AUTOMATIONEDGE SERVER

INSTALLATION GUIDE

Release 8.1.1

https://automationedge.com

General: info@automationedge.com
Support: support@automationedge.com
Training: training.support@automationedge.com

India: +91-20-6632-1000
U.S.A. : +1-281-220-1057

Contents

About the Installation guide

The guide provides details about the installation process of the AutomationEdge Server (AE Server).

The chapter includes the following topics:

Intended audience

The guide is useful for users who want to install the AE Server.

Document conventions

Conventions	Meaning	Example
Bold	The **bold** typeface is used to present references to menu options, fields, numbered captions, sections, and button names.	In the **Edit System Variable** dialog, enter the details.
Italic	The italic typeface is used to present keyboard or user entries.	Enter a new name for the system variable. For example, *Path*.
Bulleted list	A bulleted list is used for an unordered series of concepts, items, or options.	This chapter includes the following topics: • Intended audience • Document conventions
Numbered list	A numbered list indicates the sequence of processes, events, or steps.	1. Under **System Variables**, select *Path* as the system variable, and click **Edit**. 2. Scroll to the beginning of the variable value and enter *%JRE_HOME%bin*.
	The symbol indicates additional information.	 You can install the database and Apache ActiveMQ on different machines, if required.

Conventions	Meaning	Example
	The symbol indicates very important information.	 If you decide to enable the **Workflow Monitoring** feature and deploy the **Workflow Monitoring** microservice, then Nginx will be the entry point for AutomationEdge.
	The symbol indicates AutomationEdge's recommendation for users that are helpful in installation or application usage.	 Install Nginx and Metrics Service on separate machine than the one for Tomcat.

Prerequisites

In the chapter, you will learn about the prerequisites required for installing the AE Server.

System prerequisites

In this topic, you will learn about the minimum system configurations required to install AE.

Supported operating systems

Following is the list of requirements to deploy AutomationEdge (AE) server:

Operating system	Windows Server 2019 and 2022	Windows 10 and 11 (64 bit)	Linux (64 bit)
Random Access Memory (RAM)	8 GB		
Hard Disk	200 GB		
No. of virtual CPUs	4		

Following is the list of requirements for AE Agent deployment:

Operating system	Windows Sever 2019 and 2022	Windows 10 and 11 (64 bit)	Linux (64 bit)
Random Access Memory (RAM)	4 GB		
Hard Disk	200 GB		
No. of virtual CPUs	2		

Required software

Following is the list of software required for AE Server release 8.1.1:

Tool	Version	URL
Java Runtime Environment (JRE 64 bit)	Java 21	We recommend downloading the latest version from Eclipse Temurin. https://adoptium.net/en-GB/temurin/releases/?version=21
Database	PostgreSQL 14.x-16.x	https://www.enterprisedb.com/downloads/postgres-postgresql-downloads
	Oracle 19c	
	Microsoft SQL Server 2019 and 2022	
Apache Tomcat for 64 bit	Latest Tomcat 11	https://tomcat.apache.org/download-11.cgi At the time of 8.1.1 release the latest version is 11.
Apache ActiveMQ	Latest ActiveMQ Classic 6.1.x	https://activemq.apache.org/components/classic/download/classic-06-01-03 At the time of 8.1.1 release the latest version is 6.1.4.

In the *Production Environment*, we recommend you install the *database* and *ActiveMQ* on different machines, that is, machines other than where *Tomcat* is installed.

Supported browsers

The following browsers are supported:

Release version	Chrome	Microsoft Edge	Firefox
8.1.1	120 and above	120 and above	120 and above

Pre-installation activities

In the chapter, you will learn about activities you need to perform before proceeding with the AE Server installation, and includes the following topic, Setup AE home directory structure.

Setup AE home directory structure

Create an AE Home directory structure before you start with the AE Server installation.

AE directory structure on Windows or Linux

Create various directories on Windows or Linux, as required.

To create AE directory structure on Windows or Linux:

1. Create the top-level directory of AE installation, which is also known as AE base directory on your system.

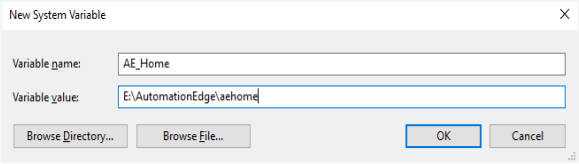
- All AE software and components are installed in the AE base directory.
- The complete file path of AE Home is referred to as **<AE Home>**. AE Home is the working directory for AE Server.

Windows	Linux
1. Top-level directory, for example, **AutomationEdge Base**: *<preferred_drive>:\AutomationEdge*. 2. In the AE base directory, create the *tools* and *aehome* directories as follows: • *<preferred_drive>:\AutomationEdge\tools* • *<preferred_drive>:\AutomationEdge\aehome*	1. Top-level directory, for example, **AutomationEdge Base:** */home/<user>/AutomationEdge*. 2. In the AE base directory, create the *tools* and *aehome* directories as follows: • */home/<user>/AutomationEdge/tools* • */home/<user>/AutomationEdge/aehome*

Setup AE home environment variable

After creating the AE Home directory, set its environment variable.

