AMORPH.pro
SMARTUNIFIER

# SMARTUNIFIER Installation Guide

***Release 1.9.0***

**Amorph Systems GmbH**

**Jul 18, 2024**

# PLAN SMARTUNIFIER INSTALLATION

CHAPTER

ONE

# PLANNING THE INSTALLATION

In production it is typically recommended to host SMART**UNIFIER** Manager and the Communication Instances on separate servers.

In a test environment, the Manager and the Communication Instances can run on the same hardware.

The SMART**UNIFIER** Manager and the Communication Instances can be run on dedicated hardware as well as in a virtualized environment. CPU and memory requirements are estimated based on modern hardware (e.g. Intel Xeon Coffe Lake / Core i5-7xxx or AMD Epyc / Ryzen 5 3xxx or greater). In a virtualized environment, dedicated resources are highly recommended.

## Minimum System Requirements (Manager)

- **Computer and Processor**: 4 cores
- **Memory**: 1GB free memory
- **Storage (SSD)**: 10GB free space
- **Display PC (Engineering, Dashboard)**: 1920x1080 (Full HD)
- **Mobile Devices (Dashboard)**: Latest version of Apple iPadOS, Apple iOS, Android
- **Operating System**: Latest version of Windows, Windows Server, Linux, MacOS (Not recommended for production) - For an optimal user experience always use the latest version of the operating system
- **Browser**: Latest version of Chrome, Microsoft Edge or Firefox

## Minimum System Requirements (1 Communication Instance)

- **Computer and Processor**: 4 cores
- **Memory**: 256MB free memory
- **Storage (SSD)**: 5GB free space
- **Operating System**: Latest version of Windows, Windows Server or Linux

## Production Deployment Example

Multiple SMART**UNIFIER** Instances can be operated on one server. The number of Instances for each server depends on the overall scenario.

For a deplyoment of ca. 30-40 Communication Instances, the servers minimum requirements are:

- **Computer and Processor**: 8 cores
- **Memory**: 16GB free memory
- **Storage (SSD)**: 150GB free space
- **Operating System**: Latest version of Windows Server

For high-end use cases that require high data volumes, low latency and high amount of data pre-processing, it might be that additional computing resources are required (e.g. deploy one single high-performance Communication Instance on one dedicated computing device).

---

**Note**

Communication Instances store log files on the host system therefore sufficient storage needs to be provided.

---

## Test Environment Deployment Example

In a test environment the Manager and the Communication Instances can run on the same machine. Hardware configuration for running 20 communication instances as well as the Manager:

- **Computer and Processor**: 8 cores
- **Memory**: 16GB free memory
- **Storage (SSD)**: 100GB free space
- **Operating System**: Latest version of Windows Server

CHAPTER

TWO

# WINDOWS

SMART**UNIFIER** be delivered in two formats: as an **executable (.exe)** or as a **ZIP** archive.

## Install SMARTUNIFIER Manager (Archived Package)

Follow the steps below to install SMART**UNIFIER** Manager:

- Move the SMART**UNIFIER** installation package to a suitable location. Make sure the path to the directory does not include any white spaces!
- Extract the **.zip**-archive.
- Execute the **UnifierManager.bat** script. Afterwards the SMART**UNIFIER** Manager Console appears on the screen.
- Enter your **master password**. When starting SMART**UNIFIER** for the first time go to chapter: *Master Password and Administrator Account*.

After successfully starting up the SMART**UNIFIER** Manager, it can be accessed by opening an Internet Browser (e.g., Chrome or Firefox) and navigating to http://localhost:9000. Use the administrator credentials to login.

**Note**

The console is for information purposes only. It can be moved to any suitable location on your screen or it can be hidden. Nevertheless, do not close it, because the related processes will also be terminated.

## Install SMARTUNIFIER Manager as a Service

### Apache Procrun

SMART**UNIFIER** includes **Apache Procrun**, a Windows tool that facilitates the installation and execution of Java applications as services. It simplifies the process by integrating the application with the Windows Service Control Manager.

Follow the steps below to install and operate SMART**UNIFIER** Manager as a Service under Windows:

- Move the SMART**UNIFIER** installation package to a suitable location

- Ensure that the directory path does not contain any white spaces
- Extract the **.zip**-archive
- Open a terminal window with *Administrator privileges* within the installation package
- Execute the following commands in the terminal window to:

Listing 1: Install

```
UnifierManagerService.bat install
```

Listing 2: Start

```
UnifierManagerService.bat start
```

Listing 3: Stop

```
UnifierManagerService.bat stop
```

Listing 4: Uninstall

```
UnifierManagerService.bat uninstall
```

### NSSM

---

**Hint**

SMART**UNIFIER** does not offer official support for NSSM, unlike Apache Procrun. If you choose to use NSSM, you will need to download the NSSM binary separately.

---

Follow the steps below to install and operate SMART**UNIFIER** Manager as a Service under Windows using **NSSM**:

- Move the SMART**UNIFIER** installation package to a suitable location
- Ensure that the directory path does not contain any white spaces
- Download the latest version of NSSM to a suitable location (Last tested with version 2.24)
- Go to **win64** and copy the **nssm.exe** in the installation package
- Create the **UnifierManagerService.bat** file in the installation package

Listing 5: UnifierManagerService.bat

```
@echo off
cd %~dp0
set JAVA_HOME=%~dp0\jre
set JAVA=%JAVA_HOME%\bin\java.exe
set MANAGER=%~dp0\bin\adaptermanagerweb.bat
set JAVA_OPTS=-Dunifier.administrator.credentials.file="%~dp0/conf/
↪credentials.properties"
```

(continues on next page)

(continued from previous page)

```
del RUNNING_PID
"%MANAGER%"
```

- Open a terminal window with *Administrator privileges* within the installation package
- Execute the following commands in the terminal window to:

Listing 6: Install

```
nssm install SmartUnifierManager "UnifierManagerService.bat"
```

Listing 7: Start

```
nssm start SmartUnifierManager
```

Listing 8: Stop

```
nssm stop SmartUnifierManager
```

Listing 9: Uninstall

```
nssm remove SmartUnifierManager
```

**Optional**

Listing 10: Set SERVICE_AUTO_START

```
nssm set SmartUnifierManager Start SERVICE_AUTO_START
```

CHAPTER

# THREE

# LINUX

Follow the steps below to install SMART**UNIFIER** Manager:

- Move the installation package to a suitable location. Make sure the path to the directory does not include any white spaces!
- Extract the **.tar.gz**-archive.

```
tar -xvzf SmartUnifierManager-linux-x64.tar.gz
```

- Start the Manager by opening up a terminal and executing the following commands:

```
chmod +x UnifierManager.sh
```

```
./UnifierManager.sh
```

**Note**

Execute **StartUnifierManagerInBackground.sh** when the process should run in background. To stop the process execute **StopUnifierManager.sh**.

```
./StartUnifierManagerInBackground.sh
```

```
./StopUnifierManager.sh
```

- Enter your **master password**. When starting SMART**UNIFIER** for the first time go to chapter: *Master Password and Administrator Account*.

