software. This section describes how to configure additional repositories that give you access to these components and provides examples of using these tools to achieve a fully functional system.

#### 3.2.2.3 The installation procedure

1. On the *remote* host, run the script SUSE-AI-mirror-nvidia.sh from the air-gapped stack (see [Section 2.1, “Air-gapped stack”](#)) to download all required NVIDIA RPM packages to a local directory, for example:

```
> SUSE-AI-mirror-nvidia.sh \
  -p /LOCAL_MIRROR_DIRECTORY \
  -l https://nvidia.github.io/libnvidia-container/stable/rpm/x86_64 \
  https://developer.download.nvidia.com/compute/cuda/repos/sles15/x86_64/
```

After the download is complete, transfer the downloaded directory with all its content to each GPU-enabled *local* host.

2. On each *local* GPU-enabled host, open up a **transactional-update shell** session to create a new read/write snapshot of the underlying operating system so that we can make changes to the immutable platform.

```
# transactional-update shell
```

3. On each GPU-enabled *local* host in its **transactional-update shell** session, add a package repository from the safely transferred NVIDIA RPM packages directory. This allows pulling in additional utilities, for example, `nvidia-smi`.

```
transactional update # zypper ar --no-gpgcheck \  
file://LOCAL_MIRROR_DIRECTORY \  
nvidia-local-mirror  
transactional update # zypper --gpg-auto-import-keys refresh
```

4. Install the Open Kernel driver KMP and detect the driver version.

```
transactional update # zypper install -y --auto-agree-with-licenses \  
nvidia-open-driver-G06-signed-cuda-kmp-default  
transactional update # version=$(rpm -qa --queryformat '%{VERSION}\n' \  
nvidia-open-driver-G06-signed-cuda-kmp-default \  
| cut -d "_" -f1 | sort -u | tail -n 1)
```

5. You can then install the appropriate packages for additional utilities that are useful for testing purposes.

```
transactional update # zypper install -y --auto-agree-with-licenses \  
nvidia-compute-utils-G06=${version} \  
nvidia-persistenced=${version}
```

6. Exit the **transactional-update** session and reboot to the new snapshot that contains the changes you have made.

```
transactional update # exit  
# reboot
```

7. After the system has rebooted, log back in and use the `nvidia-smi` tool to verify that the driver is loaded successfully and that it can both access and enumerate your GPUs.

```
# nvidia-smi
```

The output of this command should show you something similar to the following output. In the example below, the system has one GPU.

```
Fri Aug 1 14:53:26 2025
+-----+
+
| NVIDIA-SMI 580.82.07                Driver Version: 580.82.07      CUDA Version:
13.0   |
|-----+-----|
+-----+
| GPU   Name                               Persistence-M | Bus-Id        Disp.A | Volatile
Uncorr. ECC |
| Fan  Temp  Perf              Pwr:Usage/Cap |      Memory-Usage | GPU-Util
Compute M. |
|
| MIG M. |
|=====+=====|
+=====+
|  0  Tesla T4                               On  |  00000000:00:1E.0 Off |
0 |
| N/A   34C    P8              10W /  70W |      0MiB / 15360MiB |      0%
Default |
|
|   N/A |
+-----+-----+
+-----+

+-----+
+
| Processes:
|
| GPU   GI    CI          PID    Type    Process name                  GPU
Memory |
|      ID    ID                 |                    Usage
|
|
|=====+=====|
| No running processes found
|
+-----+
+
```



### 3.2.2.4 Validation of the driver installation

Running the `nvidia-smi` command has verified that, at the host level, the NVIDIA device can be accessed and that the drivers are loading successfully. To validate that it is functioning, you need to validate that the GPU can take instructions from a user-space application, ideally via a container and through the CUDA library, as that is typically what a real workload would use. For this, we can make a further modification to the host OS by installing `nvidia-container-toolkit`.

1. Open another transactional-update shell.

```
# transactional-update shell
```

2. Install the `nvidia-container-toolkit` package from the NVIDIA Container Toolkit repository.

```
transactional update # zypper ar \
"https://nvidia.github.io/libnvidia-container/stable/rpm/"\
nvidia-container-toolkit.repo
transactional update # zypper --gpg-auto-import-keys install \
-y nvidia-container-toolkit
```

The `nvidia-container-toolkit.repo` file contains a stable repository `nvidia-container-toolkit` and an experimental repository `nvidia-container-toolkit-experimental`. Use the stable repository for production use. The experimental repository is disabled by default.

3. Exit the `transactional-update` session and reboot to the new snapshot that contains the changes you have made.

```
transactional update # exit
# reboot
```

4. Verify that the system can successfully enumerate the devices using the NVIDIA Container Toolkit. The output should be verbose, with INFO and WARN messages, but no ERROR messages.

```
# nvidia-ctl cdi generate --output=/etc/cdi/nvidia.yaml
```

This ensures that any container started on the machine can employ discovered NVIDIA GPU devices.

5. You can then run a Podman-based container. Doing this via **podman** gives you a good way of validating access to the NVIDIA device from within a container, which should give confidence for doing the same with Kubernetes at a later stage.

Give Podman access to the labeled NVIDIA devices that were taken care of by the previous command and simply run the **bash** command.

```
# podman run --rm --device nvidia.com/gpu=all \
  --security-opt=label=disable \
  -it registry.suse.com/bci/bci-base:latest bash
```

You can now execute commands from within a temporary Podman container. It does not have access to your underlying system and is *ephemeral*—whatever you change in the container does not persist. Also, you cannot break anything on the underlying host.

6. Inside the container, install the required CUDA libraries. Identify their version from the output of the **nvidia-smi** command. From the above example, we are installing CUDA version 13.0 with many examples, demos and development kits to fully validate the GPU.

```
# zypper ar \
  http://developer.download.nvidia.com/compute/cuda/repos/sles15/x86_64/ \
  cuda-sle15-sp6
# zypper --gpg-auto-import-keys refresh
# zypper install -y cuda-libraries-13-0 cuda-demo-suite-12-9
```

7. Inside the container, run the **deviceQuery** CUDA example of the same version, which comprehensively validates GPU access via CUDA and from within the container itself.

```
# /usr/local/cuda-12.9/extras/demo_suite/deviceQuery Starting...

CUDA Device Query (Runtime API)

Detected 1 CUDA Capable device(s)

Device 0: "Tesla T4"
  CUDA Driver Version / Runtime Version      13.0 / 13.0
  CUDA Capability Major/Minor version number: 7.5
  Total amount of global memory:             14914 MBytes (15638134784 bytes)
  (40) Multiprocessors, ( 64) CUDA Cores/MP: 2560 CUDA Cores
  GPU Max Clock rate:                        1590 MHz (1.59 GHz)
  Memory Clock rate:                         5001 Mhz
  Memory Bus Width:                          256-bit
  L2 Cache Size:                             4194304 bytes
  Maximum Texture Dimension Size (x,y,z)     1D=(131072), 2D=(131072, 65536),
  3D=(16384, 16384, 16384)
```

```

Maximum Layered 1D Texture Size, (num) layers 1D=(32768), 2048 layers
Maximum Layered 2D Texture Size, (num) layers 2D=(32768, 32768), 2048 layers
Total amount of constant memory: 65536 bytes
Total amount of shared memory per block: 49152 bytes
Total number of registers available per block: 65536
Warp size: 32
Maximum number of threads per multiprocessor: 1024
Maximum number of threads per block: 1024
Max dimension size of a thread block (x,y,z): (1024, 1024, 64)
Max dimension size of a grid size (x,y,z): (2147483647, 65535, 65535)
Maximum memory pitch: 2147483647 bytes
Texture alignment: 512 bytes
Concurrent copy and kernel execution: Yes with 3 copy engine(s)
Run time limit on kernels: No
Integrated GPU sharing Host Memory: No
Support host page-locked memory mapping: Yes
Alignment requirement for Surfaces: Yes
Device has ECC support: Enabled
Device supports Unified Addressing (UVA): Yes
Device supports Compute Preemption: Yes
Supports Cooperative Kernel Launch: Yes
Supports MultiDevice Co-op Kernel Launch: Yes
Device PCI Domain ID / Bus ID / location ID: 0 / 0 / 30
Compute Mode:
    < Default (multiple host threads can use ::cudaSetDevice() with device
    simultaneously) >

```

```

deviceQuery, CUDA Driver = CUDART, CUDA Driver Version = 13.0, CUDA Runtime Version
= 13.0, NumDevs = 1, Device0 = Tesla T4
Result = PASS

```

From inside the container, you can continue to run any other CUDA workload—such as compilers—to run further tests. When finished, you can exit the container.

```
# exit
```



## Important

Changes you have made in the container and packages you have installed inside will be lost and will not impact the underlying operating system.

## 4 Preparing the cluster for AI Library

This procedure assumes that you already have the base operating system installed on cluster nodes as well as the SUSE Rancher Prime: RKE2 Kubernetes distribution installed and operational. If you are installing from scratch, refer to [Section 3, “Installing the Linux and Kubernetes distribution”](#) first.

1. Install SUSE Rancher Prime (<https://documentation.suse.com/cloudnative/rancher-manager/latest/en/installation-and-upgrade/install-rancher.html>) [↗](#).
2. Install the NVIDIA GPU Operator on the cluster.



### Tip: Installing NVIDIA GPU Operator on SUSE Rancher Prime: RKE2

If you run SUSE Rancher Prime: RKE2, follow these steps:

1. On the agent nodes, run the following command:

```
# echo PATH=$PATH:/usr/local/nvidia/toolkit >> /etc/default/rke2-agent
```

2. On the server nodes, run the following command:

```
# echo PATH=$PATH:/usr/local/nvidia/toolkit >> /etc/default/rke2-server
```

3. Follow the steps described in [https://documentation.suse.com/cloudnative/rke2/latest/en/advanced.html#\\_deploy\\_nvidia\\_operator](https://documentation.suse.com/cloudnative/rke2/latest/en/advanced.html#_deploy_nvidia_operator) [↗](#).

3. Connect the Kubernetes cluster to SUSE Rancher Prime. Refer to <https://documentation.suse.com/cloudnative/rancher-manager/latest/en/cluster-deployment/register-existing-clusters.html> [↗](#) for details.
4. Configure the GPU-enabled nodes so that the SUSE AI containers are assigned to Pods that run on nodes equipped with NVIDIA GPU hardware. Find more details about assigning Pods to nodes in [Section 4.1, “Assigning GPU nodes to applications”](#).
5. Install and configure SUSE Observability to observe the nodes used for SUSE AI application. Refer to [Section 4.2, “Setting up SUSE Observability for SUSE AI”](#) for more details.