To create environment variable on Windows or Linux:

Windows	Linux
1. Access **Control Panel → System and Security → System → Advanced system settings → System Properties** dialog. 2. In the dialog, click the **Advanced** tab, and then click **Environment Variables…**. 3. In the **Environment Variables → System variables** section, click **New**. 4. In the **New System Variable** dialog, enter the following variable values:  *Figure 1: New System Variable dialog* • **Variable name**: Enter value as, *AE_Home*. • **Variable value**: Enter value as, *<preferred drive?\AutomationEdge\aehome*. 5. Click **OK**.	To create environment variable for Linux, enter the following details using the command prompt: *AE_HOME=<path for the AE working directory>* in */etc/environment file or in /home/<user>/.bash_profile/.bash_rc/.profile*.

Install AE Server

In the chapter, you will learn how to install Java, database, ActiveMQ, and Apache Tomcat.

The chapter includes the following topics:

- Install Java
- Install database
- Install Apache ActiveMQ
- Install Apache Tomcat

For Linux installation, the path is */home/<user>/AutomationEdge*.

Install Java

You need to install Java to work with AE Server.

Setup Eclipse Temurin Java 21

You need to setup Eclipse Temurin Java 21.

To setup Eclipse Temurin Java 21:

1. Access the binaries for the JDK at https://adoptium.net/en-GB/temurin/releases/?version=21.
2. Identify the JRE binaries for the operating system, such as Windows and Linux with desired architecture (x64 or x86) and download.

On the site page, select as follows:

- **Operating System:** *Windows* or *Linux*
- **Architecture:** *x64* or *x86*
- **Package Type:** *JRE*
- **Version:** *21-TLS*

Click on the selection to download the JRE zip file.

3. On your system, unzip the JRE binaries into the **<preferred drive>\AutomationEdge\tools** folder.

Setup JRE_HOME

Set the **JRE_HOME** environment variable to point to the JRE installation directory.

To set environment variable:

1. Access the **System Properties** dialog, and then click **Environment Variables…**. The **Environment Variables** dialog appears.
2. In the **System variables** section, click **New…**, the **New System Variable** dialog appears. Enter the following details in the dialog:
   - **Variable name**: Enter as, *JRE_HOME*.
   - **Variable value**: Enter path of the JRE installation directory. For example, **<preferred drive>\AutomationEdge\tools**.

- From AE release 8.1.1 onwards, point **JRE_HOME** to the JRE21 path.
- If you install AE using the installer you must ensure that the **AE_JRE_HOME** environment variable points to the JRE installation directory.

Edit Path environment variable

After creating the **JRE_HOME** variable, edit the **Path** environment variable.

To edit Path environment variable:

1. In the **Environment Variables** dialog, select **Path** in the **System variables** section and click **Edit**.
2. In the **Edit environment variable** dialog, enter *%JRE_HOME%\bin* at the start of the path.

If you installed AE using installer your path variable must point to *%AE_JRE_HOME%/bin*.

Verify installed Java

Verify the installed Java, using the following command:

```
java –version
```

For **Eclipse Temurin Java 21**, the following details appear:

```
openjdk version "21.0.4" 2024-07-16 LTS
OpenJDK Runtime Environment Temurin-21.0.4+7 (build 21.0.4+7-LTS)
OpenJDK 64-Bit Server VM Temurin-21.0.4+7 (build 21.0.4+7-LTS, mixed mode, sharing)
```

Install database

Install any or all the following databases, as required:

- PostgreSQL
- Oracle
- Microsoft SQL Server

Install PostgreSQL database

Install PostgreSQL database, which you have downloaded from https://www.enterprisedb.com/downloads/postgres-postgresql-downloads.

For PostgreSQL installation and administration details, see https://www.postgresql.org/docs/.

Best practice: Create a dedicated database user with appropriate privileges for accessing the AE database, rather than using default administrative accounts, such as the postgres or admin.

A dedicated database user provides security, isolation, and limited scope to the database.

To create a database and database user:

1. After installing PostgreSQL, create a blank database, *vae*.
2. Create a user and role with granular access permissions, *read-write*.

Use the master user created when installing PostgreSQL for creating other users, roles, and databases only.

3. After creating the AE user, update **ae.properties** file with the user credentials.

Approach for setting up fine-grained access control in PostgreSQL

Following is the recommended approach for setting up fine-grained access control in PostgreSQL:

- Use the master user to create roles per application, such as read-write and read-only roles.
- Add permissions to allow these roles to access various database objects. For example, the read-only role can only run SELECT queries.
- Grant the roles the least possible permissions required for the functionality.
- Create new users for application or specific functionality. For example, *ae_app_user* and *reporting_user*.

- Assign the applicable roles to these users to quickly grant them the same permissions as the role. For example, grant the read-write role to *ae_app_user* and grant the read-only role to *reporting_user*.
- Revoke user permissions by removing the user's role.

Install Oracle

Install Oracle database, if required.

To install Oracle database:

1. Install Oracle database.

   For Oracle installation and administration details, see https://docs.oracle.com/en/database/oracle/oracle-database/index.html.