After successfully starting the SMART**UNIFIER** Manager, it can be accessed by opening an Internet Browser (e.g., Chrome or Firefox) and navigating to http://localhost:9000.

**Note**

The console is for information purposes only. It can be moved to any suitable location on your screen or it can be hidden. Nevertheless, do not close it, because the related processes will also be terminated.

CHAPTER

FOUR

# MAC OS

Follow the steps below to install SMART**UNIFIER** Manager:

- Move the installation package to a suitable location. Make sure the path to the directory does not include any white spaces!
- Extract the **.tar**-archive.
- Start the Manager by opening up a terminal and executing the following commands:

```
chmod +x UnifierManager.sh
```

```
./UnifierManager.sh
```

**Note**

If you get the warning "java cannot be opened because the developer cannot be verified" - go to `System Preferences... > Security & Privacy` and click on `Allow Anyway`.

- Enter your **master password**. When starting SMART**UNIFIER** for the first time go to chapter: *Master Password and Administrator Account*.

After successfully starting up the SMART**UNIFIER** Manager, the SMART**UNIFIER** Manager can be accessed by opening an Internet Browser (e.g., Safari, Chrome or Firefox) and navigating to http://localhost:9000.

**Note**

The console is for information purposes only. It can be moved to any suitable location on your screen or it can be hidden. Nevertheless, do not close it, because the related processes will also be terminated.

CHAPTER

FIVE

# DOCKER

## Requirements

The following example shows how to set up SMART**UNIFIER** using Docker Volumes mount to local paths on the machine.

1. Create the following directories:

**conf**
Manager configuration files, keystore and database

```
mkdir -p /opt/amorph/smartunifier/manager/conf
```

**repository**
Storing compiled artifacts

```
mkdir -p /opt/amorph/smartunifier/manager/repository
```

**versioning (optional)**
Storing of component sources, not required when using an external git server like gitea.

```
mkdir -p /opt/amorph/smartunifier/manager/versioning
```

**logs (optional)**
Storing of logs files from the manager

```
mkdir -p /opt/amorph/smartunifier/manager/log
```

2. Copy the **conf** and the **repository** folder from the SMART**UNIFIER** installation package into the newly created corresponding volumes:

```
cp -r conf/* /opt/amorph/smartunifier/manager/conf
cp -r repository/* /opt/amorph/smartunifier/manager/repository
```

**Note**

Edit the **application.conf** in **/opt/amorph/smartunifier/manager/conf** and remove the lines 'javaHome = "jre"'

```
nano /opt/amorph/smartunifier/manager/conf/application.conf
```

3. Create Docker Volumes mounted to the directories just created:

```
docker volume create --driver local \
 --opt type=bind \
 --opt device=/opt/amorph/smartunifier/manager/conf \
 --opt o=bind smartunifier_conf

docker volume create --driver local \
 --opt type=none \
 --opt device=/opt/amorph/smartunifier/manager/repository \
 --opt o=bind smartunifier_repository

docker volume create --driver local \
 --opt type=none \
 --opt device=/opt/amorph/smartunifier/manager/versioning \
 --opt o=bind smartunifier_versioning

docker volume create --driver local \
 --opt type=none \
 --opt device=/opt/amorph/smartunifier/manager/log \
 --opt o=bind smartunifier_logs
```

## Start Up

Go to the SMART**UNIFIER** package and open the **docker** directory with the console.

1. Build docker image

```
docker-compose build
```

2. Start the manager with attached console

```
docker-compose run smartunifier
```

3. Enter Master password and admin user credentials on request

---

**Note**

Remove the default user set up in the **credentials.properties** file in order to set the master password and to create a new admin user.

---

After the setup is done, a credentials file containing the master password can be used to start the manager without having to input the password.

```
docker-compose up -d
```

CHAPTER

SIX

# AMAZON ELASTIC COMPUTE CLOUD (EC2)

## Overview

SMART**UNIFIER** (SU) supports deployment using an Amazon EC2 Instance on the AWS Cloud. This guide steps you through the process of creating the needed AWS resources and the deployment of SMART**UNIFIER** Manager on an Amazon EC2 Instance.

### Cost and Licenses

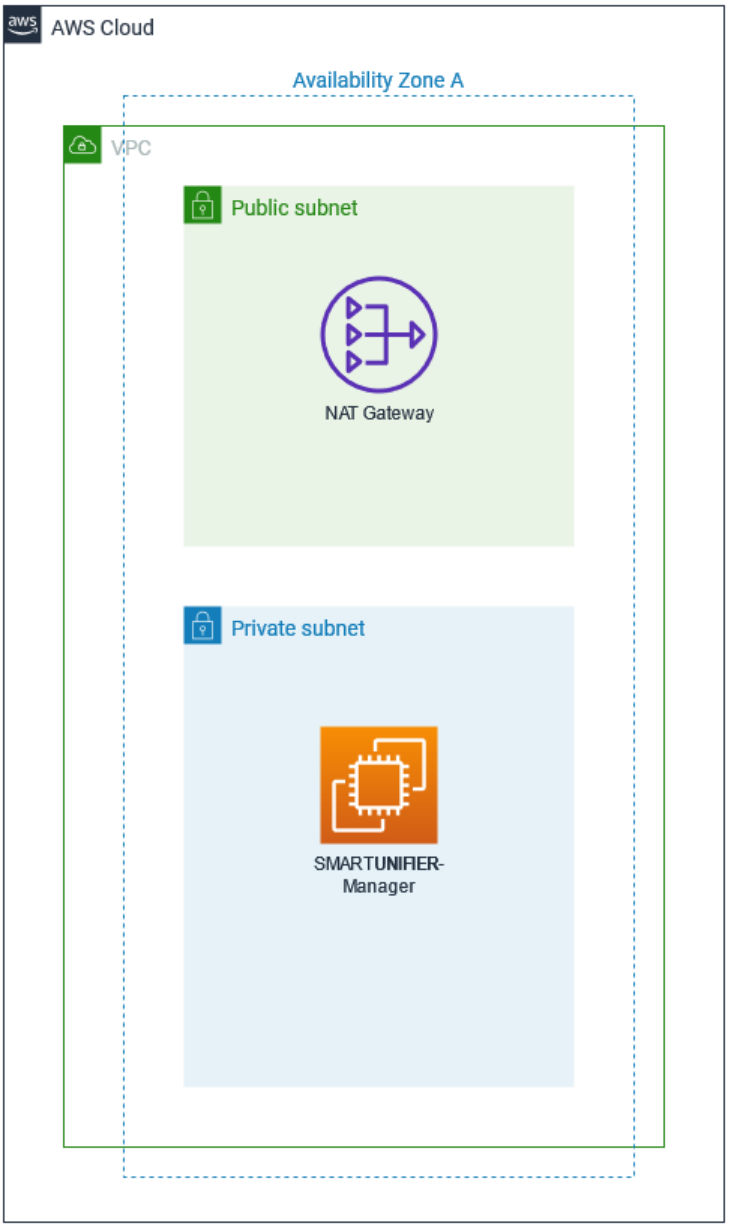
SMART**UNIFIER** charges you per running SU Instance. There are no charges for running the SMART**UNIFIER** Manager. If you don't have a licence yet, contact info@amorphsys.com.

