

test.guide Operations Manual

tracetrone GmbH

Version 1.220.0, 2026-07-10

Table of Contents

1. Getting started	1
2. System requirements	6
2.1. Security Requirements	6
2.2. Hardware requirements	8
2.3. Software Requirements	9
2.4. File permissions	11
3. Recommended infrastructure	12
4. Notes on license	13
4.1. Local standard license	13
5. Run modes	14
5.1. Stand-Alone via batch file	14
5.2. Environment parameter configuration	14
5.3. Implementation as Linux-Service	15
5.4. Implementation as Windows-Service	16
5.5. Implementation with Docker	21
5.6. Configure	23
5.7. Implementation with Podman	24
5.8. SSL certificate configuration	25
5.9. Enabling Server Name Indication (SNI)	25
5.10. Adding self-signed root certificates for third party tool connections	26
6. Update procedures	27
6.1. Updating in Stand-Alone mode	27
6.2. Updating Linux-Service	27
6.3. Updating Stand-Alone Mode of the Windows-Service	28
6.4. Notes on migration path	28
7. Link with PostgreSQL database	29
7.1. Installing PostgreSQL	29
7.2. Security data advice / backup strategy	29
7.3. User creation	29
7.4. Database creation	30
7.5. Configure the database in test.guide	30
7.6. Configure the database via environment variables	31
7.7. Configuring and monitoring databases	32
8. Antivirus Plugin	34
9. Monitoring with Prometheus	35
10. test.guide configuration	36
10.1. Application settings	36
10.2. Report upload settings	39

11. Troubleshooting	41
11.1. Database issues	41
11.2. Upgrades	41
11.3. Error message "Could not acquire change log lock" during startup	41
11.4. Error message "Too many open files" on Linux operating systems	42
11.5. Failure on startup related to MonitoringDBManager or ArtifactDatabaseManager	42
11.6. Failure on startup "Maintenance migrations pending"	43

Chapter 1. Getting started

1

System design

It is possible to run test.guide with both an external database as well as an external file repository (e.g. a network drive). This way the system may be spread on up to three different machines.



To which degree the system needs to be spread on several machines depends on the expected number of simultaneous accesses and the expected data volume itself.

2

System requirements

The machine on which test.guide is meant to run, must at least match the system requirements. test.guide also requires a runtime environment installed (Java 17), but if you **use Docker or Podman (which we would recommend), then you don't need to install Java.**

Java installation (Java 17)	
Windows	Linux
Windows version by Eclipse Adoptium: Eclipse Adoptium/Temurin 17	Linux (Ubuntu and Debian) OpenJDK JRE: <code>sudo apt-get install openjdk-17-jre</code>

3

Preparing the installation directory for test.guide

To install the test.guide web application, the provided `TEST-GUIDE-1.220.0-enterprise-bin.zip` file must be extracted into an appropriate directory.

If [environment parameters](#) are left unchanged, test.guide will store all files in the directory `TTS-TM` in the user directory of the active user. If the directory needs to be set manually we recommend the following structure:

Example installation test.guide	
Windows	Linux
Create directory <code>C:\test.guide</code>	Create directory <code>/opt/test.guide</code>
Extract test.guide to <code>C:\test.guide\bin</code>	Extract test.guide to <code>/opt/test.guide/bin</code>
create test.guide Workspace <code>C:\test.guide\MyProjectWorkspace</code>	create test.guide Workspace <code>/opt/test.guide/MyProjectWorkspace</code>

4

Running test.guide

Setting up test.guide with docker or podman

We recommend running test.guide as a container application, e.g., with [Docker](#) or [Podman](#). This has the advantage of making setup and updates much easier and ensuring that all runtime dependencies are always up to date with the latest test.guide releases.

Here is a small Docker example:

Then use the [Docker command](#) (on Linux) to load the image into the Docker host:

```
docker load < TEST-GUIDE-1.220.0-ubi9.docker.tar.gz
```

Alternatively (after unpacking on Windows):

```
docker load --input TEST-GUIDE-1.220.0-ubi9.docker.tar
```

Create a volume to store its license key, logs, secret.txt etc. so that these changes are not lost after the reboot:

The container is then run with the following command:

```
docker container run -d \  
  --name=testguide \  
  --publish 1234:8085 \  
  --log-opt max-size=1m --log-opt max-file=3 \  
  --memory=16g \  
  --volume /opt/test.guide/MyProjectWorkspace:/var/testguide/TTS-TM \  
  --hostname testguide \  
  tracetronic/testguide-ubi9:1.220.0
```

At <http://localhost:1234> test.guide is then ready for setup after ~60sec.

Further information can be found in the chapter [Docker](#) or [Podman](#).

Setting up test.guide as service

Windows

There are two scripts to install and uninstall test.guide Windows-Services in the directory `windowsService`. Before these scripts can be run, `JAVA_HOME` has to be set or the paths to `java.exe` and `jvm.dll` must be set inside the script files. Additionally, the [same parameters that can be set for the manual start](#) may be set for the installation script. If an error occurs during the installation procedure, please run the uninstall script before trying to install again.



Admin rights are necessary to run the installation scripts; right-click the bat-file and choose "Run as administrator"!



Further information about Windows-Service can be found in [Chapter Implementation as Windows-Service](#).

Linux

The script `linuxService/installService.sh` allows setting up test.guide as Linux-Service (Debian, Ubuntu). The script allows to set the [same parameters as the manual start setup](#).



Admin rights are necessary to run the installation scripts, e.g. `sudo -E ./installService.sh`.



Further information about the Linux service can be found in [Chapter Implementation as Linux-Service](#).

Running test.guide manually

Before running test.guide via script on Windows (`startTTSTM.bat`) or Linux (`startTTSTM.sh`) the following parameters may be adjusted.

Parameter	Description	Default
JAVA_EXE	Path to java installation	java or %JAVA_HOME%\bin\java.exe
HTTP_PORT	Port to access test.guide via HTTP	8085
HTTPS_PORT	Port to access test.guide via HTTPS	0
TG_WORKSPACE	Set workspace dir of test.guide (used for logs, databases, ...)	%USERPROFILE%
TG_LIC_USER	Set license user	%USERNAME%



If special test.guide installation directories were created these [environment parameters](#) must be adjusted!

5

Database set up

After starting test.guide successfully, please navigate to the web interface of test.guide (i.e. <http://localhost:8085>) to complete the setup. The first step is to enter the character string from the file in the mentioned path. After doing so, a form is displayed to setup the database connection for test.guide.

The freely available Postgres database is required for setting up the database:

PostgreSQL installation	
Windows	Linux
Windows installation packages: https://www.postgresql.org/download/windows/	Linux (Ubuntu and Debian): <code>sudo apt-get install postgresql</code>

6

Setting up ServerAdmin

After the database connection has been configured, the password for the **ServerAdmin** account has to be set. For security reasons the password has to match the following criteria:

- minimum length: 8 characters
- at least one letter
- at least one number
- at least one special character

7

Request license

After setting up the **ServerAdmin** password successfully, a web page to enter the license is displayed. It also contains further advice.



The license key depends on the machines hardware and the (possibly customized) **user directory** used by test.guide. These parameters must be set correctly before requesting a license key.

8

Activating user authentication

Per default only the **ServerAdmin** has access to test.guide. However he can create new users and set their permissions.

9

What's next

Further setting options for email, authentication and much more can be found in the user help (e.g. <https://localhost:8443/help-docs>) under the point *Administering*.



Before going into production, it is important to take a look at the **test.guide configuration parameters**

Chapter 2. System requirements

2.1. Security Requirements

When setting up test.guide, the following security requirements apply:

Company IT security policy

test.guide must be set-up and operated in accordance with your company's applicable IT security policy at all times. Compliance with your organization's IT security policies is the responsibility of each test.guide user!

Infrastructure

It is important to plan, implement and document the infrastructure pursuant to your company's applicable security policy. Responsibilities, access rights, purposes and incident management will play particularly important roles here.