## 4.1 Assigning GPU nodes to applications

When deploying a containerized application to Kubernetes, you need to ensure that containers requiring GPU resources are run on appropriate worker nodes. For example, Ollama, a core component of SUSE AI, can deeply benefit from the use of GPU acceleration. This topic describes how to satisfy this requirement by explicitly requesting GPU resources and labeling worker nodes for configuring the node selector.

### REQUIREMENTS

- Kubernetes cluster—such as SUSE Rancher Prime: RKE2—must be available and configured with more than one worker node in which certain nodes have NVIDIA GPU resources and others do not.
- This document assumes that any kind of deployment to the Kubernetes cluster is done using Helm charts.

### 4.1.1 Labeling GPU nodes

To distinguish nodes with the GPU support from non-GPU nodes, Kubernetes uses *labels*. Labels are used for relevant metadata and should not be confused with annotations that provide simple information about a resource. It is possible to manipulate labels with the **kubectl** command, as well as by tweaking configuration files from the nodes. If an IaC tool such as Terraform is used, labels can be inserted in the node resource configuration files.

To label a single node, use the following command:

```
> kubectl label node GPU_NODE_NAME accelerator=nvidia-gpu
```

To achieve the same result by tweaking the `node.yaml` node configuration, add the following content and apply the changes with **kubectl apply -f node.yaml**:

```
apiVersion: v1
kind: Node
metadata:
  name: node-name
  labels:
    accelerator: nvidia-gpu
```



## Tip: Labeling multiple nodes

To label multiple nodes, use the following command:

```
> kubectl label node \
  GPU_NODE_NAME1 \
  GPU_NODE_NAME2 ... \
  accelerator=nvidia-gpu
```



## Tip

If Terraform is being used as an IaC tool, you can add labels to a group of nodes by editing the `.tf` files and adding the following values to a resource:

```
resource "node_group" "example" {
  labels = {
    "accelerator" = "nvidia-gpu"
  }
}
```

To check if the labels are correctly applied, use the following command:

```
> kubectl get nodes --show-labels
```

### 4.1.2 Assigning GPU nodes

The matching between a container and a node is configured by the explicit resource allocation and the use of labels and node selectors. The use cases described below focus on NVIDIA GPUs.

#### 4.1.2.1 Enable GPU passthrough

Containers are isolated from the host environment by default. For the containers that rely on the allocation of GPU resources, their Helm charts must enable GPU passthrough so that the container can access and use the GPU resource. Without enabling the GPU passthrough, the container may still run, but it can only use the main CPU for all computations. Refer to [Ollama Helm chart \(https://documentation.suse.com/suse-ai/1.0/html/AI-deployment-intro/index.html#ollama-helmchart\)](https://documentation.suse.com/suse-ai/1.0/html/AI-deployment-intro/index.html#ollama-helmchart) for an example of the configuration required for GPU acceleration.

#### 4.1.2.2 Assignment by resource request

After the NVIDIA GPU Operator is configured on a node, you can instantiate applications requesting the resource `nvidia.com/gpu` provided by the operator. Add the following content to your `values.yaml` file. Specify the number of GPUs according to your setup.

```
resources:
  requests:
    nvidia.com/gpu: 1
  limits:
    nvidia.com/gpu: 1
```

#### 4.1.2.3 Assignment by labels and node selectors

If affected cluster nodes are labeled with a label such as `accelerator=nvidia-gpu`, you can configure the node selector to check for the label. In this case, use the following values in your `values.yaml` file.

```
nodeSelector:
  accelerator: nvidia-gpu
```

### 4.1.3 Verifying Ollama GPU assignment

If the GPU is correctly detected, the Ollama container logs this event:

```
| [...] source=routes.go:1172 msg="Listening on :11434 (version 0.0.0)"
|
| [...] source=payload.go:30 msg="extracting embedded files" dir=/tmp/ollama2502346830/
runners
|
| [...] source=payload.go:44 msg="Dynamic LLM libraries [cuda_v12 cpu cpu_avx cpu_avx2]"
|
| [...] source=gpu.go:204 msg="looking for compatible GPUs"
|
| [...] source=types.go:105 msg="inference compute" id=GPU-c9ad37d0-d304-5d2a-c2e6-
d3788cd733a7 library=cuda compute |
```

## 4.2 Setting up SUSE Observability for SUSE AI

SUSE Observability provides comprehensive monitoring and insights into your infrastructure and applications. It enables efficient tracking of metrics, logs and traces, helping you maintain optimal performance and troubleshoot issues effectively. This procedure guides you through setting up SUSE Observability for the SUSE AI environment using the SUSE AI Observability Extension.

### 4.2.1 Deployment scenarios

You can deploy SUSE Observability and SUSE AI in two different ways:

- **Single-Cluster setup:** Both SUSE AI and SUSE Observability are installed in the same Kubernetes cluster. This is a simpler approach ideal for testing and proof-of-concept deployments. Communication between components can use internal cluster DNS.
- **Multi-Cluster setup:** SUSE AI and SUSE Observability are installed on separate, dedicated Kubernetes clusters. This setup is recommended for production environments because it isolates workloads. Communication requires exposing the SUSE Observability endpoints externally, for example, via an Ingress.

This section provides instructions for both scenarios.

### 4.2.2 Requirements

To set up SUSE Observability for SUSE AI, you need to meet the following requirements:

- Have access to SUSE Application Collection
- Have a valid SUSE AI subscription
- Have a valid license for SUSE Observability in SUSE Customer Center
- Instrument your applications for telemetry data acquisition with OpenTelemetry.

For details on how to collect traces and metrics from SUSE AI components and user-developed applications, refer to [Monitoring SUSE AI with OpenTelemetry and SUSE Observability \(https://documentation.suse.com/suse-ai/1.0/html/AI-monitoring/index.html\)](https://documentation.suse.com/suse-ai/1.0/html/AI-monitoring/index.html). It includes configurations that are essential for full observability.



## ! Important: SUSE Application Collection not instrumented by default

Applications from the SUSE Application Collection are not instrumented by default. If you want to monitor your AI applications, you need to follow the instrumentation guidelines that we provide in the document [Monitoring SUSE AI with OpenTelemetry and SUSE Observability](https://documentation.suse.com/suse-ai/1.0/html/AI-monitoring/index.html) (<https://documentation.suse.com/suse-ai/1.0/html/AI-monitoring/index.html>) [↗](#).

### 4.2.3 Setup process overview

The following chart shows the high-level steps for the setup procedure. You will first set up the SUSE Observability cluster, then configure the SUSE AI cluster, and finally instrument your applications. Execute the steps in each column from left to right and top to bottom.

- *Blue steps* are related to Helm chart installations.
- *Gray steps* represent another type of interaction, such as coding.

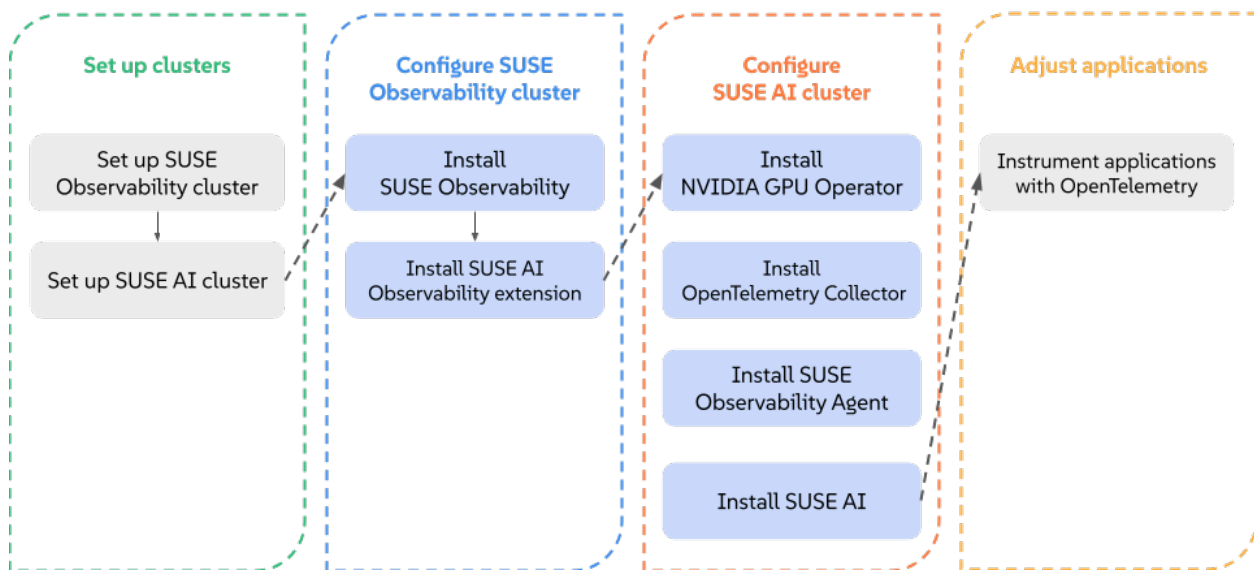


FIGURE 17: HIGH-LEVEL OVERVIEW OF THE SUSE OBSERVABILITY SETUP



## Tip: Setup clusters

You can create and configure Kubernetes clusters for SUSE AI and SUSE Observability as you prefer. If you are using SUSE Rancher Prime, check its [documentation](https://ranchermanager.docs.rancher.com/how-to-guides/new-user-guides/kubernetes-clusters-in-rancher-setup) (<https://ranchermanager.docs.rancher.com/how-to-guides/new-user-guides/kubernetes-clusters-in-rancher-setup>). For testing purposes, you can even share one cluster for both deployments. You can skip instructions on setting up a specific cluster if you already have one configured.

The diagram below shows the result of the above steps. There are two clusters represented, one for the SUSE Observability workload and another one for SUSE AI. You may use identical setup or customize it for your environment.