You are responsible for the cost of AWS services used in the reference deployment of this guide. For cost estimates, see the pricing pages for each AWS service this guide is using. Some cost can be minimized e.g., by opting for a smaller Amazon EC2 Instance type.

## Architecture

This guide will set up the following SMART**UNIFIER** environment on AWS:

- A virtual private cloud (VPC) configured with one Availability Zones (AZ), with public and private subnet.
- Appropriate security groups for each instance of function to restrict access to only necessary protocols and ports.
- SMART**UNIFIER** Manager running on an EC2 Instance is located in the private subnet.




You can also deploy SMART**UNIFIER** into your existing VPC.

## Prerequisites

### AWS account

If you don't already have an AWS account, create one at https://aws.amazon.com by following the on-screen instructions. Your AWS account is automatically signed up for all AWS services. You are charged only for the used services.

### Specialized Knowledge

Before deploying and operating the SMART**UNIFIER** Manager on Amazon EC2, it is recommended that you become familiar with the following AWS services (If you are new to AWS, see Getting Started with AWS):

- Amazon Elastic Compute Cloud (EC2)
- Amazon Virtual Private Cloud (VPC)
- Amazon Elastic Block Store (EBS)

## Planning the Deployment

Before you deploy SMART**UNIFIER** on AWS, please review the following sections for guidelines on instance types, storage and high availability / disaster recovery.

### Deployment Options

There are two options available for the deployment:

- **Deploy** SMART**UNIFIER** into a *new* VPC (end-to-end deployment). This option builds a new AWS environment consisting of the VPC, subnets, security groups and other infrastructure components.
- **Deploy** SMART**UNIFIER** into an *existing* VPC. This option provisions SMART**UNIFIER** in your existing AWS infrastructure.

### Storage

We recommend using an **EBS volume** since it is persistent, even in the case of instance termination or crash. Using EBS volumes help to ensure high availability and durability for the instance.

- Volume Type: **General Purpose SSD (gp2)**.
- Size (GiB): **8**.

### Instance Selection

Which instance type to use depends on how much workload will be delivered to the instance. For information on instance types, see the AWS Website. The workload is dependent on the complexity of the integration scenario within SMART**UNIFIER** . It also can be measured based on the total amount of managed SU Instances.

| Instance type | SU Workload (Number of SU Instances) |
|---|---|
| t2.micro | <= 5 |
| t2.small | <= 20 |
| t2.medium | <= 100 |
| t2.large | > 100 |

## Deployment Steps

**Single-AZ Deployment creating new VPC - expected deployment time: 10-20 min**

### Step 1. Prepare Your AWS Account

1. If you don't already have an AWS account, create one at https://aws.amazon.com/ by following the on-screen instructions.
2. Select the AWS Region in the navigation bar where you want to deploy SMART**UNIFIER** on AWS.
3. Create a key pair in your preferred region.
4. If necessary, request a quota limit increase for the EC2 instance type that you've decided to deploy SMART**UNIFIER** on. You might need to do this if you already have an existing deployment that uses this instance type, and you think you might exceed the default quota.

### Step 2. Create Cloud Formation Stack

The Infrastructure for the deployment in a new VPC is provided in the CloudFormation template below.

1. Create a new AWS CloudFormation stack.
2. Select **Upload a template file** and upload the **yml-file** below then click next.
3. On the **Specify Page** enter a **Stack name** and configure the following **parameters** within the template:

Table 1: Parameters for deploying SMART**UNIFIER** Manager into a new VPC

| Parameter label | Default value | Description |
|---|---|---|
| VpcCIDR | 10.192.0.0/16 | CIDR block for the VPC |
| PublicSubnet1CIDR | 10.192.10.0/24 | CIDR block for the public subnet located in Availability Zone 1. |
| PublicSubnet2CIDR | 10.192.11.0/24 | CIDR block for the public subnet located in Availability Zone 2. |
| PrivateSubnet1CIDR. | 10.192.20.0/24 | CIDR block for the private subnet located in Availability Zone 1. |
| PrivateSubnet2CIDR. | 10.192.21.0/24 | CIDR block for the private subnet located in Availability Zone 2. |

1. Leave the default configuration on the **Options** page and select **Next**.
2. On the **Review** page, review and confirm the template settings.
3. Choose **Create stack** to deploy the stack.
4. When the status is **CREATE_COMPLETE**, the AWS infrastructure for SMART**UNIFIER** is ready.

### Step 3. Deploy SMARTUNIFIER Manager

1. Deploy SMART**UNIFIER** through the AWS Manangement Console.
   a. Open a web browser
   b. Go to the following URL: https://console.aws.amazon.com
2. In the AWS Manangement Console, select **Services -> EC2**
3. Click **Launch Instance** and follow the configuration below:
   a. Step 1: Enter a name for the instance
   b. Step 2: Search for the SMART**UNIFIER** AMI you want to launch
   c. Step 3: Select the *Instance Type*
   d. Step 4: Select the key-pair from *Step 1. Prepare Your AWS Account* to access the instance
   e. Step 5: Select the appropriate VPC and subnet
      - If you used the provided *SMARTUNIFIER Cloud Formation Template*, select the VPC and Subnet created by the stack
   f. Step 6: Verify that the storage settings are correct for the SMART**UNIFIER** instance. For all supported instance types the following storage settings are correct:
      - For the Root volume, 10 GB of General Purpose SSD
   g. Step 7: Review the instance settings
4. Connect to the instance by using the key pair created in *Step 1. Prepare Your AWS Account*
5. Start the SMART**UNIFIER** Manager

```
./SMARTUNIFIER/UnifierManager.sh
```