IT monitoring

- *Monitoring the systems:* Implement an IT monitoring system that monitors all relevant components of the test.guide IT infrastructure. This includes servers, networks, applications, databases and more.
- *Proactive error detection:* The monitoring should provide early warning of incidents such as denial-of-service (DoS) attacks before they lead to critical failures.
- *Real-time monitoring:* Ensure that you can respond to events and alerts in real time.

Data classification, processing of sensitive data

Your company's confidentiality classification of the data that you process in test.guide is an important factor when setting up and operating test.guide. Processing sensitive data requires setting up and using test.guide in accordance with appropriate security safeguards.

Encrypted connections

Ensure that test.guide is only operated using HTTPS with valid certificates and that all calls to external tools are also made via encrypted connections.

Certificates

The certificates used in the company must be kept up to date in test.guide.

Confidentiality and access rights

Make sure that only authorized persons have access to test.guide. Clarify which employees need access and what permissions they require.

Disaster plans

Follow disaster plans in the event of system failures, security breaches or other unexpected events.

Regular checks

Schedule regular reviews of security checks and test.guide configuration.

User training

Make your employees aware of how to set-up and use test.guide. Training can help to minimize security risks.

Updates of test.guide

The latest available test.guide version must be used at all times, as (only) the latest version will contain all available bug and security fixes.

Recommendation for intranet operation

As the data processed by test.guide is often sensitive, we strongly recommend operating test.guide exclusively on your company's intranet.

Internet operation

If you decide to operate test.guide on the Internet against our recommendation, it is your sole responsibility to consider, mitigate and/ or accept the associated security risks. Important aspects that must typically be considered in addition to the company's IT security policies may include: data privacy guidelines (please consult your company's data privacy officer), imprint obligations, liability vis-à-vis third parties, insurance, regulatory requirements, etc...



In case of doubt regarding any conflicts between these security requirements and the IT security policy or other policies of your company, you must please contact your responsible manager(s), your company's responsible IT Security officer(s) and inform tracetronic in writing (Email would suffice), so any such conflict can be resolved.

2.2. Hardware requirements

The Hardware requirements assume 250.000 test case executions per day, with up to 75 people working with the data in test.guide.

	Minimum system requirements	recommended system requirements
RAM	6 GB	8 GB or higher
CPU: Up-to-date multi-core architecture	Dual-Core	Quad-Core or higher
Bandwidth	100 MBit/s	1 GBit/s or higher
Free disc space (test.guide application)	500 MB (for installation, configuration and logs)	
disc space (database)	varies with amount of recorded data	
disc space (archiving activated)	varies with amount of recorded data	

Data base

It is not possible to make exact predictions on disc space in relation to the amount of recorded data. Therefore it is recommended to reserve 10 GB of disc space with an option to reserve more space when needed later on.

It is also recommended to run the database on a separate database server to reduce the load from the application server - check [Recommended infrastructure](#).

File repository

If file repository is enabled the transferred data is deposited in a uncompressed way at the configured file repository.

It is not recommended to host the file repository on the application server, instead we recommend using a dedicated file repository server to reduce writing processes on the application server - check [Recommended infrastructure](#).

Bandwidth

Especially in case archiving is enabled, the transferred data volume may be very high. Therefore, to allow quick processing of the data, a high network connection transfer rate is mandatory - check [Recommended infrastructure](#) once again.

2.3. Software Requirements

2.3.1. Supported Operating Systems (OS)

test.guide runs on the following operating systems:

- Linux Debian 9 / 10 / 11
- Linux Ubuntu 18.04 / 20.04 / 22.04
- Windows 10 / 11
- Windows Server 2016 / 2019 / 2022

test.guide requires a 64-bit operating system.

2.3.2. Supported Container Environments

test.guide runs in the following [container environments](#):

- minimal *Docker Engine 20.10.15* version
- or minimal *podman 4.3.0* version

2.3.3. Supported Java Runtime Environments

test.guide runs on the following Java Runtime Environments:

- [Eclipse Adoptium/Temurin 17](#)
- [Zulu JRE 17](#)
- [OpenJDK 17](#)
- [Oracle JDK 17](#)

test.guide requires a 64-bit Java Runtime Environment.

2.3.4. Database

It is **required** to run the application in combination with:

- [PostgreSQL Database Version >=13](#)



Exactly one database per application instance must be accessible on the database server.



A guide to set up a PostgreSQL database can be found in chapter: [Link with PostgreSQL database](#).

Database data security / backup strategy



Prior to installing and/ or running test.guide, access and processing of data within

test.guide must be qualified, setup and operated pursuant to the applicable internal company policies of the user's/ customer's organization; in case of any questions or concerns in connection with these internal company policies, the user must contact the user's responsible internal supervisor. tracetronic is not privy to these internal company policies and related information and no information provided by tracetronic must be construed as any recommendation or decision with respect to these internal company policies.

Prior to installing and/ or running test.guide, a database backup must be configured by the user/ customer in accordance with the applicable internal company policies of the user's/ customer's organization. test.guide must not be operated without appropriate and effective database backup and recovery strategies. Database backup and recovery are critical, as no software can ever be programmed, installed and/ or operated error-free, so errors (including, but not limited to errors within test.guide) can occur and may compromise data.

Subject to the applicable company policies of the user's/ customer's organization, it is recommended to establish internal recovery service level agreements (SLAs) for test.guide within the user's/ customer's organization, which may define acceptable risks, recovery times and data access conditions; please note that test.guide database will be offline in case of a recovery for a certain period of time (depending on the amount of data), which must be factored in the user's/ customer's database backup and recovery strategies.

Hardware requirements the database server

	Minimum system requirements	Recommended requirements
RAM	4 GB	8 GB or higher
CPU: Up-to-date multi-core architecture	Dual-Core	Quad-Core or higher
disc space for database	varies with amount of recorded data	

2.3.5. Firewall configuration

test.guide must be able to access some network ports, depending on its configuration. By default test.guide uses HTTP port 8085 for network communication. This port may be customized, additionally there may be a HTTPS port in use as well.



When using test.guide in a cluster scenario (using multiple application instances accessing the same underlying database), to ensure data consistency, the test.guide instances need to communicate using additional network ports for cache coordination. Therefore, the application will attempt to open ephemeral TCP ports in the range 50101 to 50601. This port range must be considered when configuring the firewall.

2.3.6. Browser

Only up-to-date browsers with JavaScript activated are supported for web access.

Officially supported browsers:

- Microsoft Edge from version 84
- Mozilla Firefox from version 78
- Google Chrome from version 84

2.4. File permissions

2.4.1. Used directories

- The application's log and license files as well as the configurations file are stored in the designated user directory under *TTS-TM*.



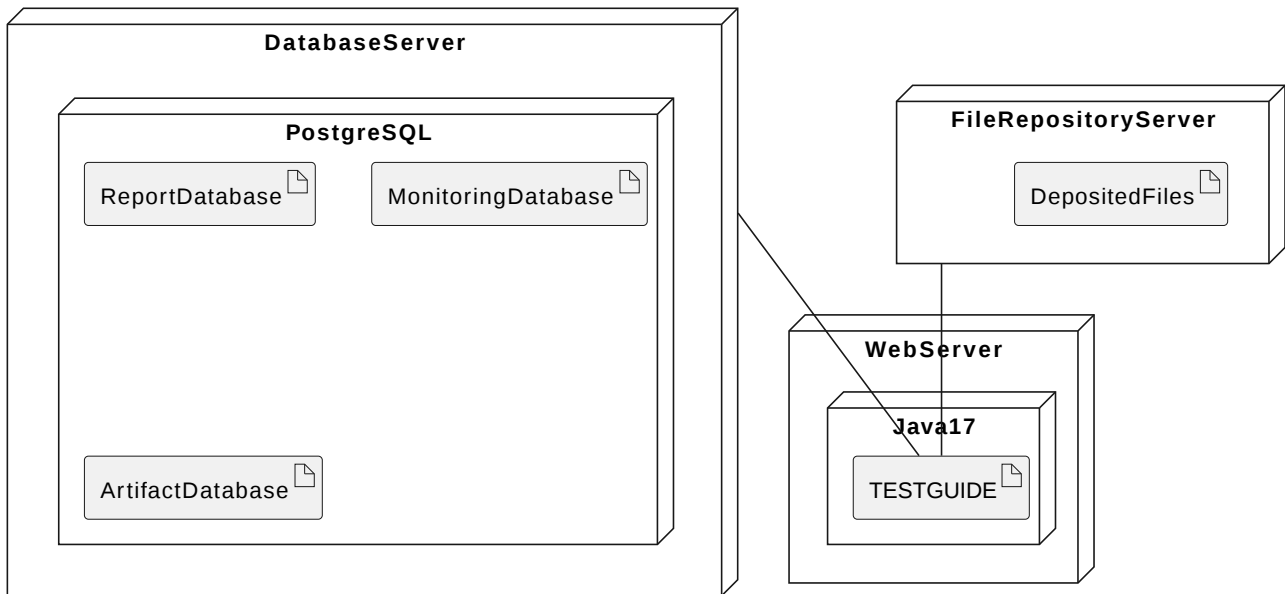
The path to the current working directory is displayed in test.guide under *System configuration* → *System status*.