2. Create a user, *vae* and grant appropriate permission. For example, *read-write*.

Install Microsoft SQL Server

Install Microsoft SQL Server, if required.

We recommend using SQL Server 2019+.

To install Microsoft SQL Server:

1. Download Microsoft SQL Server and apply the required service packs to install the MSSQL Server.
2. Create a database, *vae* on MSSQL Server.

The default database collation on SQL Server is *SQL_Latin1_General_CP1_CI_AS*. If you need Unicode support, you will need to create the database with *Latin1_General_100_CI_AS_SC_UTF8* collation.

Install Apache ActiveMQ

1. Install or upgrade Apache ActiveMQ.

- If you have the **Apache ActiveMQ 5.18.x** version on your system, then upgrade to the latest **ActiveMQ Classic 6.1.x** version. For release 8.1.1, the latest version of **ActiveMQ Classic** is *6.1.4*.

For Windows: Download *apache-activemq-6.1.4-bin.zip*

For Linux: Download *apache-activemq-6.1.4-bin.tar.gz*

- For **Apache ActiveMQ** installation and administration details, see https://activemq.apache.org/components/classic/documentation/getting-started.

2. Configure username and password for ActiveMQ.

We highly recommend enabling JMX on Apache ActiveMQ with password protection. If you enable JMX, then you can monitor the health of the ActiveMQ.

To enable JMX follow the instructions at https://activemq.apache.org/components/classic/documentation/jmx.

Install Apache Tomcat

Install Apache Tomcat and configure the details.

To install Apache Tomcat:

1. Download Apache Tomcat for 64-bit from https://tomcat.apache.org/download-11.cgi.
2. Install Apache Tomcat and configure the details, as required.

For Apache Tomcat installation and administration details, see https://tomcat.apache.org/tomcat-11.0-doc/index.html.

After Apache Tomcat installation and configurations, complete the following configurations:

Delete Tomcat OOTB applications

Before deploying AE apps, user must delete the out-of-the-box apps, such as ROOT, manager, docs, and so on that Tomcat installation packages provide.

To delete the apps:

1. Go to **<Tomcat home>/webapps**.
2. Delete following directories: docs, examples, host-manager, manager, and ROOT.

Make aeui the default application

Configure Tomcat's default settings to make aeui the default application.

To configure the default application settings:

1. Add docBase detail in the **<Tomcat home>/conf/server.xml** file. In the **<Host></Host>** tags, enter:

```
<Context path="" docBase="aeui"/>
```

2. (Only) for SSL setup, edit **<TOMCAT_HOME>/conf/web.xml** file to redirect to the aeui application. Before the **</web-app>** tag, enter:

```
<!-- Require HTTPS for everything except /img (favicon) and css -->
<security-constraint>
<web-resource-collection>
<web-resource-name>HTTPSOnly</web-resource-name>
<url-pattern>/*</url-pattern>
</web-resource-collection>
<user-data-constraint>
<transport-guarantee>CONFIDENTIAL</transport-guarantee>
</user-data-constraint>
</security-constraint>
<security-constraint>
<web-resource-collection>
<web-resource-name>HTTPSOrHTTP</web-resource-name>
<url-pattern>*.ico</url-pattern>
<url-pattern>/img/*</url-pattern>
<url-pattern>/css/*</url-pattern>
</web-resource-collection>
<user-data-constraint>
<transport-guarantee>NONE</transport-guarantee>
</user-data-constraint>
</security-constraint>
```

3. After all the steps are complete, start or restart Tomcat server.

To check if the application is running on SSL and is redirecting the *http* request to *https* automatically, enter the URL as *https://<application URL>*.

Hide server name and version

Hide the Tomcat server name and version. Complete the process of hiding the details and then restart the Tomcat server.

To hide Tomcat server name and version:

1. Go to **<Tomcat home>/lib** and create the directory **org/apache/catalina/util** under **lib**.

2. In the new directory, create a **ServerInfo.properties** file, and add an entry for **server.info** property in the file.

   Keep the value for this property blank. For example, **server.info**=.

3. Edit **server.xml** configuration file available under **<Tomcat home>/conf**, and add the following details in the **Connector** node:

   The step is optional.

   server="AutomationEdge"

Figure 2: Connector node view

Tomcat: Access Log Format configurations

Customize the **access log format** with a specific pattern for better tracing and logging details. Modify the **AccessLogValve** configuration in **server.xml**.

```
<Valve className="org.apache.catalina.valves.AccessLogValve"
    directory="logs"
    maxDays="30"
    prefix="localhost_access_log"
    suffix=".txt"
    pattern="%{yyyy-MM-dd'T'HH:mm:ss.SSSXXX}t [%{X-Trace-Id}o] [%I] [%h] - %{u}i &quot;%r&quot; %s %b %F %{ms}T
&quot;%{Referer}i&quot; &quot;%{User-Agent}i&quot;" />
```

where *pattern* consists of the following:

- *%{yyyy-MM-dd'T'HH:mm:ss.SSSXXX}t*: ISO 8601 formatted timestamp, including milliseconds and timezone.
- *%{X-Trace-Id}o*: Custom HTTP header for **Trace ID** (ensure your application sends this header).
- *%I*: Thread ID.