6. You can access SMART**UNIFIER** Manager on `https://private_or_public_IP_address:9000`

Listing 1: CloudFormation Template

```
Description:  This template deploys a VPC, with a pair of public and private␣
↪subnets spread
across two Availability Zones. It deploys an internet gateway, with a default
route on the public subnets. It deploys a pair of NAT gateways (one in each␣
↪AZ),
and default routes for them in the private subnets.

Parameters:
 EnvironmentName:
  Description: An environment name that is prefixed to resource names
  Type: String

VpcCIDR:
  Description: Please enter the IP range (CIDR notation) for this VPC
  Type: String
```

(continues on next page)

(continued from previous page)

```
  Default: 10.192.0.0/16

PublicSubnet1CIDR:
  Description: Please enter the IP range (CIDR notation) for the public␣
↪subnet in the first Availability Zone
  Type: String
  Default: 10.192.10.0/24

PrivateSubnet1CIDR:
  Description: Please enter the IP range (CIDR notation) for the private␣
↪subnet in the first Availability Zone
  Type: String
  Default: 10.192.20.0/24

Resources:
VPC:
  Type: AWS::EC2::VPC
  Properties:
    CidrBlock: !Ref VpcCIDR
    EnableDnsSupport: true
    EnableDnsHostnames: true
    Tags:
      - Key: Name
        Value: !Ref EnvironmentName

InternetGateway:
  Type: AWS::EC2::InternetGateway
  Properties:
    Tags:
      - Key: Name
        Value: !Ref EnvironmentName

InternetGatewayAttachment:
  Type: AWS::EC2::VPCGatewayAttachment
  Properties:
    InternetGatewayId: !Ref InternetGateway
    VpcId: !Ref VPC

PublicSubnet1:
  Type: AWS::EC2::Subnet
  Properties:
    VpcId: !Ref VPC
    AvailabilityZone: !Select [ 0, !GetAZs '' ]
    CidrBlock: !Ref PublicSubnet1CIDR
    MapPublicIpOnLaunch: true
    Tags:
      - Key: Name
        Value: !Sub ${EnvironmentName} Public Subnet (AZ1)
```

(continues on next page)

(continued from previous page)

```
PrivateSubnet1:
  Type: AWS::EC2::Subnet
  Properties:
    VpcId: !Ref VPC
    AvailabilityZone: !Select [ 0, !GetAZs  '' ]
    CidrBlock: !Ref PrivateSubnet1CIDR
    MapPublicIpOnLaunch: false
    Tags:
      - Key: Name
        Value: !Sub ${EnvironmentName} Private Subnet (AZ1)

NatGateway1EIP:
  Type: AWS::EC2::EIP
  DependsOn: InternetGatewayAttachment
  Properties:
    Domain: vpc

NatGateway1:
  Type: AWS::EC2::NatGateway
  Properties:
    AllocationId: !GetAtt NatGateway1EIP.AllocationId
    SubnetId: !Ref PublicSubnet1

PublicRouteTable:
  Type: AWS::EC2::RouteTable
  Properties:
    VpcId: !Ref VPC
    Tags:
      - Key: Name
        Value: !Sub ${EnvironmentName} Public Routes

DefaultPublicRoute:
  Type: AWS::EC2::Route
  DependsOn: InternetGatewayAttachment
  Properties:
    RouteTableId: !Ref PublicRouteTable
    DestinationCidrBlock: 0.0.0.0/0
    GatewayId: !Ref InternetGateway

PublicSubnet1RouteTableAssociation:
  Type: AWS::EC2::SubnetRouteTableAssociation
  Properties:
    RouteTableId: !Ref PublicRouteTable
    SubnetId: !Ref PublicSubnet1


PrivateRouteTable1:
  Type: AWS::EC2::RouteTable
  Properties:
```

(continues on next page)

(continued from previous page)

```
    VpcId: !Ref VPC
    Tags:
      - Key: Name
        Value: !Sub ${EnvironmentName} Private Routes (AZ1)

DefaultPrivateRoute1:
  Type: AWS::EC2::Route
  Properties:
    RouteTableId: !Ref PrivateRouteTable1
    DestinationCidrBlock: 0.0.0.0/0
    NatGatewayId: !Ref NatGateway1

PrivateSubnet1RouteTableAssociation:
  Type: AWS::EC2::SubnetRouteTableAssociation
  Properties:
    RouteTableId: !Ref PrivateRouteTable1
    SubnetId: !Ref PrivateSubnet1

NoIngressSecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupName: "no-ingress-sg"
    GroupDescription: "Security group with no ingress rule"
    VpcId: !Ref VPC

Outputs:
VPC:
  Description: A reference to the created VPC
  Value: !Ref VPC

PublicSubnet1:
  Description: A reference to the public subnet in the 1st Availability Zone
  Value: !Ref PublicSubnet1

PrivateSubnet1:
  Description: A reference to the private subnet in the 1st Availability Zone
  Value: !Ref PrivateSubnet1

NoIngressSecurityGroup:
  Description: Security group with no ingress rule
  Value: !Ref NoIngressSecurityGroup
```

### Step 4. Auto Scaling Group

After making sure that the SMART**UNIFIER** Manager is running as intended it's recommended to create an Auto Scaling group using the EC2 instance. With Amazon EC2 Auto Scaling your instance becomes more fault tolerance. In order to create an Auto Scaling group follow the steps described in the AWS documentation - Creating a launch template from an existing instance.