- Furthermore, the user's temporary directory is used to verify uploaded files, these are deleted afterwards.

2.4.2. Required permissions

To use the archiving option the active user must have write permissions in the configured directory.

Chapter 3. Recommended infrastructure



It is recommended to reserve three separate servers for the following components:

- [Web server \(to run the application itself\)](#)
- [Database server](#)
- File repository server

Web server, file repository server and database server must be able to communicate via a network connection with at least 1 GBit/s bandwidth to guarantee best performance.

Chapter 4. Notes on license

4.1. Local standard license

The license is created on dependency of the [user directory](#). Therefore the user directory and the user must be set before the license is requested. When [running the application](#) the license page gets displayed, providing all the information necessary for obtaining a valid license.



If the license is expired the applications backend continues working and accepts new data. However, there is no access to the user interface, hence no way of analyzing the data as well - until the license is renewed and the user interface is de-blocked.

Chapter 5. Run modes

test.guide contains an integrated web server and can be run either as Java application or Docker/podman container.

5.1. Stand-Alone via batch file

The included batch file *startTTSTM.bat* runs the integrated server. Please note that the parameters in the batch file must be set correctly.

Example: `java.exe -jar enterprise.war 8085 0`

Command line parameters in shown order:

- HTTP port number for the connection
- HTTPS port number to connect with a valid SSL certificate, set to 0 if HTTPS should not be used.



If HTTPS is active, all queries on test.guides API must be switched to HTTPS as well.



The integrated server can be installed as [Windows-Service](#).

5.2. Environment parameter configuration

Setting up working directory

The application saves all configuration files, log files and the internal file based databases in the directory "TTS-TM" (see [Used directories](#)) inside the user directory of the user running test.guide (on Windows: `C:\Users\<name of active user>`). It is possible to customize the user directories that are passed on to JVM with the following parameter:

```
TG_WORKSPACE=C:\test.guide\MyProjectWorkspace
```



The workspace is also important for your license do not change after first installation!



When using the Docker image the directory is fixed to `/var/testguide` and cannot be changed with the `TG_WORKSPACE` environment variable.

Passing arguments to the JVM

Should it be necessary to add additional parameters to the JVM that runs test.guide, the environment variable `JDK_JAVA_OPTIONS` can be used. Specifying an HTTP proxy server is a typical use case for this, as shown below:

```
JDK_JAVA_OPTIONS="-Dhttp.proxyHost=proxy.company.org -Dhttp.proxyPort=8080
```

5.3. Implementation as Linux-Service

5.3.1. Installation

`test.guide` with `stand-alone Server` can be run as Linux-Service. The shell script "installService.sh" can be found in the directory `linuxService`.



Currently the contained installation script only supports the service administration system `systemd`.

The parameters needed by "installService.sh" can be set at the beginning of the script file itself or declared as environment variables before running the script. Some parameters have defaults set in the script (i.e. http and https ports, temp directory), the remaining parameters can be set interactively (i.e. JAVA_HOME, user). It is recommended to also set these parameters inside the script file, so they are available when re-running the script is necessary (e.g. after updating test.guide).



If the JAVA_HOME variable is not set, you can also determine the path with the following command: `update-alternatives --list java`

The script uses the following environment variables:

Description:	environment variable	Notes
Path to Java Runtime Environment	TG_JAVA_HOME	If JAVA_HOME was already set, the value can be applied interactively
user name that runs test.guide	TG_LIC_USER	can be set interactively if not set
Port on which test.guide is listening	HTTP_PORT	default: 8085
HTTPS usage	HTTPS_PORT or 0	default: 0

To alter the working directory the `environment parameters` must be changed at the beginning of the script.

We recommend to run test.guide on an unprivileged port (≥ 1024). If you want to provide test.guide on a privileged port, like port 80 or port 443, then you can do one of the following things:

- Redirect the ports, e.g. via iptables
- Selectively allow privileged ports, e.g. via CAP_NET_BIND_SERVICE or authbind
- Run test.guide as root (not recommended)

The script must be run with root permissions.



If you can't run the shell script, check if the permission for the script is correct, using `ls -a -l` and `chmod`.

Run in command line as follows:

```
sudo -E ./installService.sh
```

The script may require some yes and no feedback from the user. The questions may be answered with "y" and "n" respectively, or left blank for using the default option (which is shown in uppercase letters).

5.3.2. Stopping / Restarting / Deactivating the Linux-Service

The installation script creates a systemd unit file "TEST-GUIDE.service", which is by default stored in "/etc/systemd/system" and activates this unit. The test.guide service is now ready to be activated, deactivated, started, restarted or stopped with the systemd standard commands.

```
# activate
sudo systemctl enable TEST-GUIDE.service
# deactivate
sudo systemctl disable TEST-GUIDE.service
# start
sudo systemctl start TEST-GUIDE.service
# stop
sudo systemctl stop TEST-GUIDE.service
# restart
sudo systemctl restart TEST-GUIDE.service
```

The current status of the test.guide service can be requested by using this command:

```
systemctl status TEST-GUIDE.service
```

5.3.3. Changing parameters of the service

To change the parameters of the service the installation script can be re-run. The systemd unit file will be updated.

5.4. Implementation as Windows-Service

[test.guide with integrated server](#), as well as the ResourceAdapter can be installed as Windows service. The relevant batch files are located in the *windowsService* directory.



The service can be used on all **64-Bit-Windows-Systems** with a **Java 64-Bit-Version**.

5.4.1. Installing the Windows-Services

Installing the test.guide-Service

Some changes to *installService.bat* are necessary before it is run. The following paths must be set:

- `JAVA_EXE_:` Path to `java.exe` (e.g. "C:\Program Files\Eclipse Adoptium\jdk-17.0.7-hotspot\bin\java.exe")
- `JAVA_JVM_:` Path to `jvm.dll` (e.g. "C:\Program Files\Eclipse Adoptium\jdk-17.0.7-hotspot\bin\server\jvm.dll")

Optional: Adjusting test.guide working directory

If the working directory has been changed or needs to be adjusted, as per chapter [Environment Parameter Configuration](#) the following line must be modified accordingly:

```
SET TG_WORKSPACE=C:\test.guide\MyProjectWorkspace
```

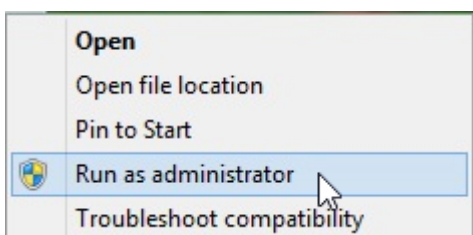
Firewall configuration for test.guide Service

Please ensure that the firewall is not blocking the used ports (i.e. *installService.bat* Parameter *HTTP_PORT* and *HTTPS_PORT*) for the Windows-Service. Sometimes the following processes that start test.guide must be allowed in the firewall:

- *prunsrv.exe*
- *java.exe*

Installation

installService.bat must be run as administrator to install the test.guide service. Right-click the .bat file and select 'Run as admin' in the context menu.