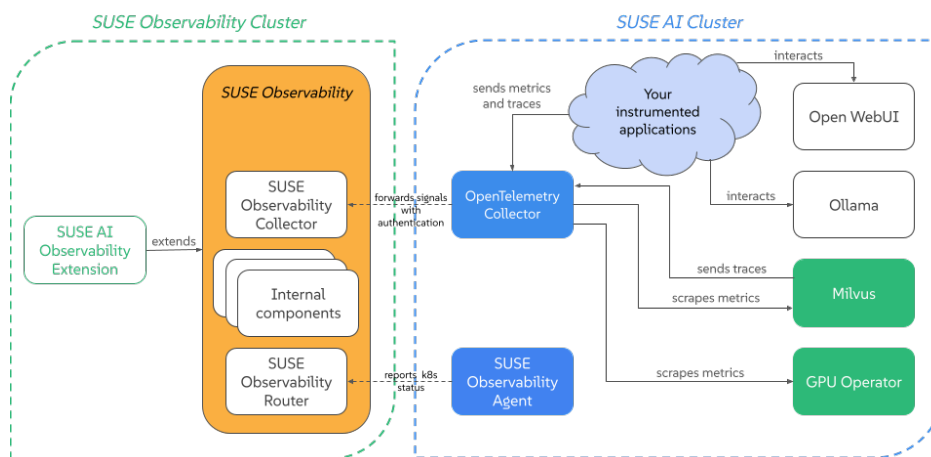


FIGURE 18: SEPARATE CLUSTERS FOR SUSE AI AND SUSE OBSERVABILITY

## POINTS TO NOTICE

- You can install SUSE AI Observability Extension alongside SUSE Observability. It means that you can confidently use the internal Kubernetes DNS.
- SUSE Observability contains several components and the following two of them need to be accessible by the AI Cluster:
  - The Collector endpoint. Refer to [Exposing SUSE Observability outside of the cluster](https://documentation.suse.com/cloudnative/suse-observability/next/en/setup/install-stackstate/kubernetes_openshift/ingress.html) ([https://documentation.suse.com/cloudnative/suse-observability/next/en/setup/install-stackstate/kubernetes\\_openshift/ingress.html](https://documentation.suse.com/cloudnative/suse-observability/next/en/setup/install-stackstate/kubernetes_openshift/ingress.html)) or [Self-hosted SUSE Observability](https://documentation.suse.com/cloudnative/suse-observability/next/en/setup/otel/otlp-apis.html#_self_hosted_suse_observability) ([https://documentation.suse.com/cloudnative/suse-observability/next/en/setup/otel/otlp-apis.html#\\_self\\_hosted\\_suse\\_observability](https://documentation.suse.com/cloudnative/suse-observability/next/en/setup/otel/otlp-apis.html#_self_hosted_suse_observability)) for details about exposing it.
  - The SUSE Observability API. Refer to [Exposing SUSE Observability outside of the cluster](https://documentation.suse.com/cloudnative/suse-observability/next/en/setup/install-stackstate/kubernetes_openshift/ingress.html) ([https://documentation.suse.com/cloudnative/suse-observability/next/en/setup/install-stackstate/kubernetes\\_openshift/ingress.html](https://documentation.suse.com/cloudnative/suse-observability/next/en/setup/install-stackstate/kubernetes_openshift/ingress.html)) for details about exposing it.
- Milvus metrics and traces can be scraped by the OpenTelemetry Collector with simple configurations, provided below. The same is true for GPU metrics.
- To get information from Open WebUI or Ollama, you must have a specific instrumentation set. It can be an application instrumented with the OpenLIT SDK or other form of instrumentation following the same patterns.



### Important

Remember that in multi-cluster setups, it is **critical** to properly expose your endpoints. Configure TLS, be careful with the configuration, and make sure to provide the right keys and tokens. More details are provided in the respective instructions.

## 4.2.4 Setting up the SUSE Observability cluster

This initial step is identical for both single-cluster and multi-cluster deployments.

1. **Install SUSE Observability.** You can follow the official [SUSE Observability air-gapped installation documentation](https://documentation.suse.com/cloudnative/suse-observability/latest/en/k8s-suse-rancher-prime-air-gapped.html) (<https://documentation.suse.com/cloudnative/suse-observability/latest/en/k8s-suse-rancher-prime-air-gapped.html>) for all installation instructions. Re-

member to expose your APIs ([https://documentation.suse.com/cloudnative/suse-observability/latest/en/setup/install-stackstate/kubernetes\\_openshift/ingress.html](https://documentation.suse.com/cloudnative/suse-observability/latest/en/setup/install-stackstate/kubernetes_openshift/ingress.html)) and collector endpoints to your SUSE AI cluster.

## ! Important: Multi-cluster setup

For multi-cluster setups, you must expose the SUSE Observability API and collector endpoints so that the SUSE AI cluster can reach them. Refer to the guide on [exposing SUSE Observability outside of the cluster](https://documentation.suse.com/cloudnative/suse-observability/latest/en/setup/install-stackstate/kubernetes_openshift/ingress.html) ([https://documentation.suse.com/cloudnative/suse-observability/latest/en/setup/install-stackstate/kubernetes\\_openshift/ingress.html](https://documentation.suse.com/cloudnative/suse-observability/latest/en/setup/install-stackstate/kubernetes_openshift/ingress.html)).

2. **Install the SUSE Observability extension.** Create a new Helm values file named `genai_values.yaml`. Before creating the file, review the placeholders below.

### SUSE\_OBSERVABILITY\_API\_URL

The URL of the SUSE Observability API. For multi-cluster deployments, this is the external URL. For single-cluster deployments, this can be the internal service URL. Example: `http://suse-observability-api.your-domain.com`

### SUSE\_OBSERVABILITY\_API\_KEY

The API key from the `baseConfig_values.yaml` file used during the SUSE Observability installation.

### SUSE\_OBSERVABILITY\_API\_TOKEN\_TYPE

Can be `api` for a token from the Web UI or `service` for a Service Token.

### SUSE\_OBSERVABILITY\_TOKEN

The API or Service token itself.

### OBSERVED\_SERVER\_NAME

The name of the cluster to observe. It must match the name used in the Kubernetes StackPack configuration. Example: `suse-ai-cluster`.

Create the `genai_values.yaml` file with the following content:

```
global:
  imagePullSecrets:
    - application-collection ❶
  imageRegistry: LOCAL_DOCKER_REGISTRY_URL:5043
  serverUrl: SUSE_OBSERVABILITY_API_URL
```

```
apiKey: SUSE_OBSERVABILITY_API_KEY
tokenType: SUSE_OBSERVABILITY_API_TOKEN_TYPE
apiToken: SUSE_OBSERVABILITY_TOKEN
clusterName: OBSERVED_SERVER_NAME
```

- 1 Instructs Helm to use credentials from the SUSE Application Collection. For instructions on how to configure the image pull secrets for the SUSE Application Collection, refer to the [official documentation \(https://docs.apps.rancher.io/get-started/authentication/\)](https://docs.apps.rancher.io/get-started/authentication/).

Run the install command.

```
> helm upgrade --install ai-obs \
  charts/suse-ai-observability-extension-X.Y.Z.tgz \
  -f genai_values.yaml --namespace so-extensions --create-namespace
```



### Note: Self-signed certificates not supported

Self-signed certificates are not supported. Consider running the extension in the same cluster as SUSE Observability and then use the internal Kubernetes address.

After the installation is complete, a new menu called *GenAI* is added to the Web interface and also a Kubernetes cron job is created that synchronizes the topology view with the components found in the SUSE AI cluster.

3. **Verify SUSE Observability extension.** After the installation, you can verify that a new lateral menu appears:

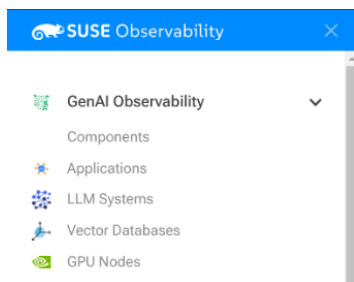


FIGURE 19: NEW GENAI OBSERVABILITY MENU ITEM

## 4.2.5 Setting up the SUSE AI cluster

Follow the instructions for your deployment scenario.

### Single-cluster deployment

In this setup, the SUSE AI components are installed in the same cluster as SUSE Observability and can communicate using internal service DNS.

### Multi-cluster deployment

In this setup, the SUSE AI cluster is separate. Communication relies on externally exposed endpoints of the SUSE Observability cluster.

The difference between deployment scenarios affects the **OTEL Collector exporter configuration** and the **SUSE Observability Agent URL** as described in the following list.

#### SUSE\_OBSERVABILITY\_API\_URL

The URL of the SUSE Observability API.

**Single-cluster example:** `http://suse-observability-otel-collector.suse-observability.svc.cluster.local:4317`

**Multi-cluster example:** `https://suse-observability-api.your-domain.com`

#### SUSE\_OBSERVABILITY\_COLLECTOR\_ENDPOINT

The endpoint of the SUSE Observability Collector.

**Single-cluster example:** `http://suse-observability-router.suse-observability.svc.cluster.local:8080/receiver/stsAgent`

**Multi-cluster example:** `https://suse-observability-router.your-domain.com/receiver/stsAgent`

1. **Install NVIDIA GPU Operator.** Follow the instructions in [https://documentation.suse.com/cloudnative/rke2/latest/en/advanced.html#\\_deploy\\_nvidia\\_operator](https://documentation.suse.com/cloudnative/rke2/latest/en/advanced.html#_deploy_nvidia_operator).
2. **Install OpenTelemetry collector.** Create a secret with your SUSE Observability API key in the namespace where you want to install the collector. Retrieve the API key using the Web UI or from the `baseConfig_values.yaml` file that you used during the SUSE Observability installation. If the namespace does not exist yet, create it.

```
kubectl create namespace observability
kubectl create secret generic open-telemetry-collector \
  --namespace observability \
  --from-literal=API_KEY='SUSE_OBSERVABILITY_API_KEY'
```