## Backups

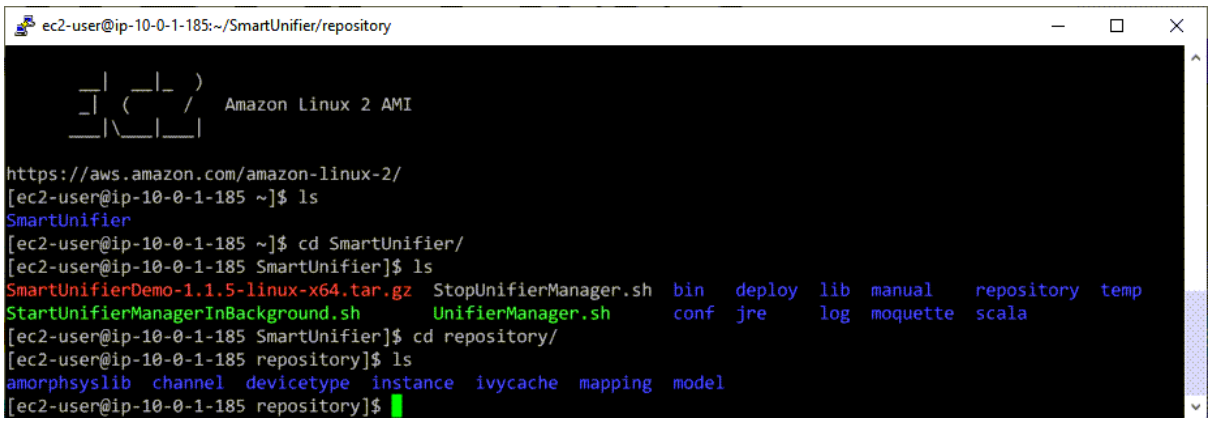
### Manager Repository and Database

#### Repository

Every artifacts (Information Models, Communication Channels, Mappings, Device Types and Instances) created with SMART**UNIFIER** are located in the **repository** directory.

We recommend to make regular backups of the repository. Follow the steps below in order to backup the repository:

- SHH into the EC2 instance, which is running SMART**UNIFIER** Manager.
- Change into the SMART**UNIFIER** Manager directory.
- Copy the **repository** directory to a suitable location.

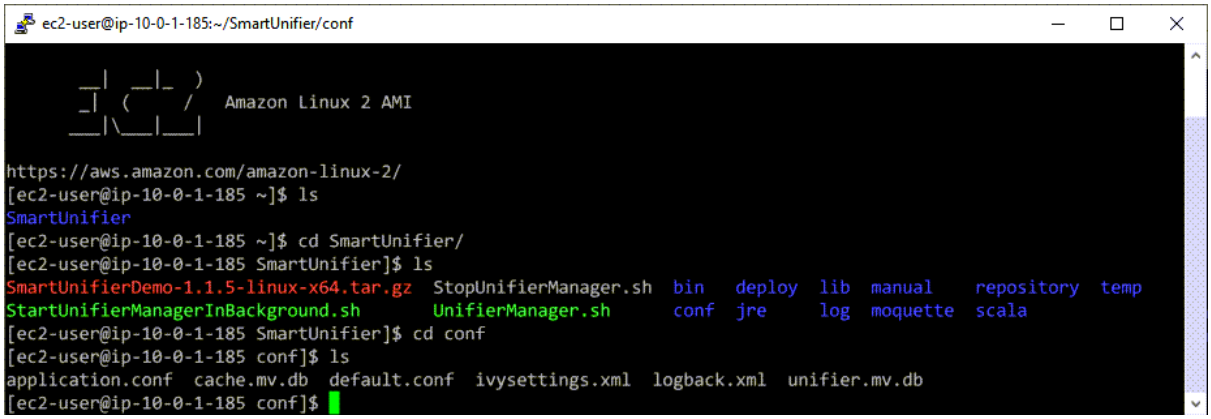



#### Database

Databases in SMART**UNIFIER** are used to store Deployment Endpoints and User Accounts. Credentials of users are stored in separate Keystore file.

It's recommended to make regular backups of the database **unifier.mv.db**. Follow the steps below to backup the database:

- SHH into the EC2 instance, which is running SMART**UNIFIER** Manager.
- Change into the SMART**UNIFIER** Manager directory.
- Change into the conf directory.
- You can now copy the **unifier.mv.db** database file to a suitable location.




### EC2 Snapshots

We recommend to create a Snapshot of the EBS volume which is attached to the EC2 instance that runs SMART**UNIFIER** . Follow the guide on how to create an Amazon EBS snapshot how to create an Amazon EBS snapshot.

If SMART**UNIFIER** is not in use and you don't want to be charged for EBS volumes, delete the EBS volume after creating a snapshot.

## Recovery

It's recommended to use Amazon CloudWatch with recover actions in order to recover instances in case of an instance failure.

CHAPTER

SEVEN

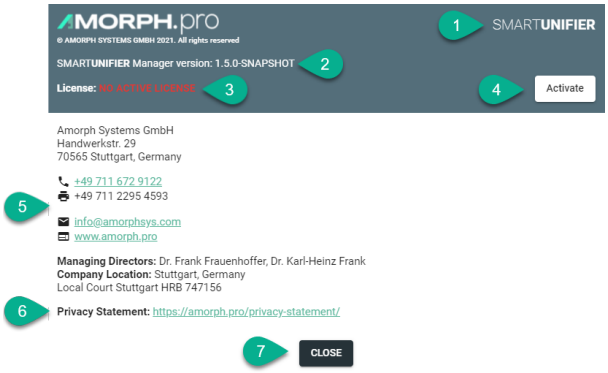
# PRODUCT INFORMATION AND ACTIVATION

## Product Information

To open the product information section click on the **Account** icon **(1)** and select the **About SMARTUNIFIER** section **(2)**.


The **About SMARTUNIFIER** section provides the following information:

1. Product name
2. Manager version
3. License information
4. Activation button
5. Company details
6. Privacy Statement URL




To exit the **About SMARTUNIFIER** section, click on the **Close** button **(7)**.

## Product Activation

The SMART**UNIFIER** product requires a license (demo/paid) for activation. The license details are displayed in the **About SMARTUNIFIER** section, as seen above.

The product activation can be done in two ways:

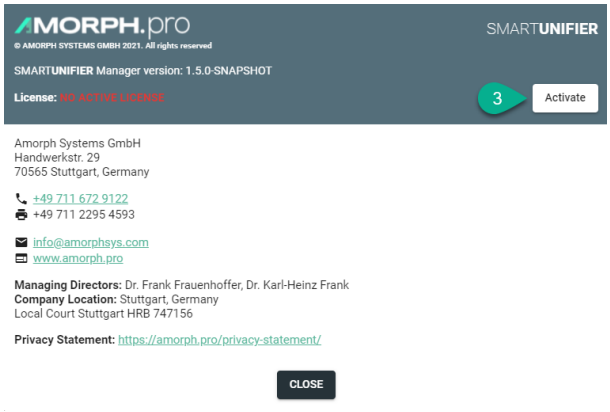
- Online
- Offline

### Online Activation

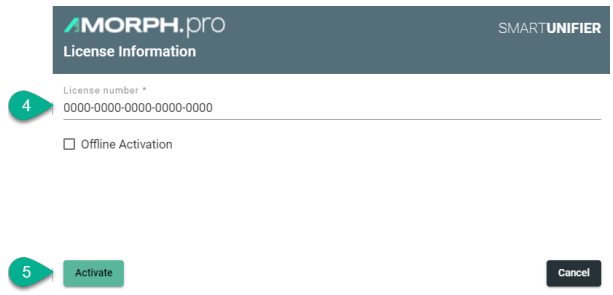
Follow the steps bellow to activate the product online:

- Click on the **Account** icon **(1)** and select the **About SMARTUNIFIER** section **(2)**

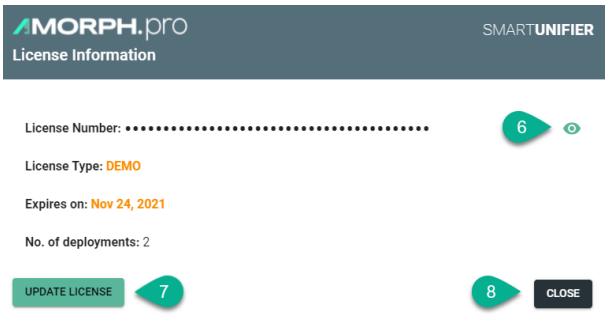

- Click on the **Activate** button **(3)**




- Input the key **license number (4)** and click on the **Activate** button **(5)**




- **The license key is registered, displaying the following details:**
  - License Number, visible by clicking on the **Show** button **(6)**
  - License Type
  - Expiration date
  - Maximum number of available deployments
- Click on the **Update License** button **(7)** to register a new license key or click on the **Close** button **(8)** to exit.

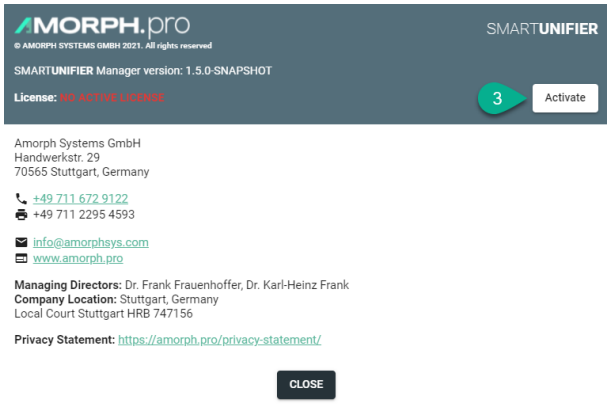


## Offline Activation

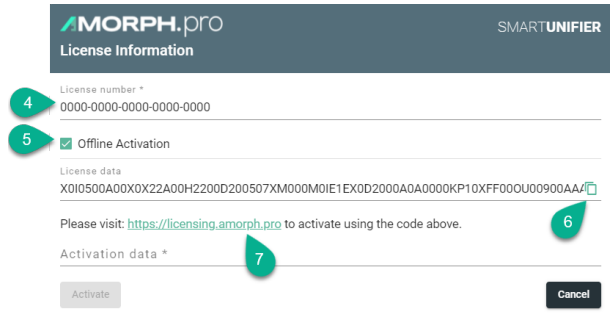
Follow the steps bellow to activate the product offline:

- Click on the **Account** icon **(1)** and select the **About SMARTUNIFIER** section **(2)**

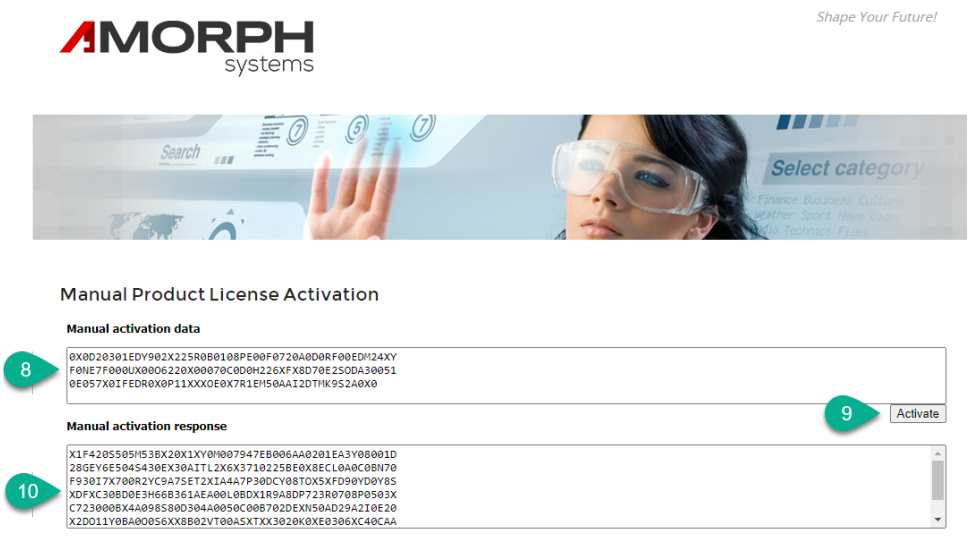

- Click on the **Activate** button **(3)**



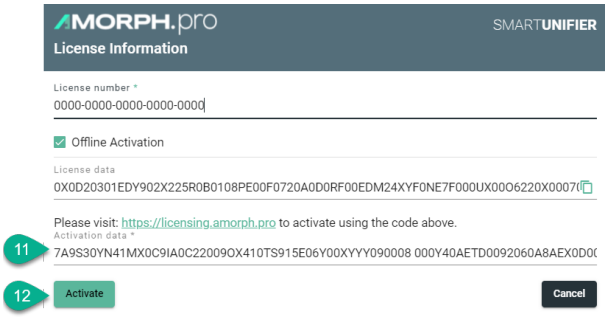

- Input the key **license number (4)** and check the box for **Offline Activation (5)**
- Copy the **License Data (6)** to an external storage unit



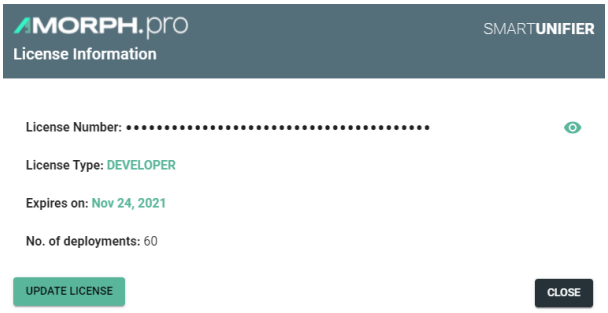

- From a device connected to the internet open the license URL **(7)**
- Paste the **License Data (6)** into the **Manual activation data** field **(8)** and click on the **Activate** button **(9)**




- Copy the **Manual activation response (10)** to an external storage unit and paste it into the **Activation data** field **(11)**




- Click on the **Activate** button **(12)** to finish. The license is registered, as seen bellow.