A windows command line window (cmd) opens.

```
C:\Windows\System32\cmd.exe

#####
#                                     #
#                               TEST-GUIDE                               #
#                         Winodws Service Installation                  #
#                                     #
#####

=== JAVA-VERSION ===
java version "1.8.0_91"
Java(TM) SE Runtime Environment (build 1.8.0_91-b14)
Java HotSpot(TM) 64-Bit Server VM (build 25.91-b14, mixed mode)

=== TEST-GUIDE SETTINGS ===
working directory ..... C:\dev\work
server port ..... 8085
use https ..... False

=== SERVICE ACCOUNT ===
Please enter a valid windows account that executes the service.

Note: It is not possible to have one or more quotes (") in your password.
      If you have quotes in your password, please set the password under
      control panel / services / TEST-GUIDE-Service
      and let the following fields empty.

Username (domain\username):
```

The prompt requests user name and password of a Windows user that is set up on the system. This user will be used to run the service. Unfortunately, these user details cannot be verified at this point. If user name or password were entered incorrectly, they must be corrected in *Control Panel/Administrative Tools/Services*.



It is not possible to enter the password at this point if it contains quotation marks ("). If this is the case, it must also be entered in *Control Panel/Administrative Tools/Services*. After the service has been started, test.guide will create a directory called "TTS-TM" in the user's user directory. This is used by test.guide as working directory.

```
C:\Windows\System32\cmd.exe

#####
#                                     #
#                               TEST-GUIDE                               #
#                         Winodws Service Installation                  #
#                                     #
#####

=== JAVA-VERSION ===
java version "1.8.0_91"
Java(TM) SE Runtime Environment (build 1.8.0_91-b14)
Java HotSpot(TM) 64-Bit Server VM (build 25.91-b14, mixed mode)

=== TEST-GUIDE SETTINGS ===
working directory ..... C:\dev\work
server port ..... 8085
use https ..... False

=== SERVICE ACCOUNT ===
Please enter a valid windows account that executes the service.

Note: It is not possible to have one or more quotes (") in your password.
      If you have quotes in your password, please set the password under
      control panel / services / TEST-GUIDE-Service
      and let the following fields empty.

Username (domain\username): musterdomain\MustermannMax
Password: *****
```

Once the password is confirmed, the user will be provided with feedback on the success or failure of the installation. After successful installation of the service, the user may choose to start the service. Possible selections are:

- Y = start service
- N = do not start service

```

C:\Windows\System32\cmd.exe

#####
#                                     #
#                               TEST-GUIDE                               #
#                         Winodws Service Installation                 #
#                                     #
#####

=== JAVA-VERSION ===
java version "1.8.0_91"
Java(TM) SE Runtime Environment (build 1.8.0_91-b14)
Java HotSpot(TM) 64-Bit Server VM (build 25.91-b14, mixed mode)

=== TEST-GUIDE SETTINGS ===
working directory ..... C:\dev\work
server port ..... 8085
use https ..... False

=== SERVICE ACCOUNT ===
Please enter a valid windows account that executes the service.

Note: It is not possible to have one or more quotes (") in your password.
      If you have quotes in your password, please set the password under
      control panel / services / TEST-GUIDE-Service
      and let the following fields empty.

Username (domain\username): musterdomain\MustermannMax
Password: *****

=== INSTALL SERVICE ===

TEST-GUIDE-Service successfully installed!

Do you want to start the service now? [Y/N]?

```

When selecting Y a further notice will be prompted, telling the user if the service was started. After this the installation is complete.

```

C:\Windows\System32\cmd.exe

#####
#                                     #
#                               TEST-GUIDE                               #
#                         Winodws Service Installation                 #
#                                     #
#####

=== JAVA-VERSION ===
java version "1.8.0_91"
Java(TM) SE Runtime Environment (build 1.8.0_91-b14)
Java HotSpot(TM) 64-Bit Server VM (build 25.91-b14, mixed mode)

=== TEST-GUIDE SETTINGS ===
working directory ..... C:\dev\work
server port ..... 8085
use https ..... False

=== SERVICE ACCOUNT ===
Please enter a valid windows account that executes the service.

Note: It is not possible to have one or more quotes (") in your password.
      If you have quotes in your password, please set the password under
      control panel / services / TEST-GUIDE-Service
      and let the following fields empty.

Username (domain\username): musterdomain\MustermannMax
Password: *****

=== INSTALL SERVICE ===

TEST-GUIDE-Service successfully installed!

Do you want to start the service now? [Y/N]? Y
=== START SERVICE ===

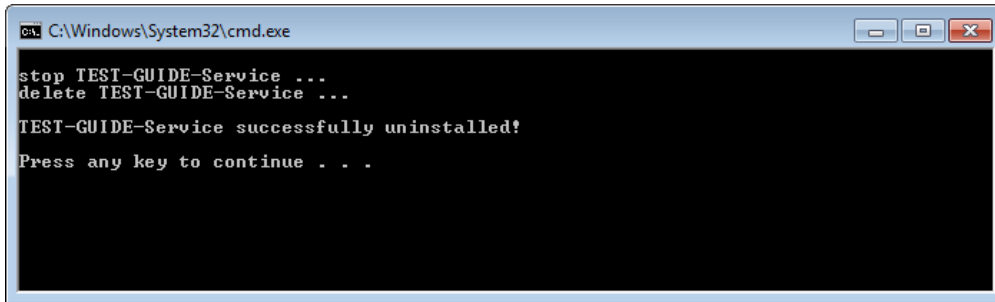
TEST-GUIDE-Service successfully started!

Press any key to continue . . .

```

Uninstallation

To uninstall the test.guide service the file *uninstallService.bat* must be run as administrator. This may take a couple of seconds because the service must be stopped first. A prompt will give feedback if the uninstallation was successful.



```
C:\Windows\System32\cmd.exe
stop TEST-GUIDE-Service ...
delete TEST-GUIDE-Service ...
TEST-GUIDE-Service successfully uninstalled!
Press any key to continue . . .
```

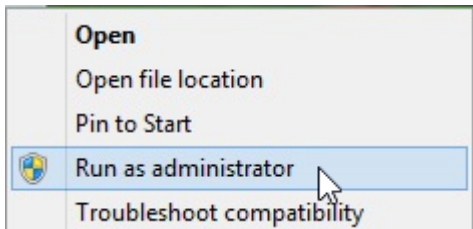
Install the test.guide-Resource-Adapter-Service

Installation and uninstallation

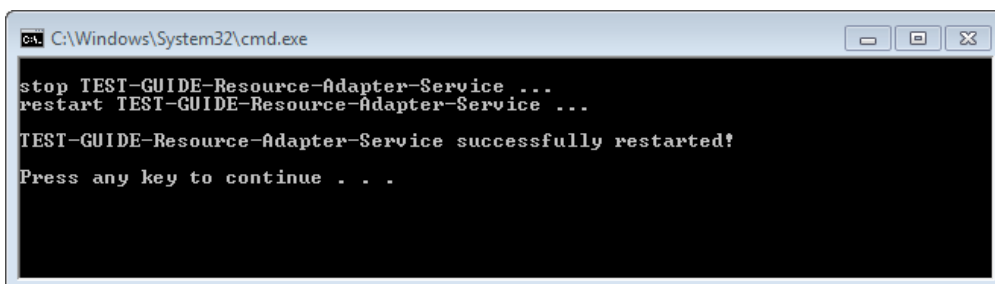
Installation and uninstallation of the test.guide-Resource-Adapter-Service is largely carried out in the same steps as described above under *Installing the test.guide-Service*. Only the working directory cannot be changed for the ResourceAdapter, so setting the parameter `TG_WORKSPACE` has no effect.

Restart

The contained *restartService.bat* is needed to restart the Resource-Adapter when the configuration file has been modified. A restart is required for the test.guide-Resource-Adapter-Service to honor the changed settings. *restartService.bat* must be run as administrator.