- *%h*: Remote host IP address.
- *%{u}i*: Remote user, if any.
- *"%r"*: The request line (method, URI, and HTTP version).
- *%s*: HTTP status code.
- *%b*: Bytes sent, excluding headers.
- *%F*: Time taken to commit the response, in milliseconds.
- *%{ms}T*: Time taken to process the request, in milliseconds.
- *"%{Referer}i"*: The HTTP referrer header.
- *"%{User-Agent}i"*: The HTTP User-Agent header.

The configuration will produce an access log entry. For example,

```
2024-09-24T17:25:47.346+05:30 [66f2a8c302140f88f1e8c64fd917db55] [http-nio-8080-exec-5] [127.0.0.1] - - "GET /aeengine/rest/system/health HTTP/1.1" 200 893 149 152 "-" "Mozilla/5.0 "
```

Tomcat: Logging configuration

To minimize unnecessary console logging in Tomcat, you can configure logging to avoid output to **catalina.out** by removing the **java.util.logging.ConsoleHandler** and using only **org.apache.juli.AsyncFileHandler**. The configuration will help reduce redundant logging in the console while maintaining file-based logging.

Configure Tomcat logging

To configure the Tomcat logging configure the **logging.properties** file and ensure that file-based logging is configured properly.

To edit the logging.properties file:

1. Go to the Tomcat installation's **conf/** folder and open the **logging.properties** file, that is, **$TOMCAT_HOME/conf/logging.properties**.
2. **Modify the Root Logger Handlers**: By default, Tomcat is configured to log on to both, the console and a file. To prevent logging to the console, **catalina.out**, replace:

   ```
   .handlers = 1catalina.org.apache.juli.AsyncFileHandler, java.util.logging.ConsoleHandler
   ```

   with

   ```
   .handlers = 1catalina.org.apache.juli.AsyncFileHandler
   ```

   The change will ensure that logging is handled only by **AsyncFileHandler** and not by **ConsoleHandler**, effectively preventing unnecessary console output.

Configure file-based logging:

Make sure that **org.apache.juli.AsyncFileHandler** is properly configured to log messages to a file. Verify or set the following properties in the same **logging.properties** file:

```
1catalina.org.apache.juli.AsyncFileHandler.level = FINE

1catalina.org.apache.juli.AsyncFileHandler.directory = ${catalina.base}/logs

1catalina.org.apache.juli.AsyncFileHandler.maxDays = 90
```

The settings will direct the logging to a file under the **${catalina.base}/logs** directory with a retention period of 90 days.

Tomcat: Configure CATALINA OPTS

First create the **setenv.bat** and **setenv.sh** files for Windows and Linux, respectively. Configure CATALINA_OPTS to point to **ae.properties** and relevant log location, that is, **$CATALINA_HOME/logs**.

Windows	Linux
Under **${TOMCAT_HOME}\bin** folder: **setenv.bat** `set CATALINA_OPTS=-Dspring.config.additional-location="%AE_HOME%/conf/ae.properties" -DLOG_DIR="%CATALINA_HOME%/logs"`	Under **${TOMCAT_HOME}/bin** folder: **setenv.sh** `CATALINA_OPTS="-Dspring.config.additional-location=\"$AE_HOME/conf/ae.properties\" -DLOG_DIR=\"$CATALINA_HOME/logs\""`

Tomcat: Memory Settings

You can set Tomcat memory settings in the following two ways:

- Memory settings for Tomcat as service

OR

- Memory settings for Tomcat using command line

The memory setting for Tomcat is mandatory.

Memory settings for Tomcat as service

When running Tomcat as a service, modify the memory settings, as required.

To configure Tomcat as service details:

1. Stop Tomcat service, if it is up.
2. Open the command prompt, go to **%TOMCAT_HOME%/bin** execute **service.bat install**. The Tomcat service is installed.
3. Go to **%TOMCAT_HOME%/bin**, and select the Tomcat executable, and run it. For example, **tomcat10w.exe**. Running the executable will allow you to make changes to the Tomcat properties, if required.
4. In the **Java** tab, make the following changes:

   Java Options section, enter:

```
-Djava.util.logging.config.file=<yourPath>\conf\logging.properties

-Dspring.config.additional-location=<yourPath>\AutomationEdge\aehome\conf\ae.properties

-DLOG_DIR=<yourPath>\AutomationEdge\tools\apache-tomcat-10.1.30-windows-x64\apache-tomcat-10.1.30\logs

-Dmanagement.health.tomcat.enabled=false

-Dmanagement.health.file-descriptor.enabled=false
```

   Change the following properties:

   - **Initial memory pool**: *1024 MB**
   - **Maximum memory pool**: *2048 MB**

You can increase the allocated memory to higher values depending upon the load on the AE server. For example, if a machine has 8 GB RAM set, the Initial memory pool: *2048 MB** and Maximum memory pool: *4096 MB**.