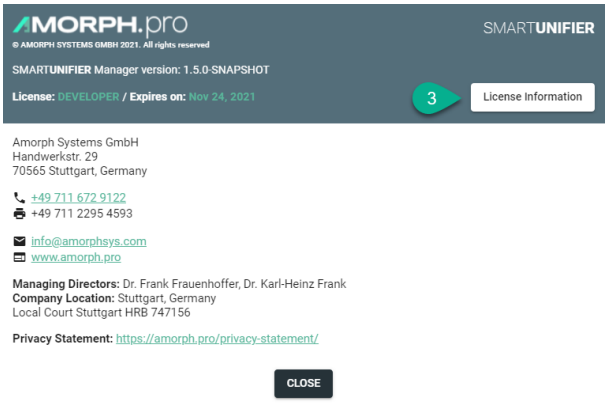


## Update License

Follow the steps bellow to update the license:

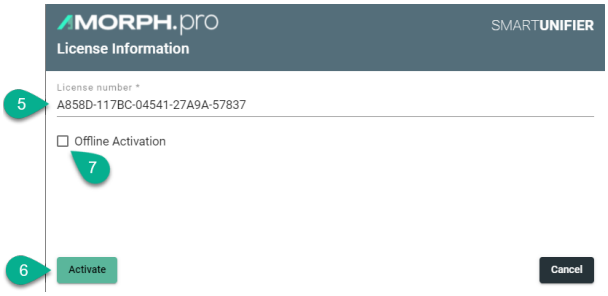
- Click on the **Account** icon **(1)** and select the **About SMARTUNIFIER** section **(2)**


- Click on the **License Information** button **(3)**




- Click on the **Update License** button **(4)**


- Input the key **license number (5)** and continue with *Online Activation* **(6)** or *Offline Activation* **(7)**


MORPH.pro
SMARTUNIFIER
License Information
License number *
A858D-117BC-04541-27A9A-57837
Offline Activation
5
6
7
Activate
Cancel

CHAPTER

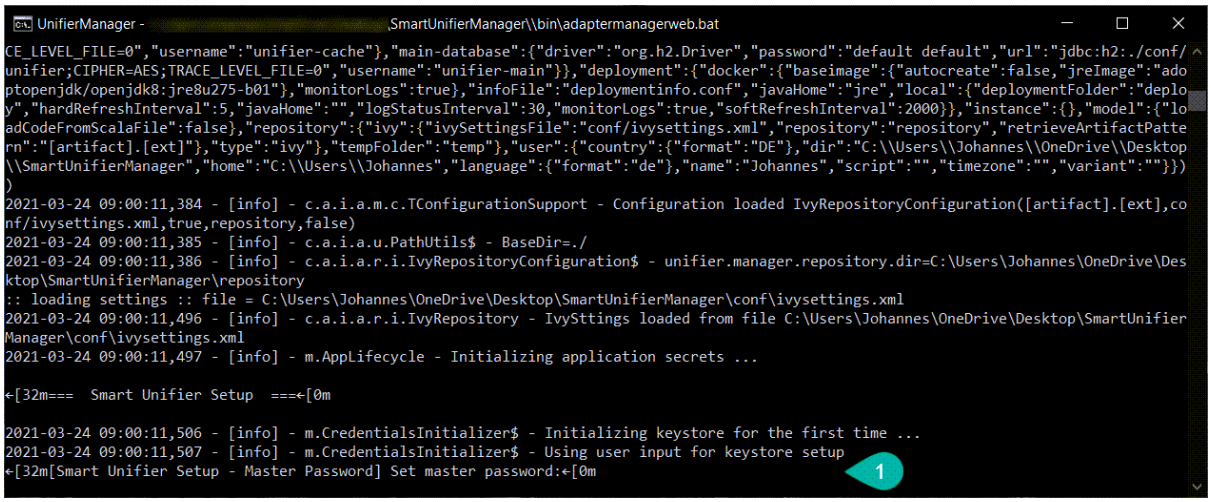
EIGHT

# CREDENTIALS MANAGEMENT

## Master Password and Administrator Account

When starting SMART**UNIFIER** for the first time you will be asked to enter a master password. The master password is needed in order to store credentials securely inside a KeyStore (”**unifier.jceks**”) file located on the user’s local machine.

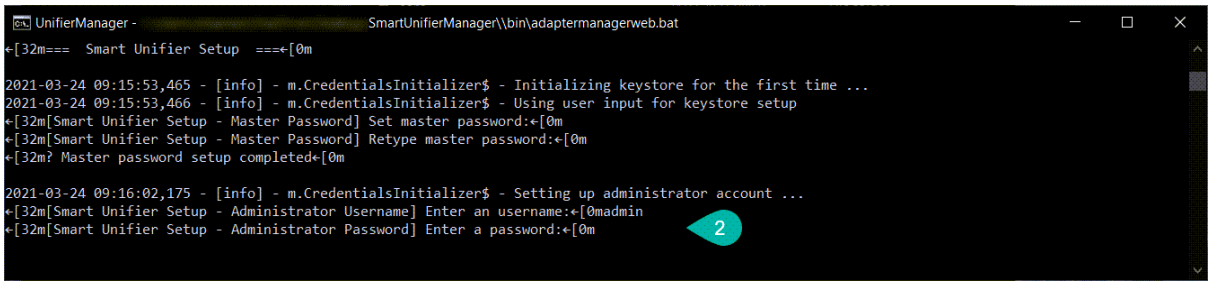
- Enter your master password in the console and re-enter it **(1)**. If the passwords do not match, simply close the console, execute the **UnifierManager.bat** and enter the passwords again.



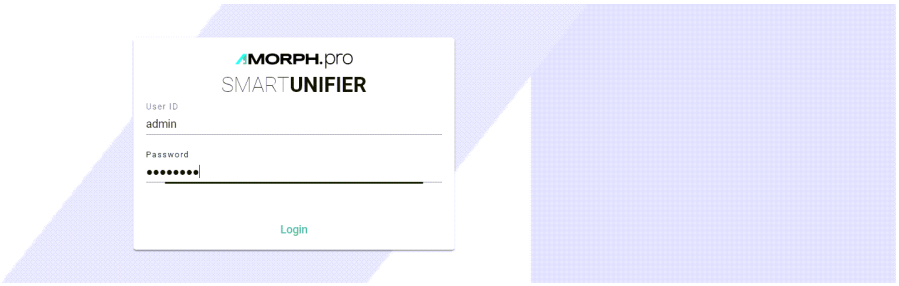

**Warning**

If the master password is lost it cannot be recovered!