Create a new file named `otel-values.yaml` with the following content.

```
global:
  imagePullSecrets:
    - application-collection
```

```

repository: LOCAL_DOCKER_REGISTRY_URL:5043/opentelemetry-collector-k8s
extraEnvsFrom:
  - secretRef:
      name: open-telemetry-collector
mode: deployment
ports:
  metrics:
    enabled: true
presets:
  kubernetesAttributes:
    enabled: true
    extractAllPodLabels: true
config:
  receivers:
    prometheus:
      config:
        scrape_configs:
          - job_name: 'gpu-metrics'
            scrape_interval: 10s
            scheme: http
            kubernetes_sd_configs:
              - role: endpoints
                namespaces:
                  names:
                    - gpu-operator
          - job_name: 'milvus'
            scrape_interval: 15s
            metrics_path: '/metrics'

            static_configs:
              - targets:
                ['MILVUS_SERVICE_NAME.SUSE_AI_NAMESPACE.svc.cluster.local:9091'] ❶
        exporters:
          otlp:
            endpoint: https://OPEN_TELEMETRY_COLLECTOR_NAME.suse-observability.svc.cluster.local:4317 ❷
            headers:
              Authorization: "SUSEobservability ${env:API_KEY}"
            tls:
              insecure: true
        processors:
          tail_sampling:
            decision_wait: 10s
            policies:
              - name: rate-limited-composite
                type: composite
                composite:

```

```

    max_total_spans_per_second: 500
    policy_order: [errors, slow-traces, rest]
    composite_sub_policy:
      - name: errors
        type: status_code
        status_code:
          status_codes: [ ERROR ]
      - name: slow-traces
        type: latency
        latency:
          threshold_ms: 1000
      - name: rest
        type: always_sample
    rate_allocation:
      - policy: errors
        percent: 33
      - policy: slow-traces
        percent: 33
      - policy: rest
        percent: 34
  resource:
    attributes:
      - key: k8s.cluster.name
        action: upsert
        value: CLUSTER_NAME ③
      - key: service.instance.id
        from_attribute: k8s.pod.uid
        action: insert
  filter/dropMissingK8sAttributes:
    error_mode: ignore
  traces:
    span:
      - resource.attributes["k8s.node.name"] == nil
      - resource.attributes["k8s.pod.uid"] == nil
      - resource.attributes["k8s.namespace.name"] == nil
      - resource.attributes["k8s.pod.name"] == nil
  connectors:
    spanmetrics:
      metrics_expiration: 5m
      namespace: otel_span
    routing/traces:
      error_mode: ignore
      table:
        - statement: route()
          pipelines: [traces/sampling, traces/spanmetrics]
  service:
    extensions:

```



```

- health_check
pipelines:
  traces:
    receivers: [otlp, jaeger]
    processors: [filter/dropMissingK8sAttributes, memory_limiter, resource]
    exporters: [routing/traces]
  traces/spanmetrics:
    receivers: [routing/traces]
    processors: []
    exporters: [spanmetrics]
  traces/sampling:
    receivers: [routing/traces]
    processors: [tail_sampling, batch]
    exporters: [debug, otlp]
  metrics:
    receivers: [otlp, spanmetrics, prometheus]
    processors: [memory_limiter, resource, batch]
    exporters: [debug, otlp]

```

- ❶ Configure the Milvus service and namespace for the Prometheus scraper. Because Milvus will be installed in subsequent steps, you can return to this step and edit the endpoint if necessary.
- ❷ Set the exporter to your exposed SUSE Observability collector. Remember that the value can be distinct, depending on the deployment pattern. For production usage, we recommend using TLS communication.
- ❸ Replace `CLUSTER_NAME` with the cluster's name.

Finally, run the installation command.

```

> helm upgrade --install opentelemetry-collector \
  oci://dp.apps.rancher.io/charts/opentelemetry-collector \
  -f otel-values.yaml --namespace observability

```

Verify the installation by checking the existence of a new deployment and service in the observability namespace.

3. The GPU metrics scraper that we configure in the OTEL Collector requires custom RBAC rules. Create a file named `otel-rbac.yaml` with the following content:

```

apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: suse-observability-otel-scraper
rules:

```

```

- apiGroups:
  - ""
  resources:
    - services
    - endpoints
  verbs:
    - list
    - watch

---
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: suse-observability-otel-scraper
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: Role
  name: suse-observability-otel-scraper
subjects:
  - kind: ServiceAccount
    name: opentelemetry-collector
    namespace: observability

```

Then apply the configuration by running the following command.

```
> kubectl apply -n gpu-operator -f otel-rbac.yaml
```

#### 4. Install the SUSE Observability Agent.

```


> helm upgrade --install \
  --namespace suse-observability --create-namespace \
  --set-string 'stackstate.apiKey'='YOUR_API_KEY' ❶ \
  --set-string 'stackstate.cluster.name'='CLUSTER_NAME' ❷ \
  --set-string 'stackstate.url'='http://suse-observability-router.suse-
observability.svc.cluster.local:8080/receiver/stsAgent' ❸ \
  --set 'nodeAgent.skipKubeletTLSVerify'=true suse-observability-agent \
  suse-observability/suse-observability-agent

```

- ❶ Retrieve the API key using the Web UI or from the `baseConfig_values.yaml` file that you used during the SUSE Observability installation.
- ❷ Replace `CLUSTER_NAME` with the cluster's name.
- ❸ Replace with your SUSE Observability server URL.

#### 5. Install SUSE AI. Refer to [Section 5, “Installing applications from AI Library”](#) for the complete procedure.

## 4.2.6 Instrument applications

Instrumentation is the act of configuring your applications for telemetry data acquisition. Our stack employs OpenTelemetry standards as a vendor-neutral and open base for our telemetry. For a comprehensive guide on how to set up your instrumentation, please refer to [Monitoring SUSE AI with OpenTelemetry and SUSE Observability \(https://documentation.suse.com/suse-ai/1.0/html/AI-monitoring/index.html\)](https://documentation.suse.com/suse-ai/1.0/html/AI-monitoring/index.html) .

By following the instructions in the document referenced above, you will be able to retrieve all relevant telemetry data from Open WebUI, Ollama and Milvus by simply applying specific configuration to their Helm chart values. You can find links for advanced use cases (auto-instrumentation with OTEL Operator) at the end of the document.

# 5 Installing applications from AI Library

SUSE AI is delivered as a set of components that you can combine to meet specific use cases. To enable the full integrated stack, you need to deploy multiple applications in sequence. Applications with the fewest dependencies must be installed first, followed by dependent applications once their required dependencies are in place within the cluster.


## 5.1 Installation procedure

This procedure includes steps to install AI Library applications in an air-gapped environments.



### Note

If the following steps do not specify on which part of the air-gapped architecture—*local* or *remote*—the task should be performed, assume *remote*. The isolated *local* part is always be specified.

1. Purchase the SUSE AI entitlement. It is a separate entitlement from SUSE Rancher Prime.
2. Access SUSE AI via the SUSE Application Collection at <https://apps.rancher.io/>  to perform the check for the SUSE AI entitlement.

3. If the entitlement check is successful, you are given access to the SUSE AI-related Helm charts and container images, and can deploy directly from the SUSE Application Collection.
4. Visit the SUSE Application Collection, sign in and get the user access token as described in <https://docs.apps.rancher.io/get-started/authentication/>.
5. On the *local* cluster, create a Kubernetes namespace if it does not already exist. The steps in this procedure assume that all containers are deployed into the same namespace referred to as `SUSE_AI_NAMESPACE`. Replace its name to match your preferences.

```
> kubectl create namespace SUSE_AI_NAMESPACE
```

6. Create the SUSE Application Collection secret.

```
> kubectl create secret docker-registry application-collection \
  --docker-server=dp.apps.rancher.io \
  --docker-username=APPCO_USERNAME \
  --docker-password=APPCO_USER_TOKEN \
  -n SUSE_AI_NAMESPACE
```

7. Log in to the Helm registry.

```
> helm registry login dp.apps.rancher.io/charts \
  -u APPCO_USERNAME \
  -p APPCO_USER_TOKEN
```

8. On the *remote* host, download the `SUSE-AI-get-images.sh` script from the [Section 2.1, "Air-gapped stack"](#) and run it.

```
> ./SUSE-AI-get-images.sh
```

This script creates a subdirectory with all necessary Helm charts plus `suse-ai-containers.tgz` and `suse-ai-containers.txt` files.

9. Create a Docker registry on one of the *local* hosts so that the *local* Kubernetes cluster can access it.
10. Securely transfer the created subdirectory with Helm charts plus `suse-ai-containers.tgz` and `suse-ai-containers.txt` files from the *remote* host to a *local* host and load all container images to the *local* Docker registry. Set `DST_REGISTRY_USERNAME` and `DST_REGISTRY_PASSWORD` environment variables if they are required to access the registry.

```
> ./SUSE-AI-load-images.sh \
```

```
-d LOCAL_DOCKER_REGISTRY_URL \
-i charts/suse-ai-containers.txt \
-f charts/suse-ai-containers.tgz
```

11. Install cert-manager as described in [Section 5.2, “Installing cert-manager”](#).
12. Install AI Library components.
  - a. Install Milvus as described in [Section 5.3, “Installing Milvus”](#).
  - b. (Optional) Install Ollama as described in [Section 5.4, “Installing Ollama”](#).
  - c. Install Open WebUI as described in [Section 5.5, “Installing Open WebUI”](#).

## 5.2 Installing cert-manager

cert-manager is an extensible X.509 certificate controller for Kubernetes workloads. It supports certificates from popular public issuers as well as private issuers. cert-manager ensures that the certificates are valid and up-to-date, and attempts to renew certificates at a configured time before expiry.

In previous releases, cert-manager was automatically installed together with Open WebUI. Currently, cert-manager is no longer part of the Open WebUI Helm chart and you need to install it separately.

### 5.2.1 Details about the cert-manager application

Before deploying cert-manager, it is important to know more about the supported configurations and documentation. The following command provides the corresponding details:

```
helm show values oci://dp.apps.rancher.io/charts/cert-manager
```

Alternatively, you can also refer to the cert-manager Helm chart page on the SUSE Application Collection site at <https://apps.rancher.io/applications/cert-manager> [↗](#). It contains available versions and the link to pull the cert-manager container image.

## 5.2.2 cert-manager installation procedure



### Tip

Before the installation, you need to get user access to the SUSE Application Collection, create a Kubernetes namespace, and log in to the Helm registry as described in [Section 5.1, “Installation procedure”](#).