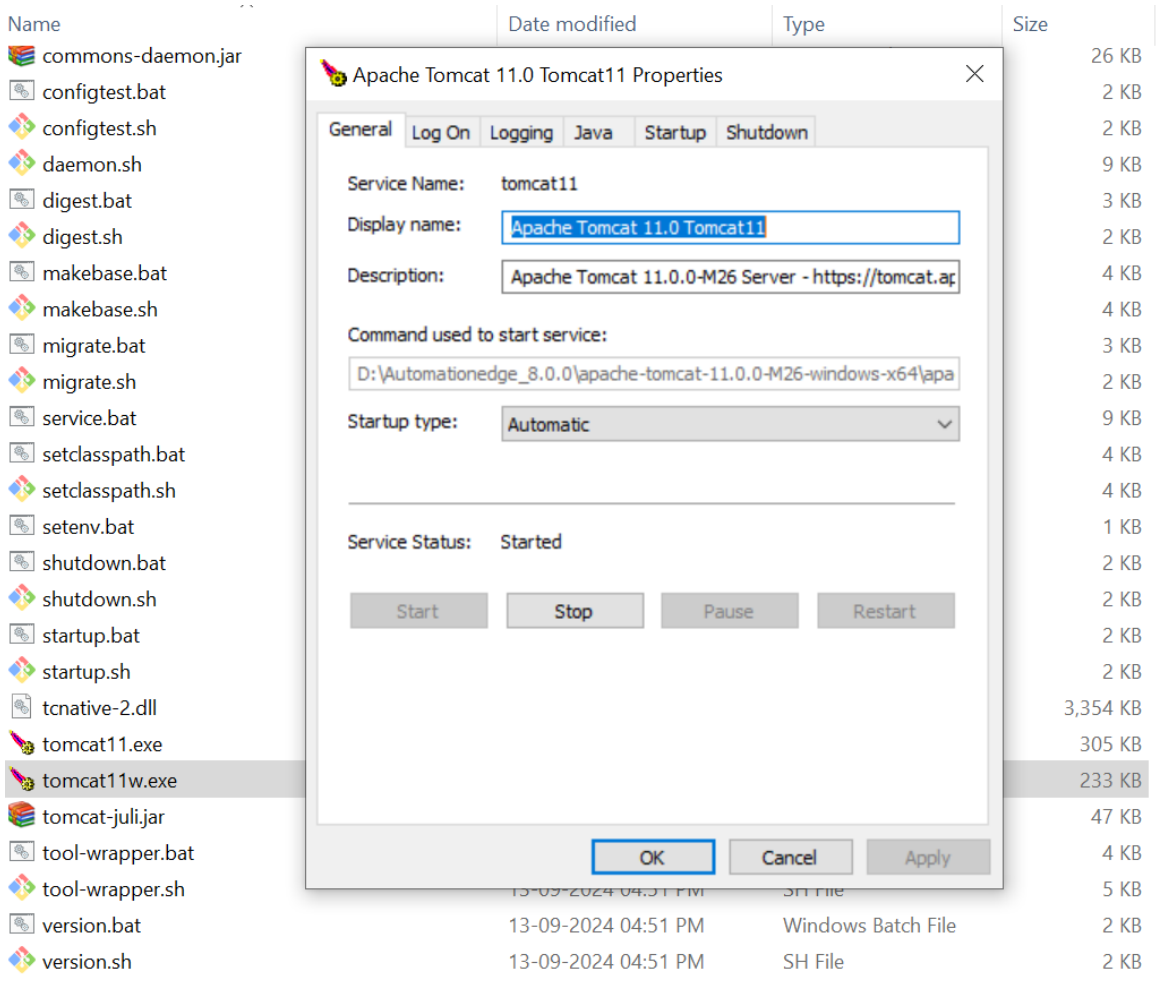
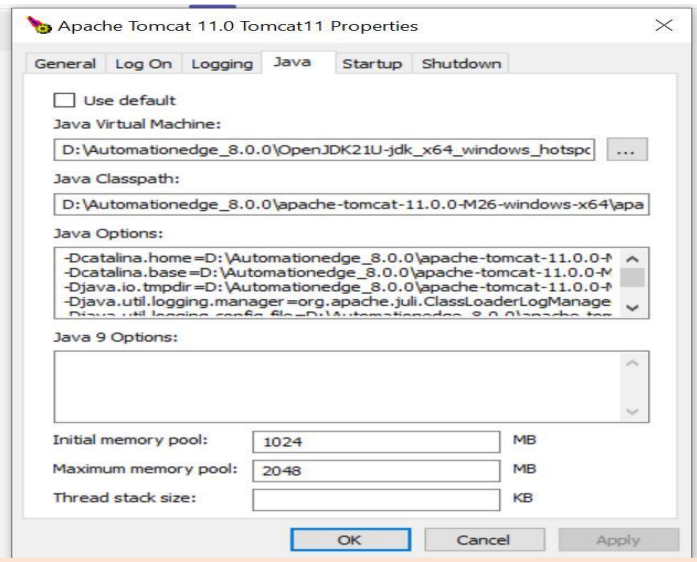


Figure 3: Apache Tomcat 11 Properties dialog

Remove all details from the **Java 9 Options** section.