The progress of the process is prompted in the cmd console.



```
C:\Windows\System32\cmd.exe
stop TEST-GUIDE-Resource-Adapter-Service ...
restart TEST-GUIDE-Resource-Adapter-Service ...
TEST-GUIDE-Resource-Adapter-Service successfully restarted!
Press any key to continue . . .
```

5.5. Implementation with Docker

5.5.1. Initialization of the setup

Deploy or download a new release e.g. *TEST-GUIDE-1.220.0-ubi9.docker.tar.gz*

Then use the [Docker command](#) (on Linux) to load the image into the Docker host:

```
docker load < TEST-GUIDE-1.220.0-ubi9.docker.tar.gz
```

Alternatively (after unpacking on Windows):

```
docker load --input TEST-GUIDE-1.220.0-ubi9.docker.tar
```

Create a volume to store its license key, logs, secret.txt etc. so that these changes are not lost after the reboot:

The container is then run with the following command:

```
docker container run -d \  
  --name=testguide \  
  --publish 1234:8085 \  
  --log-opt max-size=1m --log-opt max-file=3 \  
  --memory=16g \  
  --volume /home/<username>/docker/TTS-TM:/var/testguide/TTS-TM \  
  --hostname testguide \  
  tracetronic/testguide-ubi9:1.220.0
```

In the directory */home/<username>/docker/TTS-TM* test.guide stores its license key, logs, secret.txt etc. so that these changes are not lost after a reboot.

At <http://localhost:1234> test.guide is then ready for setup after ~60sec.

As a liveness probe and readiness probe <http://localhost:1234/api/health/live> respectively <http://localhost:1234/api/health/ready> can be used.

Please read the [Configure](#) section on how to customize the setup.

5.5.2. Update

Deploy or download a new release e.g. *TEST-GUIDE-1.220.0-ubi9.docker.tar.gz*

Then load the image into the Docker host (on Linux):

```
docker load < TEST-GUIDE-1.220.0-ubi9.docker.tar.gz
```

Alternatively (after unpacking on Windows):

```
docker load --input TEST-GUIDE-1.220.0-ubi9.docker.tar
```

Stop the execution of the old container and remove the old container:

```
docker container stop testguide  
docker container rm testguide
```

Then restart the execution of the new container:

```
docker container run -d \  
  --name=testguide \  
  --publish 1234:8085 \  
  --log-opt max-size=1m --log-opt max-file=3 \  
  --memory=16g \  
  --volume /home/<username>/docker/TTS-TM:/var/testguide/TTS-TM \  
  --hostname testguide \  
  tracetrone/testguide-ubi9:1.220.0
```

5.6. Configure

Following environment variables can be set:

Variable	Default	Description
TESTGUIDE_CSRF_PROTECTION	true	Set this to true to prevent Cross-Site-Request-Forgery attacks
TESTGUIDE_CSRF_ACCEPTED_ORIGINS	""	Allows whitelisting of specific domains / host names. Requests from these domains are not scanned for possible CSRF attacks, even if CSRF protection is enabled. Multiple domains can be whitelisted (separated by comma character), e.g. <code>local.testguide,test-guide.de</code>
TESTGUIDE_SECURE_COOKIES_OVER_HTTP	false	Set this to true to enable the 'secure' flag on cookies even when HTTP is used.
TESTGUIDE_MAX_RAM_PERCENTAGE	50	Sets JVM options <code>-XX:InitialRAMPercentage</code> and <code>-XX:MaxRAMPercentage</code>
ENABLE_SNI	<not set>	See Enabling Server Name Indication (SNI)
<code>JDK_JAVA_OPTIONS</code>	<not set>	Additional JVM arguments

The container exposes the HTTP service at TCP port 8085 by default. This port can be bound to any host port with the `-p` option of Docker.

At the container directory `/var/testguide/TTS-TM` testguide stores data that should be persistent over restarts. It is recommended to mount this directory to a local directory or a Docker volume.

Here is an extended example:

```
docker container run -d \  
  --name=testguide \  
  --publish 1234:8085 \  
  --log-opt max-size=1m --log-opt max-file=3 \  
  --memory=16g \  
  --volume $HOME/docker/TTS-TM:/var/testguide/TTS-TM \  
  --volume $JAVA_HOME/lib/security/cacerts:/lib/jvm/jre/lib/security/cacerts \  
  -e TESTGUIDE_CSRF_PROTECTION=true \  
  -e TESTGUIDE_CSRF_ACCEPTED_ORIGINS=testguide.example.org \  
  -e TESTGUIDE_SECURE_COOKIES_OVER_HTTP=false \  
  -e TESTGUIDE_MAX_RAM_PERCENTAGE=50 \  
  -e ENABLE_SNI=true \  
  -e "JDK_JAVA_OPTIONS=-XX:+PrintCommandLineFlags -XX:+PrintFlagsFinal" \  
  --
```

```
--hostname testguide \  
tracetronic/testguide-ubi9:1.220.0
```



Some migration steps are uniquely determined by the assigned *hostname*, so it is important that the *hostname* is assigned and does not change from deployment to deployment until a migration is complete.



When allocating the available memory (e.g. `--memory=16g`) for the started container, only 50% of the application test.guide is allocated within the container! Use the environment variable `TESTGUIDE_MAX_RAM_PERCENTAGE` to change the percentage.



Further setting options for the configuration of the container and its resources can be found in the following documentation: <https://docs.docker.com/reference/cli/docker/container/run/>

5.7. Implementation with Podman

The container image can be also used with Podman. Please use the commands from [Implementation with Docker](#) and replace `docker` with `podman`. Only the `run` command has different log related options:

```
podman container run -d \  
  --name=testguide \  
  --publish 1234:1234 \  
  --log-driver=k8s-file --log-opt max-size=1mb \  
  --memory=16g \  
  --volume /home/<name>/podman/TTS-TM:/var/testguide/TTS-TM \  
  --hostname testguide \  
  tracetronic/testguide-ubi9:1.220.0
```

5.8. SSL certificate configuration

When HTTPS is active, it is strongly recommended to create a custom certificate for test.guide. This custom certificate and its private key must be stored in the folder "SSL" in the TTS-TM directory (see [Used directories](#)). The following requirements must be honored:

- The file names of certificate and private key must be identical (excluding the file name extension)
- The certificate must be in PEM-encoded X.509 format (this is the case if the first line of the certificate-file is `-----BEGIN CERTIFICATE-----`)
- The private key must have the extension `.key`
- The private key must be generated with RSA and be in unencrypted, PEM-encoded PKCS8 format (this is the case if the first line of the key-file is `-----BEGIN PRIVATE KEY-----`)
- Make sure that the stored HTTPs certificate contains the correct hostname so that SNI can be enabled in the next step



If more than one certificate-key pair exists in the directory, the first one (sorted alphabetically) is used.

For testing purposes, you can choose to not create a custom certificate. test.guide will then use a self-signed certificate to encrypt the connections. But be aware that this is insecure and clients like browsers will show warnings.

5.9. Enabling Server Name Indication (SNI)

After successfully setting up the SSL certificate with the correct hostnames, the [SNI](#) flag should be enabled by setting the *environment variable* `ENABLE_SNI` to `true`.

If, after doing so, no connection is possible and HTTP status code 400 is returned, please check the certificate and validate that it was issued for the correct hostnames.



When accessing test.guide via the hostname `localhost` (for example, for testing purposes), it is not possible to enable the flag.



If the clients access test.guide via a proxy and HTTPS is used, make sure that the certificates for accessing the proxy are stored correctly.

5.10. Adding self-signed root certificates for third party tool connections

If test.guide accesses tools via HTTPS and an internal/private certificate authority (CA) is used in the organization, the organization's root certificate must be added to the key store of the Java runtime that executes test.guide.

The command line tool *keytool*, which is delivered with every JDK, can be used to extend the Java key store.

Every JDK installation on the host system has its own key store. Hence, the root certificate must specifically added to the key store of the JDK which is used to run test.guide.

Example:

```
~javaJdkPath~\bin\keytool -importcert -file "C:\Users\~~~\Company_root.cer" -alias  
CA_ALIAS -keystore cacerts -storepass changeit
```

Make sure that the correct Java key store *cacerts* is used by specifying an explicit path to the correct file. The Java key store *cacerts* is extended by the required *Company_root.cer* certificate. This means that test.guide accepts connections to tools that are accessible via the imported certificates - e.g. *Company_root.cer*.

For docker and podman executions

If test.guide is run as a container, the *cacerts* file must be extended on the container host as shown above and then mounted in the test.guide container.