- Install the cert-manager chart.

```
> helm upgrade
--install cert-manager charts/cert-manager-X.Y.Z.tgz \
  -n CERT_MANAGER_NAMESPACE \
  --set crds.enabled=true \
  --set 'global.imagePullSecrets[0].name'=application-collection \
  --set 'global.imageRegistry'=LOCAL_DOCKER_REGISTRY_URL:5043
```

## 5.2.3 Uninstalling cert-manager

To uninstall cert-manager, run the following command:

```
> helm uninstall cert-manager -n CERT_MANAGER_NAMESPACE
```

## 5.3 Installing Milvus

Milvus is a scalable, high-performance vector database designed for AI applications. It enables efficient organization and searching of massive unstructured datasets, including text, images and multi-modal content. This procedure walks you through the installation of Milvus and its dependencies.


### 5.3.1 Details about the Milvus application

Before deploying Milvus, it is important to know more about the supported configurations and documentation. The following command provides the corresponding details:

```
helm show values oci://dp.apps.rancher.io/charts/milvus
```




Alternatively, you can also refer to the Milvus Helm chart page on the SUSE Application Collection site at <https://apps.rancher.io/applications/milvus>. It contains Milvus dependencies, available versions and the link to pull the Milvus container image.

applications / milvus

 **Milvus** SUSE AI  
Last updated 17 minutes ago

Milvus is a high-performance vector database built for scale. It is used by AI applications to organize and search through large amount of unstructured data, such as text, images, and multi-modal information.

This application depends on:

```
helm pull oci://dp.apps.rancher.io/charts/milvus --version 4.2.2
```

Branch 2.4  
1 Version

Version	Latest Revision	Latest Tag	App Version	Registered	Digest	Size
4.2.2	11	4.2.2-11	2.4.6	4 days ago	75b3655	104.2 kB

**Components**  
List of components this application depends on

**Milvus** Main  
Last updated 4 days ago

Branch 2.4  
1 Version

[VIEW ALL REVISIONS](#)

FIGURE 20: MILVUS PAGE IN THE SUSE APPLICATION COLLECTION

### 5.3.2 Milvus installation procedure



#### Tip

Before the installation, you need to get user access to the SUSE Application Collection, create a Kubernetes namespace, and log in to the Helm registry as described in [Section 5.1, “Installation procedure”](#).

1. When installed as part of SUSE AI, Milvus depends on etcd, MinIO and Apache Kafka. Because the Milvus chart uses a non-default configuration, create an override file `milvus_custom_overrides.yaml` with the following content.



## Tip

As a template, you can use the `values.yaml` file that is included in the `charts/milvus-X.Y.Z.tgz` TAR archive.

```
global:
  imagePullSecrets:
    - application-collection
  imageRegistry: LOCAL_DOCKER_REGISTRY_URL:5043
cluster:
  enabled: True
standalone:
  persistence:
    persistentVolumeClaim:
      storageClassName: "local-path"
etcd:
  replicaCount: 1
  persistence:
    storageClassName: "local-path"
minio:
  mode: distributed
  replicas: 4
  rootUser: "admin"
  rootPassword: "adminminio"
  persistence:
    storageClass: "local-path"
  resources:
    requests:
      memory: 1024Mi
kafka:
  enabled: true
  name: kafka
  replicaCount: 3
  broker:
    enabled: true
  cluster:
    listeners:
      client:
        protocol: 'PLAINTEXT'
      controller:
        protocol: 'PLAINTEXT'
  persistence:
    enabled: true
  annotations: {}
```



```

labels: {}
existingClaim: ""
accessModes:
  - ReadWriteOnce
resources:
  requests:
    storage: 8Gi
  storageClassName: "local-path"
extraConfigFiles: ❶
user.yaml: |+
  trace:
    exporter: jaeger
    sampleFraction: 1
    jaeger:
      url: "http://opentelemetry-collector.observability.svc.cluster.local:14268/
      api/traces" ❷

```

- ❶ The `extraConfigFiles` section is optional, required only to receive telemetry data from Open WebUI.
- ❷ The URL of the OpenTelemetry Collector installed by the user.



### Tip

The above example uses local storage. For production environments, we recommend using an enterprise class storage solution such as SUSE Storage in which case the `storageClassName` option must be set to `longhorn`.

2. Install the Milvus Helm chart using the `milvus_custom_overrides.yaml` override file.

```

> helm upgrade --install \
  milvus charts/milvus-X.Y.Z.tgz \
  -n SUSE_AI_NAMESPACE \
  --version X.Y.Z -f milvus_custom_overrides.yaml

```

#### 5.3.2.1 Using Apache Kafka with SUSE Storage

When Milvus is deployed in cluster mode, it uses Apache Kafka as a message queue. If Apache Kafka uses SUSE Storage as a storage back-end, you need to create an XFS storage class and make it available for the Apache Kafka deployment. Otherwise deploying Apache Kafka with a storage class of an Ext4 file system fails with the following error:

```
"Found directory /mnt/kafka/logs/lost+found, 'lost+found' is not
```

```
in the form of topic-partition or topic-partition.uniqueId-delete
(if marked for deletion)"
```

To introduce the XFS storage class, follow these steps:

1. Create a file named `longhorn-xfs.yaml` with the following content:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: longhorn-xfs
provisioner: driver.longhorn.io
allowVolumeExpansion: true
reclaimPolicy: Delete
volumeBindingMode: Immediate
parameters:
  numberOfReplicas: "3"
  staleReplicaTimeout: "30"
  fromBackup: ""
  fsType: "xfs"
  dataLocality: "disabled"
  unmapMarkSnapChainRemoved: "ignored"
```

2. Create the new storage class using the `kubectl` command.

```
> kubectl apply -f longhorn-xfs.yaml
```

3. Update the Milvus overrides YAML file to reference the Apache Kafka storage class, as in the following example:

```
[...]
  kafka:
    enabled: true
    persistence:
      storageClassName: longhorn-xfs
```

### 5.3.3 Uninstalling Milvus

To uninstall Milvus, run the following command:

```
> helm uninstall milvus -n SUSE_AI_NAMESPACE
```

## 5.4 Installing Ollama

Ollama is a tool for running and managing language models locally on your computer. It offers a simple interface to download, run and interact with models without relying on cloud resources.



### Tip

When installing SUSE AI, Ollama is installed by the Open WebUI installation by default. If you decide to install Ollama separately, disable its installation during the installation of Open WebUI as outlined in *Example 4, “Open WebUI override file with Ollama installed separately”*.

### 5.4.1 Details about the Ollama application

Before deploying Ollama, it is important to know more about the supported configurations and documentation. The following command provides the corresponding details:

```
helm show values oci://dp.apps.rancher.io/charts/ollama
```

Alternatively, you can also refer to the Ollama Helm chart page on the SUSE Application Collection site at <https://apps.rancher.io/applications/ollama>. It contains the available versions and a link to pull the Ollama container image.

### 5.4.2 Ollama installation procedure



### Tip

Before the installation, you need to get user access to the SUSE Application Collection, create a Kubernetes namespace, and log in to the Helm registry as described in *Section 5.1, “Installation procedure”*.

1. Create the `ollama_custom_overrides.yaml` file to override the values of the parent Helm chart. Refer to *Section 5.4.4, “Values for the Ollama Helm chart”* for more details.
2. Install the Ollama Helm chart using the `ollama-custom-overrides.yaml` override file.

```
> helm upgrade \
  --install ollama charts/ollama-X.Y.Z.tgz \
  -n SUSE_AI_NAMESPACE \
  -f ollama_custom_overrides.yaml
```



## Important: Downloading AI models

Ollama normally needs to have an active Internet connection to download AI models. In an air-gapped environment, you must download the models manually and copy them to your local Ollama instance, for example:

```
kubectl cp
  PATH_TO_LOCALLY_DOWNLOADED_MODELS/blobs/* \
  OLLAMA_POD_NAME:~/.ollama/models/blobs/
```



## Tip: Hugging Face models

Models downloaded from Hugging Face need to be converted before they can be used by Ollama. Refer to <https://github.com/ollama/ollama/blob/main/docs/import.md> for more details.

### 5.4.3 Uninstalling Ollama

To uninstall Ollama, run the following command:

```
> helm uninstall ollama -n SUSE_AI_NAMESPACE
```

### 5.4.4 Values for the Ollama Helm chart

To override the default values during the Helm chart installation or update, you can create an override YAML file with custom values. Then, apply these values by specifying the path to the override file with the `-f` option of the `helm` command.



## Important: GPU section

Ollama can run optimized for NVIDIA GPUs if the following conditions are fulfilled:

- The NVIDIA driver and NVIDIA GPU Operator are installed as described in [Installing NVIDIA GPU Drivers on SLES \(https://documentation.suse.com/suse-ai/1.0/html/NVIDIA-GPU-driver-on-SLES/index.html\)](https://documentation.suse.com/suse-ai/1.0/html/NVIDIA-GPU-driver-on-SLES/index.html) or [Installing NVIDIA GPU Drivers on SUSE Linux Micro \(https://documentation.suse.com/suse-ai/1.0/html/NVIDIA-GPU-driver-on-SL-Micro/index.html\)](https://documentation.suse.com/suse-ai/1.0/html/NVIDIA-GPU-driver-on-SL-Micro/index.html).
- The workloads are set to run on NVIDIA-enabled nodes as described in <https://documentation.suse.com/suse-ai/1.0/html/AI-deployment-intro/index.html#ai-gpu-nodes-assigning>.

If you do not want to use the NVIDIA GPU, remove the `gpu` section from `ollama_custom_overrides.yaml` or disable it.