5. Start or restart the Tomcat Service.

- For details about starting the Tomcat Security Manager, see https://tomcat.apache.org/tomcat-11.0-doc/security-howto.html. You can also see Enable Apache Tomcat Security Manager.
- Ensure you use the absolute path to the policy file. For example, see the following screenshot:



Memory settings for Tomcat using command line

When running Tomcat using command line, modify the memory settings, as required.

To configure Tomcat memory settings:

1. Stop Tomcat service, if it is up.
2. Edit **setenv.bat** or **setenv.sh** for Windows and Linux respectively in **%TOMCAT_HOME%/bin**.
3. Add the following detail to the **setenv** file:
   - For Windows set *JAVA_OPTS=-Xms1024m -Xmx2048m*.
   - For Linux export *JAVA_OPTS="-Xms1024m -Xmx2048m"*.

   You can increase the allocated memory to higher values depending upon the load on the AE server.
4. Start or restart Tomcat.

Enable CORS with AE 8.0

Add the following properties to AE_HOME/conf/ae.properties file:

```
# Allow OPTIONS method for CORS
spring.security.web.firewall.allowed-http-methods=GET,POST,PUT,DELETE,OPTIONS


# Enable CORS
endpoints.web.cors.enabled=true
endpoints.web.cors.allowed-origins=*
```

```
endpoints.web.cors.allowed-methods=GET,POST,PUT,DELETE

endpoints.web.cors.allowed-headers=Content-Type,X-Requested-With,accept,Origin,Access-Control-Request-Method,Access-Control-Request-Headers,X-session-token
```

Restart Tomcat once this change is done

Verify and start services

Change the **Startup Type** to *Automatic* for the following Windows services and start them in the following order:

1. Database installed for AE, such as PostgreSQL
2. ActiveMQ
3. ApacheTomcat, the version installed for AE

Configure AutomationEdge environment

In the chapter, you will learn about configuring the AutomationEdge environment.

The chapter includes the following topics:

- Configure application settings
- Deploy .war files
- Configure AutomationEdge UI settings
- Verify AutomationEdge UI sign-in

Configure application settings

After deploying the applications, configure their settings.

AutomationEdge Engine Settings

Create a directory named **/conf** is created under <AE home>. In the conf folder add a file **ae.properties**. Configure the properties with appropriate values.

Configure ae.properties

Set the following properties with appropriate values depending on the installed database.

List of ae.properties

Name	Description	Example
Database:		
database.type	Provide the data type from: • *Postgres* • *SQLServer* • *Oracle*	POSTGRES: **database.type**= POSTGRESQL SQL SERVER: **database.type**= MSSQLSERVER ORACLE SERVER: **database.type**= ORACLE
database.url	Provide the JDBC URL of the database.	POSTGRES:

Name	Description	Example
		database.url=jdbc:postgresql://localhost:5432/vae?profileSQL=false For workflow metrics: **database.url=** *jdbc:postgresql://localhost:5432/wfmetrics?autoReconnect=true&useSSL=false&allowPublicKeyRetrieval=true* SQL SERVER: **database.url**=jdbc:oracle:thin:@localhost:1521:orcl For workflow metrics: **database.url=** *jdbc:sqlserver://10.41.5.157:1433;databaseName=wfmetrics;encrypt=true;trustServerCertificate=true* ORACLE: **database.url=** *jdbc:sqlserver://localhost:1433;databaseName=vae;*
database.user	Provide the login username of the database.	*database.username= postgres*
database.password	Provide the password of the database.  On startup AE encrypts the password if it is in plain text.	database.password=<your database password>
ActiveMQ:		
activemq.broker.url	URL of the ActiveMQ broker.	*tcp://locahost:61616*

Name	Description	Example
mq.username	Login user of the broker.	*admin*
mq.password	Login password of the broker.  On startup AE encrypts the password if it is in plain text.	*admin*
activemq.jmx.endpoints	Configure the connection endpoints to ActiveMQ's JMX. Used to monitor ActiveMQ cluster and queue status.	*localhost:1234*
activemq.jmx.user	Login user of ActiveMQ jmx endpoint.	*monitorrole*
activemq.jmx.password	Login password of ActiveMQ jmx endpoint.  On startup AE encrypts the password if it is in plain text.	
Hazelcast:		
ae.clusters.members	In the case of HA, list of member addresses (IP or hostname csv separated) that form the cluster. If there are 3 nodes in HA cluster the values will be 10.41.4.1, 10.41.4.2, 10.41.4.3	*Localhost*
ae.clusters.port	Port that the Hazelcast instance will bind to for communication with other nodes.	*5900*

Name	Description	Example
	It must be the same for all nodes in the cluster.	
AE-UI:		
ae.portal.baseURL	Base URL of the application's UI portal. This URL is used for generating links for features such as password reset verification, account activation and other user actions that require navigation to the portal. It ensures that the links generated by the backend point to the correct UI location	*tcp://locahost:8080/aeui*
Session Tokens:		
ae.sessiontoken.validityInMinutes	Duration (in minutes) for which a session can remain idle. After this time, the token will expire, and the user will need to reauthenticate.	*15*
ae.sessiontoken.autoRenewalInHours	Duration (in hours) for which a session token remains valid. After this time, the token will expire, and the user will need to reauthenticate. -1 indicates only ae.sessiontoken.validityInMinute will be applied	*-1*
File Upload:		