Here is an example of the current test.guide image to mount a *cacerts* file:



```
--volume ~HostPath~/cacerts:/lib/jvm/jre/lib/security/cacerts
```

The **HostPath** is the path where the *cacerts* file is stored on the container host and should be used by the container. After restarting the container with the mounted path, the changes take effect.

Chapter 6. Update procedures

6.1. Updating in Stand-Alone mode

The following steps must be run through to update test.guide:

1. Stop the application by closing the console window (cmd) *startTTSTM.bat*.
2. It is recommended to store the new WAR-file (1.220.0-enterprise.war) at the same location as the old one.
3. If announced in the changelog, merge all changes from the newly provided *startTTSTM.bat* into your existing *startTTSTM.bat*.
4. Update the existing *startTTSTM.bat* in the line that calls the Java interpreter with the latest version: ``%JAVA_EXE% -Xmx1024m -XX:+HeapDumpOnOutOfMemoryError -jar TEST-GUIDE-1.220.0-enterprise.war %HTTP_PORT% %HTTPS_PORT% %TEMP_SERVER_DIR%`` .
5. Re-start the application.



In addition to

- the WAR file → *1.220.0-enterprise.war*,
- the associated license file → *1.220.0-enterprise.LICENSE.html*
- and the changelog files → *changelog.html*, *changelog.eng.html*

should also be updated for reasons of consistency, even if it is not absolutely necessary for the update.

6.2. Updating Linux-Service

1. Put the new WAR file at the same location as the old one.
2. If announced in the changelog, merge all changes from the newly provided *linuxService/installService.sh* into your existing *linuxService/installService.sh*.
3. Update the variable `TG_WAR_NAME` in the existing installation script *linuxService/installService.sh* in line 19 with the new version number.
4. Run the installation script with "sudo -E ./installService.sh"
5. Confirm the prompt about updating the service with <ENTER>. The script will now update the systemd service. Enter missing parameters if necessary.
6. Confirm the prompt about restarting the test.guide service with <ENTER>. This will stop and restart the test.guide service

6.3. Updating Stand-Alone Mode of the Windows-Service

Run through the following steps to update a test.guide instance that has been set up as Windows-Service.



Run all these commands as administrator!

1. The application running as service must be stopped with *windowsService/uninstallService.bat*.
2. It is recommended to store the new WAR file at the same location as the old one.
3. If announced in the changelog, merge all changes from the newly provided *windowsService/installService.bat* into your existing *windowsService/installService.bat*.
4. Update the existing *windowsService/installService.bat* in the line that calls the Java interpreter with the latest version: `SET TG_PATH=%BASE_DIR%..\TEST-GUIDE-1.220.0-enterprise.war`.
5. Run *windowsService/installService.bat* and setup test.guide service.

6.4. Notes on migration path

When ever a new version of test.guide is started it will automatically execute the necessary migration operations to update the database and it's contents. The changelog that is included with each release of test.guide will state if an update requires a certain minimum version as predecessor.



When trying to update from a version that is too low, the update process will fail. Instead updating to the minimum version is needed first.



When required database migrations have not been concluded, the update will fail and a return to the previous version is necessary.

Chapter 7. Link with PostgreSQL database

7.1. Installing PostgreSQL

If PostgreSQL has not been installed yet then please download the installer (version ≥ 13) matching the machine's OS at <https://www.postgresql.org/download/>.

When the installation has finished the GUI tool *pgadmin* can be used to easily administrate the PostgreSQL database.



It is strongly encouraged to change the default Postgres settings.
See section [\[Configuring and Monitoring Databases\]](#)



test.guide handles **each database with 70 connections** in the default case. This means that the corresponding PostgreSQL value *max_connections* must be set in combination with the *shared_buffers*.

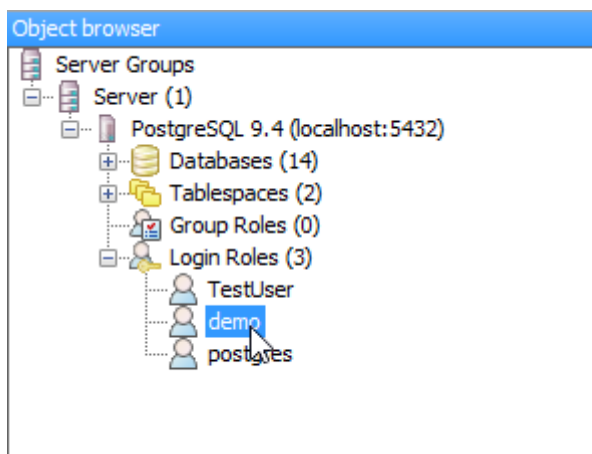
7.2. Security data advice / backup strategy



test.guide requires appropriate and effective database backup and recovery strategies. For details, please refer to [Database data security / backup strategy](#).

7.3. User creation

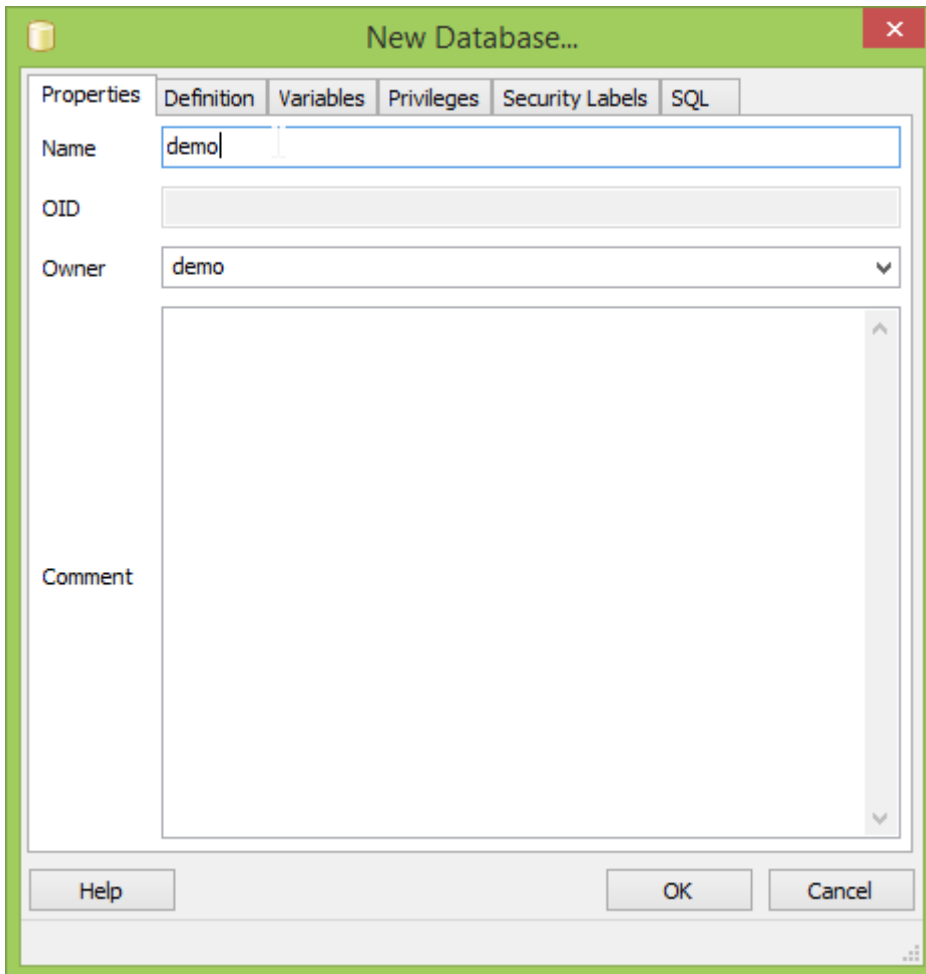
At first the current server must be connected by double-clicking it in the *Object browser*. Four selectable nodes will be displayed below the server name.



Login Roles holds the users of the database. The right-click context menu lists the entry *New Login Role...* to create a new user. In the *Properties* tab, the user name (*Role name*) can be set, the password is set in the next tab *Definition*. Confirm with OK.

7.4. Database creation

The node *Databases* can create a new database via *New Database...* in a similar fashion. Choose a name for the new database and select the previously created user as owner.