```
ollama:
  [...]
  gpu:
    enabled: false
    type: 'nvidia'
    number: 1
```

### EXAMPLE 1: BASIC OVERRIDE FILE WITH GPU AND TWO MODELS PULLED AT STARTUP

```
global:
  imagePullSecrets:
    - application-collection
ingress:
  enabled: false
defaultModel: "gemma:2b"
runtimeClassName: nvidia
ollama:
  models:
    pull:
      - "gemma:2b"
      - "llama3.1"
    run:
      - "gemma:2b"
      - "llama3.1"
  gpu:
    enabled: true
```

```

    type: 'nvidia'
    number: 1
    nvidiaResource: "nvidia.com/gpu"
persistentVolume: ❶
    enabled: true
    storageClass: local-path ❷

```

- ❶ Without the `persistentVolume` option enabled, changes made to Ollama—such as downloading other LLM—are lost when the container is restarted.
- ❷ Use `local-path` storage only for testing purposes. For production use, we recommend using a storage solution suitable for persistent storage, such as SUSE Storage.

#### EXAMPLE 2: BASIC OVERRIDE FILE WITH INGRESS AND NO GPU

```

ollama:
  models:
    pull:
      - llama2
    run:
      - llama2
  persistentVolume:
    enabled: true
    storageClass: local-path ❶
  ingress:
    enabled: true
    hosts:
      - host: OLLAMA_API_URL
    paths:
      - path: /
        pathType: Prefix

```

- ❶ Use `local-path` storage (requires installing the corresponding provisioner) only for testing purposes. For production use, we recommend using a storage solution suitable for persistent storage, such as SUSE Storage.

TABLE 1: OVERRIDE FILE OPTIONS FOR THE OLLAMA HELM CHART

Key	Type	Default	Description
affinity	object	{}	Affinity for pod assignment
autoscaling.enabled	bool	false	Enable autoscaling

Key	Type	Default	Description
autoscaling.maxReplicas	int	100	Number of maximum replicas
autoscaling.minReplicas	int	1	Number of minimum replicas
autoscaling.targetCPUUtilizationPercentage	int	80	CPU usage to target replica
extraArgs	list	[]	Additional arguments on the output Deployment definition.
extraEnv	list	[]	Additional environment variables on the output Deployment definition.
fullnameOverride	string	""	String to fully override template
global.imagePullSecrets	list	[]	Global override for container image registry pull secrets
global.imageRegistry	string	""	Global override for container image registry
hostIPC	bool	false	Use the host's IPC namespace
hostNetwork	bool	false	Use the host's network namespace


Key	Type	Default	Description
hostPID	bool	false	Use the host's PID namespace.
image.pullPolicy	string	"IfNotPresent"	Image pull policy to use for the Ollama container
image.registry	string	"dp.apps.rancher.io"	Image registry to use for the Ollama container
image.repository	string	"containers/ollama"	Image repository to use for the Ollama container
image.tag	string	"0.3.6"	Image tag to use for the Ollama container
imagePullSecrets	list	[]	Docker registry secret names as an array
ingress.annotations	object	{}	Additional annotations for the Ingress resource
ingress.className	string	""	IngressClass that is used to implement the Ingress (Kubernetes 1.18+)
ingress.enabled	bool	false	Enable Ingress controller resource
ingress.hosts[0].host	string	"ollama.local"	
ingress.hosts[0].paths[0].path	string	"/"	




Key	Type	Default	Description
ingress.hosts[0].paths[0].pathType	string	"Prefix"	
ingress.tls	list	[]	The TLS configuration for host names to be covered with this Ingress record
initContainers	list	[]	Init containers to add to the pod
knative.container-Concurrency	int	0	Knative service container concurrency
knative.enabled	bool	false	Enable Knative integration
knative.idleTimeoutSeconds	int	300	Knative service idle timeout seconds
knative.responseStartTimeTimeoutSeconds	int	300	Knative service response start timeout seconds
knative.timeoutSeconds	int	300	Knative service timeout seconds
livenessProbe.enabled	bool	true	Enable livenessProbe
livenessProbe.failureThreshold	int	6	Failure threshold for livenessProbe
livenessProbe.initialDelaySeconds	int	60	Initial delay seconds for livenessProbe

Key	Type	Default	Description
livenessProbe.path	string	"/"	Request path for livenessProbe
livenessProbe.periodSeconds	int	10	Period seconds for livenessProbe
livenessProbe.successThreshold	int	1	Success threshold for livenessProbe
livenessProbe.timeoutSeconds	int	5	Timeout seconds for livenessProbe
nameOverride	string	""	String to partially override template (maintains the release name)
nodeSelector	object	{}	Node labels for pod assignment
ollama.gpu.enabled	bool	false	Enable GPU integration
ollama.gpu.number	int	1	Specify the number of GPUs
ollama.gpu.nvidiaResource	string	"nvidia.com/gpu"	Only for NVIDIA cards; change to <u>nvidia.com/mig-1g.10gb</u> to use MIG slice
ollama.gpu.type	string	"nvidia"	GPU type: "nvidia" or "amd." If "ollama.gpu.enabled" is enabled, the default value is "nvidia." If

Key	Type	Default	Description
			set to “amd,” this adds the “rocm” suffix to the image tag if “image.tag” is not override. This is because AMD and CPU/CUDA are different images.
ollama.insecure	bool	false	Add insecure flag for pulling at container startup
ollama.models	list	[]	List of models to pull at container startup. The more you add, the longer the container takes to start if models are not present models: - llama2 - mistral
ollama.mountPath	string	""	Override ollama-data volume mount path, default: "/root/.ollama"
persistentVolume.accessModes	list	["ReadWriteOnce"]	Ollama server data Persistent Volume access modes. Must match those of existing PV or dynamic provisioner, see <a href="https://kuber-">https://kuber-</a>

Key	Type	Default	Description
			<a href="https://netes.io/docs/concepts/storage/persistent-volumes/">netes.io/docs/concepts/storage/persistent-volumes/</a>  .
<code>persistentVolume.annotations</code>	object	<code>{}</code>	Ollama server data Persistent Volume annotations
<code>persistentVolume.enabled</code>	bool	false	Enable persistence using PVC
<code>persistentVolume.existingClaim</code>	string	""	If you want to bring your own PVC for persisting Ollama state, pass the name of the created + ready PVC here. If set, this Chart does not create the default PVC. Requires <code>server.persistentVolume.enabled: true</code>
<code>persistentVolume.size</code>	string	"30Gi"	Ollama server data Persistent Volume size
<code>persistentVolume.storageClass</code>	string	""	If <code>persistentVolume.storageClass</code> is present, and is set to either a dash ("-") or empty string (" /"), dynamic provisioning is disabled. Otherwise, the storageClassName for per-

Key	Type	Default	Description
			<p>sistent volume claim is set to the given value specified by <code>persistentVolume.storageClass</code>. If <code>persistentVolume.storageClass</code> is absent, the default storage class is used for dynamic provisioning whenever possible. See <a href="https://kubernetes.io/docs/concepts/storage/storage-classes/">https://kubernetes.io/docs/concepts/storage/storage-classes/</a>  for more details.</p>
<code>persistentVolume.subPath</code>	string	""	<p>Subdirectory of Ollama server data Persistent Volume to mount. Useful if the volume's root directory is not empty.</p>
<code>persistentVolume.volumeMode</code>	string	""	<p>Ollama server data Persistent Volume Binding Mode. If empty (the default) or set to null, no <code>volumeBindingMode</code> specification is set, choosing the default mode.</p>

Key	Type	Default	Description
persistentVolume.volumeName	string	""	Ollama server Persistent Volume name. It can be used to force-attach the created PVC to a specific PV.
podAnnotations	object	{}	Map of annotations to add to the pods
podLabels	object	{}	Map of labels to add to the pods
podSecurityContext	object	{}	Pod Security Context
readinessProbe.enabled	bool	true	Enable readinessProbe
readinessProbe.failureThreshold	int	6	Failure threshold for readinessProbe
readinessProbe.initialDelaySeconds	int	30	Initial delay seconds for readinessProbe
readinessProbe.path	string	"/"	Request path for readinessProbe
readinessProbe.periodSeconds	int	5	Period seconds for readinessProbe
readinessProbe.successThreshold	int	1	Success threshold for readinessProbe
readinessProbe.timeoutSeconds	int	3	Timeout seconds for readinessProbe
replicaCount	int	1	Number of replicas
resources.limits	object	{}	Pod limit

Key	Type	Default	Description
resources.requests	object	{}	Pod requests
runtimeClassName	string	""	Specify runtime class
securityContext	object	{}	Container Security Context
service.annotations	object	{}	Annotations to add to the service
service.nodePort	int	31434	Service node port when service type is "NodePort"
service.port	int	11434	Service port
service.type	string	"ClusterIP"	Service type
serviceAccount.annotations	object	{}	Annotations to add to the service account
serviceAccount.auto-mount	bool	true	Whether to automatically mount a ServiceAccount's API credentials
serviceAccount.create	bool	true	Whether a service account should be created
serviceAccount.name	string	""	The name of the service account to use. If not set and "create" is "true", a name is generated using the full name template.

Key	Type	Default	Description
tolerations	list	[]	Tolerations for pod assignment
topologySpreadConstraints	object	{}	Topology Spread Constraints for pod assignment
updateStrategy	object	{"type":""}	How to replace existing pods.
updateStrategy.type	string	""	Can be "Recreate" or "RollingUpdate"; default is "RollingUpdate"
volumeMounts	list	[]	Additional volumeMounts on the output Deployment definition
volumes	list	[]	Additional volumes on the output Deployment definition

## 5.5 Installing Open WebUI

Open WebUI is a Web-based user interface designed for interacting with AI models.

### 5.5.1 Details about the Open WebUI application

Before deploying Open WebUI, it is important to know more about the supported configurations and documentation. The following command provides the corresponding details:

```
helm show values oci://dp.apps.rancher.io/charts/open-webui
```



Alternatively, you can also refer to the Open WebUI Helm chart page on the SUSE Application Collection site at <https://apps.rancher.io/applications/open-webui><sup>7</sup>. It contains available versions and the link to pull the Open WebUI container image.

### 5.5.2 Open WebUI installation procedure



#### Tip

Before the installation, you need to get user access to the SUSE Application Collection, create a Kubernetes namespace, and log in to the Helm registry as described in [Section 5.1, “Installation procedure”](#).

#### REQUIREMENTS

To install Open WebUI, you need to have the following:

- An installed cert-manager. If cert-manager is not installed from previous Open WebUI releases, install it by following the steps in [Section 5.2, “Installing cert-manager”](#).
1. Create the `owui_custom_overrides.yaml` file to override the values of the parent Helm chart. The file contains URLs for Milvus and Ollama and specifies whether a stand-alone Ollama deployment is used or whether Ollama is installed as part of the Open WebUI installation. Find more details in [Section 5.5.4, “Examples of Open WebUI Helm chart override files”](#). For a list of all installation options with examples, refer to [Section 5.5.5, “Values for the Open WebUI Helm chart”](#).
  2. Install the Open WebUI Helm chart using the `owui_custom_overrides.yaml` override file.