Name	Description	Example
ae.banned.file.extension	Comma separated list of file extensions to be restricted from upload	*.exe,.msi*
ae.validate.files.in.archive	Always restrict the uploading of archives that contain banned files or files except allowed file types.	*False*
ae.scan.embedded.files	Always scan .docx, .xlsx, .xls, and .doc files before upload to restrict any banned files or embedded file types that are not allowed.	*False*
ae.file.upload.size.limit.in.mb	Input,Output upload. And file managemnet tab	*200*
Workflow Requests:		
ae.pendingreq.lower.threshold	Lower threshold for pending requests For Pending Requests Threshold feature to be available, configure the details under Settings -> Email Notification, and specify the users for sending email notification. If the agent is down, requests remain in New State. If the number of pending requests goes over the lower threshold, then users receive an email notification if an email notification is configured.	*300*
ae.pendingreq.higher.threshold	Higher threshold for pending requests.	*800*

Name	Description	Example
	If number of pending requests goes over the higher threshold, then no more requests can be submitted.	
ae.pending.request.limit.per.tenant	If value is greater than 0, then whenever pending requests (New + Retry state) for a tenant crosses this threshold a mail is sent to Tenant Admins and System Admins. If the request is submitted by scheduler, then a record is created with Failure status and an error message is displayed to the users.	*0*
ae.new.request.cleanup.job.interval.minutes	Frequency (in minutes) at which the cleanup job runs to remove requests that have remained in the "new" state for longer than a defined threshold. Min value can be 15 minutes and Max value can be 60 minutes.	*30*
Archive:		
ae.archive.location	Directory path where the application should store archived files. If specified tenant archive location will be ae.archive.location/{ORG_CODE}. Note: Archive location ends with file separator. In the file path, use the forward (/) slash, and if you	*$AE_HOME/Archives/Tenants/{ORG_CODE}*

Name	Description	Example
	are using the back (\) slash, use two slashes (\\). The archive location stores thes purged data of the four entities - workflow requests, audit logs, notification history and user session history in the form of zip files.	
Quartz:		
ae.scheduler.threadpool.size	Deprecated in 8.0. Use spring.quartz.properties.org.quartz.threadPool.threadCount instead.	*20*
spring.quartz.properties.org.quartz.threadPool.threadCount	Specifies the number of threads in the Quartz scheduler's thread pool. It controls how many concurrent jobs can be executed by Quartz. A higher thread count allows more jobs to be run simultaneously.	*20*
SMTP:		
ae.mail.smtp.connectiontimeout.seconds	Maximum time, in seconds, to wait for a connection to the SMTP server.	*30*
ae.mail.smtp.timeout.seconds	Maximum time, in seconds, to wait for the SMTP server to respond during email transmission. If no response is received within this period, a timeout occurs.	*30*

Name	Description	Example
Schedule:		
ae.schedules.enable.cron.expression.input	If true you can provide cron expression in **Scheduler -> Create New Schedule**	*false*
Workflow Monitoring:		
ae.workflow.monitoring.enabled	Enable or disable monitoring of workflows.	*false*
ae.workflowmetrics.service.base-url	Base URL for the workflow metrics service.	*http://localhost:80/metrics*
System Health Checks		
ae-engine.health.cron.expression	Cron expression that determines how frequently health checks should be performed. By default, this is 1 minute	*0 */1 * ? * ****
management.health.activemq.enabled	Whether to enable activemq health check.	*true*
management.health.activemq.queue-size.threshold	Threshold for monitoring the size of an ActiveMQ queue. If the size of the queue exceeds the defined threshold, the health status of the ActiveMQ component may be marked as unhealthy.	*100*
management.health.db.enabled	Whether to enable disk space health check.	*true*
management.health.diskspace.enabled	Whether to enable database health check.	*True*

Name	Description	Example
management.health.diskspace.path	Path used to compute the available disk space.	
management.health.diskspace.threshold	Minimum disk space should be available. Example: 1024 (bytes), 1024KB, 512MB, 1GB, 1TB	*10MB*
management.health.http.enabled	Whether to enable http health check. Basically, does a ping to itself via HTTP.	*True*
management.health.jms.enabled	Whether to enable jms (activemq) health check.	*true*
management.health.memory.enabled	Whether to enable JVM memory health check.	*true*
management.health.memory.threshold	Threshold to determine when memory usage of the application is considered unhealthy. Value is between 0 and 1 (inclusive) that represents the percentage of used memory. When the memory usage exceeds this threshold, the health indicator will report the application as unhealthy.	*0.85*
management.health.stuck-thread.enabled	Enable or disable the detection of stuck threads in the application	*true*
management.health.thread.enabled	Whether to enable thread health check.	*true*
management.health.thread.threshold	Threshold for the number of active threads in the application.	*500*

Name	Description	Example
	If the number of active threads exceeds this threshold, the health indicator will report the application as unhealthy.	
management.health.thread-deadlock.enabled	Enable or disable the detection of thread deadlocks in the application.	*true*

Click the following link ae.properties and download the sample ae.properties file. Add the properties from the ae.properties table and customize the property details, as required.