Hit OK to confirm.

You have to configure three databases. This is necessary to take advantage of all test.guide features.

- Report database: Main database, stores data related to system settings, report management and test executions
- Artifact database: Stores data related to artifact management
- Monitoring database: Stores data related to test infrastructure

7.5. Configure the database in test.guide

When the databases have been created, they can now be used via JDBC. The setup wizard will guide you through the necessary steps. If needed all databases can also be reconfigured afterwards. To do this use the corresponding settings pages.

In the setup wizard or the setting pages fill the input fields with the previously set parameters (i.e. user name, password). The database URL follows the JDBC scheme consisting of IP-address (or host name), port and database name. If PostgreSQL and test.guide are installed on the same machine,

the *localhost* loopback can be used as host name. In any other case the actual IP address or host name of the database server must be entered.

Reporting database configuration

Database driver
POSTGRESQL

Database URL
jdbc:postgresql://localhost:5432/demo
URL template : jdbc:postgresql://localhost:5432/ttATXDatabase

User
demo

Password
••••

Log level
WARNING

☐ Enable performance profiler (NOTE: Enable only for maintenance!)

Submit

Reset

JDBC address examples:

- same machine: *jdbc:postgresql://localhost:5432/demo*
- remote machine via IP address: *jdbc:postgresql://192.168.178.123:5432/demo*
- remote machine via host name: *jdbc:postgresql://dbserver:5432/demo*
- remote machine with encryption: *jdbc:postgresql://dbserver:5432/demo?sslmode=require*



Encryption should always be activated for remote connections! If the connection to the database is not trusted, *sslmode=require* is not sufficient and instead *sslmode=verify-ca* or *sslmode=verify-full* should be used against man-in-the-middle (MITM) attacks. More information can be found in the PostgreSQL documentation: <https://www.postgresql.org/docs/current/libpq-ssl.html>

The connection is validated and established as soon as these settings are saved. There will be warnings if any errors occur during the process. Further details can be found in the log files.

7.6. Configure the database via environment variables

Alternatively to set up the databases via GUI, you can also configure the databases via environment variables. This allows easier configuration in cloud native environments like Kubernetes.

For each database type, you must provide the settings for the database driver, URL, user and

password. All four settings are mandatory. The following table provides an overview of the required environment variables:

Environment variable (for each respective database)	Explanation
REPORT_DB_DRIVER ARTIFACT_DB_DRIVER MONITORING_DB_DRIVER	Driver of the database, supported is "POSTGRESQL".
REPORT_DB_URL ARTIFACT_DB_URL MONITORING_DB_URL	JDBC URL as described in the section above. For example <i>jdbc:postgresql://dbserver:5432/demo</i>
REPORT_DB_USER ARTIFACT_DB_USER MONITORING_DB_USER	Username of the database as defined above.
REPORT_DB_PASSWORD ARTIFACT_DB_PASSWORD MONITORING_DB_PASSWORD	Password of the database as defined above.

7.7. Configuring and monitoring databases

To ensure optimal performance of the databases, it is essential to configure the PostgreSQL settings based on the specific workload and resource availability. Tuning the settings helps PostgreSQL operate efficiently, improving query performance, reducing latency, and handling larger volumes of data effectively.

You should continuously monitor the databases to ensure the best performance.

7.7.1. Tuning

For an initial performance optimization, you can start by using tools like [pgTune](#). pgTune provides an easy way to generate an optimized PostgreSQL configuration based on the system's available resources and workload type.



The typical workloads of the test.guide databases are often most similar to OLTP (Online Transaction Processing) workloads.

This can serve as a good starting point, but keep in mind that the best settings depend on to the specific workloads of the users.

The [PostgreSQL documentation](#) describes how to set the parameters.

7.7.2. Monitoring

You should at least monitor the following metrics regularly. Especially after changes to the database or to the underlying hardware:

- **CPU usage:** Avoid over-utilizing CPU.

- **Disk I/O:** Ensure no bottlenecks in read/write operations.
- **Memory usage:** Prevent excessive memory use and swapping.
- **Postgres statistics:** E.g. use `pg_stat_statements` to identify and optimize slow queries.

For more information, see [PostgreSQL Monitoring](#).

Chapter 8. Antivirus Plugin

Support for attaching any antivirus scan engine to test.guide through plugins.

Every file uploaded to test.guide will be passed to the linked antivirus tool, when a suitable plugin is found.

Plugins are placed in the TTS-TM sub directory *AV-Plugins* and automatically detected on startup.

The required API to implement a custom plugin is provided on request.

Chapter 9. Monitoring with Prometheus

test.guide provides internal metrics so that third-party tools such as Prometheus can monitor the current state.

There are three metric endpoints currently available: `/api/metrics` (operational metrics), `/api/metrics/performance` (performance metrics) and `/api/metrics/database` (database metrics). An HTTP GET request to the respective end point returns the current values for all available metrics in the Prometheus exposition format.

Example: <http://localhost:8085/api/metrics>

Chapter 10. test.guide configuration

10.1. Application settings

Some settings for the application can be specified to adjust e.g. for available resources or external demands. To configure any of the following parameters, place a configuration file called `applicationsettings.yml` in the [TTS-TM workspace directory](#). In this file, add an entry for each parameter you want to specify.



Please take care to use valid YAML syntax, as this file is a YAML file. You can refer to the examples for each parameter when in doubt.



For most settings, changes to this configuration file are immediately applied by test.guide. If a restart is needed to change the configuration, the documentation states it explicitly. If there is no explicit direction with in the documentation of a setting, then no restart is required.

10.1.1. Limit the number of rows in Excel exports

test.guide supports the export of report filter or coverage filter data to an Excel file. If used carelessly, this can quickly result in a very large number of Excel rows, depending on the filter. Since such documents are difficult to use in Excel (the maximum possible number of rows is around one million) and also require a lot of computing time, memory and temporary storage when created, it is possible to configure a limit for the number of rows that test.guide will allow when exporting.

To do this, add an entry called `excelRowLimit` to the [application settings file](#) and specify the desired limit. For example:

```
excelRowLimit: 50000
```

10.1.2. Configure the retention period for recycle bin items

By default, attributes and constants that have been moved to the recycle bin will be deleted permanently after 90 days. To change the retention period, add an entry called `recycleBinRetentionPeriod` to the [application settings file](#) and specify the desired limit in days. For example:

```
recycleBinRetentionPeriod: 90
```

10.1.3. Configure the session idle timeout



To change this setting, a restart of test.guide is required!

By default, users will be logged out automatically after 60 minutes of inactivity. To change the session idle timeout, add an entry called `sessionIdleTimeout` to the [application settings file](#) and specify the desired limit in minutes. For example:

```
sessionIdleTimeout: 60
```

10.1.4. Configure the time span for report removal

Each project in test.guide can define report remove rules to remove older reports. These rules are executed every night, by default for 4 hours. If this time span is insufficient to remove all out-dated reports, a longer one can be configured. To do so, add an entry called `nightlyReportRemovalTimespan` to the [application settings file](#) and specify the desired time span in hours. The maximum value is 23 hours, a time span shorter than 4 hours will default to 4 hours. For example:

```
nightlyReportRemovalTimespan: 16
```



While the report remove rules are running, the maximum number of parallel uploads is reduced by 1 (see section [Parallel maximum uploads](#)). Increasing the value of this setting might therefore reduce the upload capacity during the configured time span.

10.1.5. Configure a OpenID Connect group mandatory for login

It is possible to define a property that must be included in the response provided by the OpenID Connect Provider to a query of the UserInfo endpoint so that a user can log in. This can be used, for example, to implement a role that is required for a user to log in successfully.

```
restrictLoginViaUserInfoClaim:  
  claimJsonPath: $.groups.*  
  expectedClaimValue: SPECIFIC_REQUIRED_GROUP_NAME
```

The `claimJsonPath` property is used to specify a JsonPath expression that references the location at which the existence of the value specified using `expectedClaimValue` is checked in the response JSON issued by the the UserInfos endpoint. The JsonPath can either reference a JSON array (in this case, the specified `expectedClaimValue` must be an element of the array for the users to be permitted) or directly to an individual value (the value must be equal to the specified `expectedClaimValue` for the users login to be permitted).