```
> helm upgrade --install \
  open-webui charts/open-webui-X.Y.Z.tgz \
  -n SUSE_AI_NAMESPACE \
  --version X.Y.Z -f owui_custom_overrides.yaml
```

### 5.5.3 Uninstalling Open WebUI

To uninstall Open WebUI, run the following command:

```
> helm uninstall open-webui -n SUSE_AI_NAMESPACE
```

## 5.5.4 Examples of Open WebUI Helm chart override files

To override the default values during the Helm chart installation or update, you can create an override YAML file with custom values. Then, apply these values by specifying the path to the override file with the `-f` option of the `helm` command.

### EXAMPLE 3: OPEN WEBUI OVERRIDE FILE WITH OLLAMA INCLUDED

The following override file installs Ollama during the Open WebUI installation. Replace `SUSE_AI_NAMESPACE` with your Kubernetes namespace.

```
global:
  imagePullSecrets:
    - application-collection
  imageRegistry: LOCAL_DOCKER_REGISTRY_URL:5043
ollamaUrls:
  - http://open-webui-ollama.SUSE_AI_NAMESPACE.svc.cluster.local:11434
persistence:
  enabled: true
  storageClass: local-path ❶
ollama:
  enabled: true
  ingress:
    enabled: false
  defaultModel: "gemma:2b"
  ollama:
    models: ❷
    - "gemma:2b"
    - "llama3.1"
  gpu: ❸
    enabled: true
    type: 'nvidia'
    number: 1
  persistentVolume: ❹
    enabled: true
    storageClass: local-path ❺
pipelines:
  enabled: False
  persistence:
    storageClass: local-path ❻
ingress:
  enabled: true
  class: ""
  annotations:
    nginx.ingress.kubernetes.io/ssl-redirect: "true"
```

```

host: suse-ollama-webui ⑦
tls: true
extraEnvVars:
- name: DEFAULT_MODELS ⑧
  value: "gemma:2b"
- name: DEFAULT_USER_ROLE
  value: "user"
- name: WEBUI_NAME
  value: "SUSE AI"
- name: GLOBAL_LOG_LEVEL
  value: INFO
- name: RAG_EMBEDDING_MODEL
  value: "sentence-transformers/all-MiniLM-L6-v2"
- name: VECTOR_DB
  value: "milvus"
- name: MILVUS_URI
  value: http://milvus.SUSE_AI_NAMESPACE.svc.cluster.local:19530
- name: INSTALL_NLTK_DATASETS ⑨
  value: "true"

```

- ② Specifies that two large language models (LLM) will be loaded in Ollama when the container starts.
- ③ Enables GPU support for Ollama. The type must be nvidia because NVIDIA GPUs are the only supported devices. number must be between 1 and the number of NVIDIA GPUs present on the system.
- ④ Without the persistentVolume option enabled, changes made to Ollama—such as downloading other LLM—are lost when the container is restarted.
- ① ⑤ ⑥ local-path storage only for testing purposes. For production use, we recommend using a storage solution suitable for persistent storage, such as SUSE Storage.
- ⑧ Specifies the default LLM for Ollama.
- ⑦ Specifies the host name for the Open WebUI Web UI.
- ⑨ Installs the *natural language toolkit* (NLTK) datasets for Ollama. Refer to <https://www.nltk.org/index.html> for licensing information.

#### EXAMPLE 4: OPEN WEBUI OVERRIDE FILE WITH OLLAMA INSTALLED SEPARATELY

The following override file installs Ollama separately from the Open WebUI installation. Replace SUSE\_AI\_NAMESPACE with your Kubernetes namespace.

```

global:
  imagePullSecrets:
  - application-collection

```

```

imageRegistry: LOCAL_DOCKER_REGISTRY_URL:5043
ollamaUrls:
- http://ollama.SUSE_AI_NAMESPACE.svc.cluster.local:11434
persistence:
  enabled: true
  storageClass: local-path ❶
ollama:
  enabled: false
pipelines:
  enabled: False
  persistence:
    storageClass: local-path ❷
ingress:
  enabled: true
  class: ""
  annotations:
    nginx.ingress.kubernetes.io/ssl-redirect: "true"
  host: suse-ollama-webui
  tls: true
extraEnvVars:
- name: DEFAULT_MODELS ❸
  value: "gemma:2b"
- name: DEFAULT_USER_ROLE
  value: "user"
- name: WEBUI_NAME
  value: "SUSE AI"
- name: GLOBAL_LOG_LEVEL
  value: INFO
- name: RAG_EMBEDDING_MODEL
  value: "sentence-transformers/all-MiniLM-L6-v2"
- name: VECTOR_DB
  value: "milvus"
- name: MILVUS_URI
  value: http://milvus.SUSE_AI_NAMESPACE.svc.cluster.local:19530
- name: ENABLE_OTEL ❹
  value: "true"
- name: OTEL_EXPORTER_OTLP_ENDPOINT ❺
  value: http://opentelemetry-collector.observability.svc.cluster.local:4317 ❻

```


- ❶ ❷ Use local-path storage only for testing purposes. For production use, we recommend using a storage solution suitable for persistent storage, such as SUSE Storage.
- ❸ Specifies the default LLM for Ollama.
- ❹ ❺ These values are optional, required only to receive telemetry data from Open WebUI.
- ❻ The URL of the OpenTelemetry Collector installed by the user.

### 5.5.5 Values for the Open WebUI Helm chart

To override the default values during the Helm chart installation or update, you can create an override YAML file with custom values. Then, apply these values by specifying the path to the override file with the `-f` option of the `helm` command.

TABLE 2: AVAILABLE OPTIONS FOR THE OPEN WEBUI HELM CHART

Key	Type	Default	Description
affinity	object	{}	Affinity for pod assignment
annotations	object	{}	
cert-manager.enabled	bool	true	
clusterDomain	string	"cluster.local"	Value of cluster domain
containerSecurity-Context	object	{}	Configure container security context, see <a href="https://kubernetes.io/docs/tasks/configure-pod-container/security-context/#set-the-security-context-for-a-container">https://kubernetes.io/docs/tasks/configure-pod-container/security-context/#set-the-security-context-for-a-container</a> .
extraEnvVars	list	[{"name":"OPENAI_API_KEY", "value":"Op3n-w3bu!"}]	Environment variables added to the Open WebUI deployment. Most up-to-date environment variables can be found in <a 102="" 127="" 913="" 929"="" data-label="Page-Footer" href="https://doc-&lt;/a&gt;&lt;/td&gt;&lt;/tr&gt;&lt;/table&gt;&lt;/div&gt;&lt;div data-bbox="><p>85</p></a>

Key	Type	Default	Description
			<a href="https://s.openwebui.com/getting-started/env-configuration/">s.openwebui.com/getting-started/env-configuration/</a>  .
extraEnvVars[0]	object	<code>{"name":"OPENAI_API_KEY","value":"0p3n-w3bu!"}</code>	Default API key value for Pipelines. It should be updated in a production deployment and changed to the required API key if not using Pipelines.
global.imagePullSecrets	list	<code>[]</code>	Global override for container image registry pull secrets
global.imageRegistry	string	<code>""</code>	Global override for container image registry
global.tls.additionalTrustedCAs	bool	<code>false</code>	
global.tls.issuerName	string	<code>"suse-private-ai"</code>	
global.tls.letsEncrypt.email	string	<code>"none@example.com"</code>	
global.tls.letsEncrypt.environment	string	<code>"staging"</code>	
global.tls.letsEncrypt.ingress.class	string	<code>""</code>	

Key	Type	Default	Description
global.tls.source	string	"suse-private-ai"	The source of Open WebUI TLS keys, see <a href="#">Section 5.5.5.1, "TLS sources"</a> .
image.pullPolicy	string	"IfNotPresent"	Image pull policy to use for the Open WebUI container
image.registry	string	"dp.apps.rancher.io"	Image registry to use for the Open WebUI container
image.repository	string	"containers/open-webui"	Image repository to use for the Open WebUI container
image.tag	string	"0.3.32"	Image tag to use for the Open WebUI container
imagePullSecrets	list	[]	Configure imagePullSecrets to use private registry, see <a href="https://kubernetes.io/docs/tasks/configure-pod-container/pull-image-private-registry">https://kubernetes.io/docs/tasks/configure-pod-container/pull-image-private-registry</a> .
ingress.annotations	object	{"nginx.ingress.kubernetes.io/ssl-redirect":"true"}	Use appropriate annotations for your Ingress controller, such as <code>nginx.ingress.kubernetes.io/ssl-redirect</code> .

Key	Type	Default	Description
			<a href="https://netes.io/rewrite-target/">netes.io/rewrite-target:</a> / for NGINX.
ingress.class	string	""	
ingress.enabled	bool	true	
ingress.existingSecret	string	""	
ingress.host	string	""	
ingress.tls	bool	true	
nameOverride	string	""	
nodeSelector	object	{}	Node labels for pod assignment
ollama.enabled	bool	true	Automatically install Ollama Helm chart from <a href="https://otwld.github.io/ollama-helm/">https://otwld.github.io/ollama-helm/</a>  . Configure the following Helm values ( <a href="https://github.com/otwld/ollama-helm/#helm-values">https://github.com/otwld/ollama-helm/#helm-values</a> )  .
ollama.fullnameOverride	string	"open-webui-ollama"	If enabling embedded Ollama, update fullnameOverride to your desired Ollama name value, or else it will use the



Key	Type	Default	Description
			default ollama.name value from the Ollama chart.
ollamaUrls	list	[]	A list of Ollama API endpoints. These can be added instead of automatically installing the Ollama Helm chart, or in addition to it.
openaiBaseApiUrl	string	""	OpenAI base API URL to use. Defaults to the Pipelines service endpoint when Pipelines are enabled, or to <a href="https://api.openai.com/v1">https://api.openai.com/v1</a> if Pipelines are not enabled and this value is blank.
persistence.accessModes	list	["ReadWriteOnce"]	If using multiple replicas, you must update accessModes to ReadWriteMany.
persistence.annotations	object	{}	
persistence.enabled	bool	true	

Key	Type	Default	Description
persistence.existing-Claim	string	""	Use existingClaim to reuse an existing Open WebUI PVC instead of creating a new one.
persistence.selector	object	{}	
persistence.size	string	"2Gi"	
persistence.storage-Class	string	""	
pipelines.enabled	bool	false	Automatically install Pipelines chart to extend Open WebUI functionality using Pipelines, see <a href="https://github.com/open-webui/pipelines">https://github.com/open-webui/pipelines</a> .
pipelines.extraEnvVars	list	[]	This section can be used to pass the required environment variables to your pipelines (such as the Langfuse host name).
podAnnotations	object	{}	
podSecurityContext	object	{}	Configure pod security context, see <a href="https://kubernetes.io/docs/tasks/configure-pod-container/security-con">https://kubernetes.io/docs/tasks/configure-pod-container/security-con</a>

Key	Type	Default	Description
			<a href="#">text/#set-the-security-context-for-a-container</a> .
replicaCount	int	1	
resources	object	{}	
service	object	{"annotations": {}, "containerPort": 8080, "labels": {}, "loadBalancerClass": "", "nodePort": "", "port": 80, "type": "ClusterIP"}	Service values to expose Open WebUI pods to cluster
tolerations	list	[]	Tolerations for pod assignment
topologySpreadConstraints	list	[]	Topology Spread Constraints for pod assignment

#### 5.5.5.1 TLS sources

There are three recommended options where Open WebUI can obtain TLS certificates for secure communication.

##### Self-Signed TLS certificate

This is the default method. You need to install `cert-manager` on the cluster to issue and maintain the certificates. This method generates a CA and signs the Open WebUI certificate using the CA. `cert-manager` then manages the signed certificate.

For this method, use the following Helm chart option:

```
global.tls.source=suse-private-ai
```

## Let's Encrypt

This method also uses `cert-manager`, but it is combined with a special issuer for Let's Encrypt that performs all actions—including request and validation—to get the Let's Encrypt certificate issued. This configuration uses HTTP validation (HTTP-01) and therefore the load balancer must have a public DNS record and be accessible from the Internet.

For this method, use the following Helm chart option:

```
global.tls.source=letsEncrypt
```

## Provide your own certificate

This method allows you to bring your own signed certificate to secure the HTTPS traffic. In this case, you must upload this certificate and associated key as PEM-encoded files named `tls.crt` and `tls.key`.

For this method, use the following Helm chart option:

```
global.tls.source=secret
```

# 6 Steps after the installation is complete

Once the SUSE AI installation is finished, follow these tasks to complete the initial setup and configuration.

1. Log in to SUSE AI Open WebUI using the default credentials.
2. After you have logged in, update the administrator password for SUSE AI.
3. From the available language models, configure the one you prefer. Optionally, install a custom language model. Refer to the section [Setting base AI models \(https://documentation.suse.com/suse-ai/1.0/html/openwebui-configuring/index.html#openwebui-setting-base-models\)](https://documentation.suse.com/suse-ai/1.0/html/openwebui-configuring/index.html#openwebui-setting-base-models) and [Setting the default AI model \(https://documentation.suse.com/suse-ai/1.0/html/openwebui-configuring/index.html#openwebui-setting-default-models\)](https://documentation.suse.com/suse-ai/1.0/html/openwebui-configuring/index.html#openwebui-setting-default-models) for more details
4. Configure user management with *role-base access control* (RBAC) as described in <https://documentation.suse.com/suse-ai/1.0/html/openwebui-configuring/index.html#openwebui-managing-user-roles>

5. Integrate *single sign-on* authentication manager—such as Okta—with Open WebUI as described in <https://documentation.suse.com/suse-ai/1.0/html/openwebui-configuring/index.html#openwebui-authentication-via-okta>.
6. Configure *retrieval-augmented generation* (RAG) to let the model process content relevant to the customer.

## 7 Legal Notice

Copyright© 2006–2025 SUSE LLC and contributors. All rights reserved.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or (at your option) version 1.3; with the Invariant Section being this copyright notice and license. A copy of the license version 1.2 is included in the section entitled “GNU Free Documentation License”.

For SUSE trademarks, see <https://www.suse.com/company/legal/>. All other third-party trademarks are the property of their respective owners. Trademark symbols (®, ™ etc.) denote trademarks of SUSE and its affiliates. Asterisks (\*) denote third-party trademarks.

All information found in this book has been compiled with utmost attention to detail. However, this does not guarantee complete accuracy. Neither SUSE LLC, its affiliates, the authors, nor the translators shall be held liable for possible errors or the consequences thereof.

## Glossary

### AI, artificial intelligence

Refers to the simulation of human intelligence in machines that are designed to learn and solve problems like humans. Enables computers to understand language, make decisions and improve from experience.

### Air gap

A security measure where a computer network is physically isolated from unsecured networks, including the public Internet.

**Batch size**

The number of samples processed simultaneously during model inference, affecting processing speed and resource utilization.

**BYOC, bring your own certificate**

A practice allowing users to provide their own SSL/TLS certificates for securing communications instead of using default or auto-generated ones.

**CA, certification authority**

An entity that issues digital certificates to verify the identity of certificate holders and ensure secure communications.

**Chain-of-thought (CoT) prompting**

A prompting technique that guides AI models to break down complex problems into step-by-step reasoning processes, improving response accuracy and transparency.

**Chat template**

A structured format for organizing conversations between users and AI models, defining how system prompts, user inputs, and AI responses are formatted and processed.

**Context window**

The maximum amount of text (tokens) that an AI model can process at once, including both the input prompt and generated response.

**CRD, custom resource definitions**

Extensions of the Kubernetes API that allow users to define custom resources and their controllers in a Kubernetes cluster.

**CUDA, Compute Unified Device Architecture**

NVIDIA's parallel computing platform and programming model used to accelerate AI workloads on GPU hardware.

**Data leakage**

The unintended exposure of sensitive information through AI model responses, potentially compromising data security and privacy.

**Embeddings**

Numerical representations of data (text, images, etc.) in a high-dimensional space that capture semantic relationships and enable AI models to process information effectively.

**Fine-tuning**

The process of further training a pre-trained AI model on specific data to adapt it for particular tasks or domains, improving its performance for targeted applications.

**GenAI, generative AI**

A type of artificial intelligence that can create new content such as text, images or music.

**GPU, graphics processing unit**

Specialized hardware designed for parallel processing. In AI applications, GPUs accelerate model training and inference tasks.

**Hallucination**

An AI behavior where the model generates false or unsupported information that appears plausible but has no basis in provided context or real facts.

**Helm**

A package manager for Kubernetes that helps install and manage applications. Helm uses charts to define, install and upgrade complex Kubernetes applications.

**Helm chart**

A package format for Kubernetes applications that contains all resource definitions needed to deploy and configure application workloads.

**IaC, infrastructure as code**

The practice of managing and provisioning infrastructure through machine-readable definition files rather than manual processes.

**Inference**

The process of using a trained AI model to make predictions or generate outputs based on new input data.

**Kubernetes pods**

The smallest deployable units in Kubernetes that can host one or more containers, sharing networking and storage resources.

**LLM, large language model**

An advanced AI model trained on amounts of text data to understand and generate human-like text. Can perform tasks like translation, summarization and answering questions.

**Model weights**

The learned parameters of an AI model that determine how it processes inputs and generates outputs. These weights are adjusted during training to optimize model performance.

**NLG, natural language generation**

A process of automatically generating human-like text from structured data or other forms of input. Designed to convert raw data into coherent and meaningful language easily understood by humans.

**NLU, natural language understanding**

A process AI uses to analyze and understand the meaning of the input query.

**NVIDIA GPU driver**

Software that enables communication between the operating system and NVIDIA graphics hardware, essential for GPU-accelerated AI workloads.

**NVIDIA GPU Operator**

A Kubernetes operator that automates the management of NVIDIA GPUs in container environments, handling driver deployment, runtime configuration, and monitoring.

**Ollama**

An open source framework for running and serving AI models locally. Ollama simplifies the process of downloading, running and managing large language models.

**OpenGL**

A cross-platform API for rendering 2D and 3D graphics, commonly used in visualization applications and GPU-accelerated computing.

**Prompt Engineering**

The practice of crafting effective input queries to AI models to obtain desired and accurate outputs. Good prompt engineering helps prevent hallucinations and improves response quality.

**Prompt injection**

A security vulnerability where malicious inputs attempt to override or bypass an AI model's system prompt or safety constraints.

**Quantization**

A technique to reduce AI model size and computational requirements by converting model parameters to lower precision formats while maintaining acceptable performance.



**RAG, retrieval-augmented generation**

A technique that enhances AI responses by retrieving relevant information from a knowledge base before generating answers, improving accuracy and reducing hallucinations.

**RBAC, role-based access control**

A security model that restricts system access based on roles assigned to users, managing permissions and authorization in Kubernetes clusters.

**Semantic search**

A search method using AI to understand the meaning and context of queries rather than just matching keywords, enabling more relevant results.

**System prompt**

Initial instructions given to an AI model that define its behavior, role and response parameters. System prompts help maintain consistent and appropriate AI responses.

**Temperature**

A parameter controlling the randomness in AI model outputs. Lower values produce more focused and deterministic responses, while higher values increase creativity and variability.

**Token**

The basic unit of text processing in AI models, representing parts of words, characters or symbols. Models process text by breaking it into tokens for analysis and generation.

**Top-K**

A parameter that limits token selection during text generation to the K most likely next tokens, helping control output quality and relevance.

**Top-P**

Also known as nucleus sampling, a parameter that selects from the smallest set of tokens whose cumulative probability exceeds P, providing dynamic control over text generation diversity.

**Vector database**

A specialized database designed to store and efficiently query high-dimensional vectors that represent data in AI applications, enabling similarity searches and semantic operations.

**Vector store**

A specialized storage system optimized for managing and querying vector embeddings, essential for semantic search and RAG implementations in AI applications.

# A GNU Free Documentation License

Copyright (C) 2000, 2001, 2002 Free Software Foundation, Inc. 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA. Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

## 0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or non-commercially. Secondly, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

## 1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that

overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition. The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

## 2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or non-commercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

## 3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

## 4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
- C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
- D. Preserve all the copyright notices of the Document.

- E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- H. Include an unaltered copy of this License.
- I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
- J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
- K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
- N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
- O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

## 5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

## 6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

## 7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

## 8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.



If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

## 9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

## 10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See <https://www.gnu.org/copyleft/>.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

## ADDENDUM: How to use this License for your documents

```
Copyright (c) YEAR YOUR NAME.  
Permission is granted to copy, distribute and/or modify this document  
under the terms of the GNU Free Documentation License, Version 1.2  
or any later version published by the Free Software Foundation;  
with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.  
A copy of the license is included in the section entitled "GNU  
Free Documentation License".
```

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the “with...Texts.” line with this:

```
with the Invariant Sections being LIST THEIR TITLES, with the
Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.