- Enter the name for the administrator user account and the password **(2)**.



- Open an Internet Browser (e.g., Safari, Chrome or Firefox) and navigating to http://localhost:9000 and login using the administrator account just created.




You can add more users using the SMART**UNIFIER** Manager UI.

## Setting default credentials

You can define default credentials to avoid to re-enter the master password on startup.

1. Go to the **SmartUnifierManager** folder
2. Open the file **UnifierManager.bat** for a Windows installation (**UnifierManager.sh** for a installation on Linux/macOS)
3. Add the following line:

```
set JAVA_OPTS=-Dunifier.administrator.credentials.file="%~dp0/conf/
↪credentials.properties"
```

4. Make sure that the file **credentials.properties** exists in the **SmartUnifierManager/conf** folder
   - Set for **unifier.keystore.password** the master password as defined in the chapter *Master Password and Administrator Account*.

Listing 1: credentials.properties file content

```
# Keystore password
unifier.keystore.password=<keystore_password>

# Default Administrator account credentials
unifier.administrator.username=<administrator_username>
unifier.administrator.password=<administrator_password>
```

CHAPTER

NINE

# ENABLING HTTPS

Following configuration is required to enable https :

1. Browse to **SmartUnifierManager/conf** folder
2. Open **application.conf** for editing
3. Comment out (using #) following lines

```
play.server.http.port = 9000
play.server.http.address = "0.0.0.0"
```

4. Uncomment following lines and replace path_to_keystore and keystore_password with valid data

```
play.server.http.port=disabled
play.server.https.port=9443

play.server.https.keyStore.path="path_to_keystore"
play.server.https.keyStore.password="keystore_password"
```

5. Save and close

By default, keystore type is JKS. PEM. PKCS12 format is supported. In order to change the keystore type you need to add following configuration: **play.server.https.keyStore.type=PEM**

Generating a keystore is done using the following command:

```
keytool -keysize 2048 -genkey -alias unifier -keyalg RSA -keystore unifier.
↪keystore
```

- keysize 2048 sets the keystore size in bytes. The larger the storage, the more difficult it is to decipher an SSL key. Setting the keystore size to 2048 bytes is sufficient for high-level security.
- genkeypair generates a public key and an associated private key.
- alias unifier sets the alias for the SSL key; use this alias to reference keystore later, when configuring the application.
- keyalg RSA sets the encryption type for storage, which is RSA.
- keystore unifier.keystore, sets the name for the file into which the generated key will be written

Next, you will “fill in a questionnaire”. The data you provide is stored in the SSL key.

Once the keystore is created, you can generate a public SSL key. Recall the keystore password and run the following command (the terminal asks you to provide the correct password):

```
keytool -certreq -alias unifier -file unifier_csr.txt -keystore unifier.
↪keystore
```

- certreq generates a public SSL key (which has also the name Certificate Signing Request).-alias unifier sets the alias to refer to the key.
- file unifier_csr.txt creates a unifier_csr.txt file to store the key (this is different from the keystore).
- keystore unifier.keystore sets the key storage file.

You can skip this section if you are going to only test the HTTPS connection. However, if you are going to use the generated SSL key for production, you need to send it to a Certificate Authority.

Copy the SSL key that you can find in the home/johndoe/csr.txt file. **Note** that you must copy the entire contents of the file including the delimiters —-BEGIN NEW CERTIFICATE REQUEST—- and —-END NEW CERTIFICATE REQUEST—-. Without the delimiters, your key is not valid.

The SSL provider gives you two certificates in exchange for the key — the root and the intermediary certificates. (These certificates are called primary and secondary.) Add them both into the keystore.

Use the following command to add the intermediary certificate to the keystore:

```
keytool -importcert -alias secondary -keystore unifier.keystore -file <path_
↪to_secondary_certificate>.<ext>
```

- importcert tells the keytool library to import the certificates into storage.
- alias secondary sets the alias for the intermediary certificate.
- keystore unifier.keystore sets the necessary keystore for the certificate.
- file <path_to_intermediary_certificate>.<ext> sets the path to the file with the intermediary certificate.Remember to replace the <path_to_secondary_certificate> with the actual path; and also use the proper file extension instead of <ext>.

Similarly, you can add the root certificate to your storage, in this case you need to use a different command:

```
keytool -importcert -alias unifier -keystore unifier.keystore -trustcacerts -
↪file <path_to_root_certificate>.<ext>
```

CHAPTER

TEN

# EXTERNAL VERSION CONTROL

## Gitea

For the setup, make sure you meet the following prerequisites:

- A local installation of Gitea .
- An user account explicitly for SMART**UNIFIER** - `smartunifier`.
- An access token for this user account.

Once all prerequisites are met continue to authenticate SMART**UNIFIER** to access Gitea:

1. Go to the `application.conf` file that is located in the SMART**UNIFIER** package `SmartUnifierManager-windows-x64\conf`
2. Add the **sourceControl** JSON object inside the **unifiermanager** JSON object:

```
sourceControl {
  gitea = {
  baseUrl="**Enter Url here**"
  accessToken="**Enter access token here**"
  }
}
```

**Examples for configuration properties:**

| Property | Example |
|---|---|
| baseUrl | `http://localhost/api/v1` |
| accessToken | `8748ea571d0395434ee1a0a6f46163ba32d8c95e` |

## Local

Configuration of the local version control:

1. Go to the `application.conf` file that is located in the SMART**UNIFIER** package `SmartUnifierManager-windows-x64\conf`
2. Add the **sourceControl** JSON object inside the **unifiermanager** JSON object:

```
sourceControl {
  localgit {
  repoFolder = "**path to local direcotry**"
  }
}
```

CHAPTER

# ELEVEN

# EXTERNAL DATABASE

To connect SMART**UNIFIER** Manager to a remote database follow the steps below:

1. Go to the `application.conf` file that is located in the SMART**UNIFIER** package `SmartUnifierManager-windows-x64\conf`
2. Add the **database** JSON object inside the **unifiermanager** JSON object:

```
database {
  main-database {
      driver = "net.sourceforge.jtds.jdbc.Driver"
      url = "jdbc:jtds:sqlserver://<ip>:<port>;DatabaseName=unifier_db"
      username = "<username>"
      password = "<password>"
  }

  cache-database {
      driver = "net.sourceforge.jtds.jdbc.Driver"
      url = "jdbc:jtds:sqlserver://<ip>:<port>;DatabaseName=cache_db"
      username = "<username>"
      password = "<password>"
  }
}
```