10.1.6. Disable the user name and password authentication

When using OpenID Connect, you can completely disable the user name and password based login.

```
disableUserNameAndPasswordLogin: true
```



Be aware that by doing so, you disable the login with the `ServerAdmin` account. Make sure that you either give one of the OpenID Connect users the `ServerManagement` permission (this should be a dedicated account that is not used for other tasks) or be prepared to temporarily enable the user name and password authentication when system configuration or maintenance is necessary.

10.1.7. Configure TRF Web Viewer restrictions



To change this setting, a restart of test.guide is required!

The TRF Web Viewer converts TRF files into HTML reports. By default, conversions are limited to 5 simultaneous conversions and only up to a TRF file size of 100 MiB.

```
trfMaxParallelConversions: 5  
trfMaxFileSize: 100 # MiB
```

10.1.8. Limit the number of report filter results

If used carelessly (especially when a large filter period is selected, e.g., since the start of the project), the *Report filter* found in *Test report management* section may have very high requirements regarding memory. In extreme cases, this can result in out-of-memory situations. To avoid such situations, there is a limit (default: ten million) on the maximum number of results that the *Report filter* can return. This limit can be adjusted to the individual conditions of the test.guide instance by adding an entry called `reportFilterResultLimit` to the [application settings file](#) and specifying the desired value. For example:

```
reportFilterResultLimit: 10000000
```

10.1.9. Configure SQL query timeouts for the report management

Querying the report filter can take a long time, depending on the operating conditions and the filter query itself. test.guide limits the time an individual SQL query is allowed to take. Please note that querying the report filter might involve multiple SQL queries. This timeout affects the report filter, the group filter and the coverage filter. The default timeout is 30 minutes.

```
reportFilterSqlQueryTimeout: 1800 # seconds
```

10.2. Report upload settings

Some settings pertaining to the upload of test reports are configured in a separate file. To configure any of the following settings, a configuration file must be placed in the [TTS-TM workspace directory](#) called `atxheuristicsconfig.properties`.



Different from the application settings file, this file is not a YAML file and uses a different syntax. Please insert the values as shown in the examples.

10.2.1. Parallel maximum uploads

Because test.guide is a multi user tool it can also handle multiple incoming reports at the same time.

The number of parallel uploads and the ATX report size to be processed is mainly limited by the available memory and CPU cores (more on [hardware specifications](#)).



In this context, ATX report size refers to the size of the ATX data (e.g. run test suite with test cases, meta data like constants, attributes and test steps - look at `ecu.test.report.xml`). It does not refer to the total size of the report upload ZIP file.

The available memory and CPU cores in conjunction predetermine the number of parallel uploads test.guide can handle at a maximum ATX report size.

If the uploaded `report.xml` file is too large to be processed using the set configuration, the upload will be rejected to *avoid an OutOfMemoryError exception issue by the server*.



If there are more uploads than upload threads available, then these uploads are queued and processed based on the FIFO method.

The default behavior will handle 4 parallel uploads and determine the maximum ATX report size from the available *JVM heap memory*. When starting test.guide with the unmodified [batch starter](#) it will consume 4GiB of memory. The result is shown in table:

Available JVM heap memory	Upload threads	Maximum report.xml file size
8192MiB	8	70MiB
8192MiB	4	140MiB
4096MiB	4	70MiB
4096MiB	2	140MiB
2048MiB	4	35MiB
2048MiB	2	70MiB

This calculation is an estimate that may still fail in certain scenarios. In order to treat such situations, both parameters can be set explicitly. To set the parameters, please add an entry for each to the [properties file](#), for example:

```
# Size in mebibytes
MAX_ATX_XML_FILE_SIZE = 50
# Number of upload threads
MAX_PARALLEL_UPLOADS = 8
```

When specifying only one parameter the other will be calculated.

10.2.2. Maximum upload file size

Additionally to configuring the maximum size of the ATX report file you can also limit the total size of the uploaded ZIP file. The default limit for an upload is 42 GiB. You can change the limit by adding the configuration parameter `MAX_UPLOAD_SIZE` to the [properties file](#):

```
# Maximum size of complete uploaded file in MiB
MAX_UPLOAD_SIZE = 43008
```

The file size is specified in mebibyte (2^{20} byte). This limit applies to:

- report files uploaded via GUI on the report upload page (`/upload`)
- report files uploaded via REST-API `api/report/reports`
- files added to a test case execution on the test case detail page
- reviews (the limit applies to the complete review including its attachments and the text entered into all review fields)

This setting is completely independent from the two parameters discussed in the previous section.

Chapter 11. Troubleshooting

11.1. Database issues

If you are experiencing frequent loss of the database connection from test.guide (e.g you find `PSQLException: This connection has been closed` in the logs), there are two common reasons for this:

- The database server is closing idle connections.
- The maximum number of concurrent connections of the database server is too low. By default, test.guide uses 70 concurrent connections per database. Please note that a typical test.guide deployment uses three databases, i.e. one for report management, one for test infrastructure monitoring and one for artifact management. So in order to have some margin, the database server should allow 300 concurrent connections if you host all three databases on the same server.

The following database server settings for the `postgresql.conf` are recommended:

```
# Don't close idle connections
idle_session_timeout=0

# Allow 100 concurrent connections per database that is hosted on this database server
max_connections=300
```

Please refer to the official [PostgreSQL documentation](#) for more details on database server configuration.

11.2. Upgrades

If you are having problems (new license key is required, no data available, ...) after an upgrade, check the following points:

- Are the parameters `TG_WORKSPACE` and `TG_LIC_USER` in the installation or start scripts set correctly, i.e. are they identical to installation?
- Will test.guide be started by the same user with the same rights after the update?

If you can answer both questions with yes, contact our support@tracetronic.com and we will look at the problem together.

11.3. Error message "Could not acquire change log lock" during startup

When test.guide fails to start with the error message "Could not acquire change log lock", proceed as follows:



Before performing SQL level operations, ensure there is a recent database backup.

- Shut down the test.guide instance that is showing the error message
- Connect to the database
- Verify that there is a lock using the following SQL command: `SELECT * FROM DATABASECHANGELOGLOCK`
- When in a cluster scenario (using multiple application instances accessing the same underlying database): Check the status of the test.guide instance holding the lock. If it is still alive and migrating data, wait until this process is finished. If it has crashed, shut down also this instance.
- Remove the lock with the following SQL command: `DELETE FROM DATABASECHANGELOGLOCK`
- Restart all test.guide instances

11.4. Error message "Too many open files" on Linux operating systems

It is normal that test.guide opens several thousand files during operation. Some Linux distributions impose an upper limit on the number of open files. If you are experiencing this error message, check whether this limit is too low. The recommended value is at least 4096 files. Usually this value is set via the item `nofile` in the configuration file `/etc/security/limits.conf`. Refer to the documentation of your Linux distribution.

11.5. Failure on startup related to MonitoringDBManager or ArtifactDatabaseManager

If test.guide fails to start and the error message displayed begins with "MonitoringDBManager" or "ArtifactDatabaseManager", there is most likely a problem connecting to one of the configured module databases, e.g. the database might have been moved to another host.

To check and correct the configuration of your monitoring and artifact databases, you can open the URL `/fixModuleDbConfig` (e.g. <https://localhost:8443/fixModuleDbConfig>). If one of the databases is unreachable, its configuration is displayed and can be modified.



To access the page you have to insert the secret from the `secret.txt` in your workspace.

After changing the configuration, please stop the application and start it again for the changes to take effect.



If the aforementioned URL cannot be opened (i.e. you get a 404 error or similar), then the failed startup is caused by some other problem. Please take a look at the logs or contact our support.

11.6. Failure on startup "Maintenance migrations pending"

If test.guide fails to start and the error message contains the message "CustomChangeException: There are maintenance migrations pending" then some of the required maintenance migrations for the new version have not been concluded yet.

Please stop test.guide, start the previous version again and wait until all maintenance migrations have been completed.



All migrations are listed under *System configuration* → *System status* in the section *Database migration state*.