Deploy .war files

Deploy .war files for configuring AE environment.

To deploy .war files:

1. Copy the following .war files in **<Tomcat home>/webapps**.
   - aeengine.war
   - aeui.war
2. Start Tomcat server.

Configure AutomationEdge UI settings

Configure the AE UI settings in the **<Tomcat home>/webapps/aeui/aeui-config.properties** file.

By default, after AE installation **aeui-config.properties** file is in plain text format.

To change the UI settings:

1. Access **aeui-config.properties** file in the **<Tomcat home>/webapps/aeui** directory.
2. In the **aeui-config.properties** file configure the following properties:
   i. **AutomationEdge Base URL**

If **aeengine** and **aeui** are deployed on two separate Tomcat servers on two different machines, then change the **baseurl** property with appropriate **IP** or **Host** and **Port** values where AE server (aeengine) is running.

Default value for the **baseurl** property is, **baseurl** = */aeengine/rest*.
For example, **baseurl**=*https://10.2.4.56:8080/aeengine/rest*.

ii. Customer Logo

To view the customer logo in the UI application, add customer logo file under directory **<Tomcat home>/webapps/aeui/assets/images**.

File types supported are, *.jpeg*, *.png*, *.svg*, and *.gif*.

Update the property **tenantLogoFile** with the file name you have added in the previous step.

Default value is, **tenantLogoFile** = *customer-logo.png*.

iii. Login Message

To add custom login message that appears on the login screen, enter the text message in the **loginMessage** property. For example, **loginMessage**=*Welcome to AutomationEdge*.

Default value for the property **loginMessage** is blank.

iv. **Storage**

Use the property to store browser session data.

For security reasons, the default and recommended option is *InMemory*, which means that session data is not stored on disk and is lost on browser refresh. The other available option is *LocalStorage*.

Default value for the **storage** property is **storage** = *InMemory*.

v. Security vault

Use the property to enable the System Administrator to assign vaults to tenants.

Default value for the **security vault** property is **enableSecurityVault** = *false*.

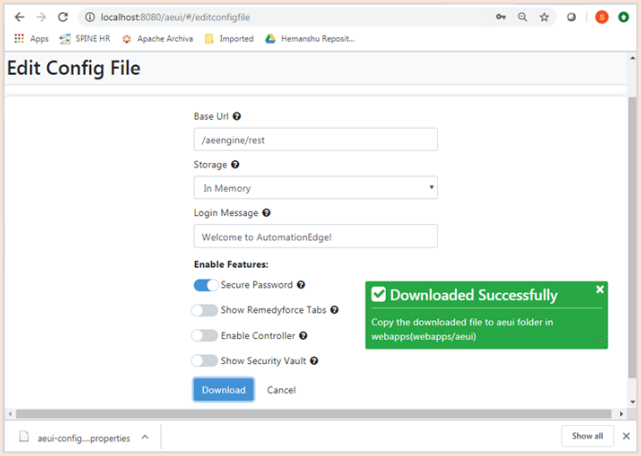
vi. **Captcha**

Use the property to enable captcha text on the AE login page. *Enabling the feature is optional.*

Default value for the **captcha** property is **enableCaptcha** = *false*.

To store **aeui-config.properties** in encrypted format:

1. Access the following URL, http://<AE server host>:8080/aeui/#/editconfigfile**.**
2. Configure the required AEUI properties and download encrypted file.



3. Replace the plain text file with the encrypted file.

Verify AutomationEdge UI sign-in

Configure all properties and verify the details as required by the environment, and then restart Tomcat service.

To verify AE UI sign-in:

1. Open any web browser. For example, Google Chrome.
2. Enter URL as *<http or https>://<IP or Host>:<PORT >/aeui*.
3. Sign in with the following user credentials, configured as part of seed data:

   username: *sysadmin*

   password: *vY78_h$ia*

If the sign-in is successful, reset the password for the Sysadmin user. After successful password reset, sign in again with the new password.

After installing the AE Server, sign-in as the System Administrator and complete the AE post installation tasks. For more details on AE Server post installation steps, see **AutomationEdge System Administrator's guide** and **AutomationEdge User's guide**.

Security Hardening

Security Hardening is the process of enhancing a server's or application's security by following the best practices and standards, which results in a very secure operating environment.

For more information, see https://docs.bmc.com/docs/security/basic-tomcat-security-configuration-recommendations-924057229.html.

Configure Tomcat with TLS (HTTPS)

For configuring Tomcat with TLS(HTTPS), see the Apache Tomcat documentation at https://tomcat.apache.org/tomcat-11.0-doc/index.html.

The following are links to some useful topics in the documentation:

- https://tomcat.apache.org/tomcat-11.0-doc/security-howto.html
- https://tomcat.apache.org/tomcat-11.0-doc/ssl-howto.html

Set up Tomcat with client authentication

Client authentication adds an extra layer of security to your Tomcat server. The authentication ensures that clients such Agents, Process Studios, or browsers must present a valid client certificate to connect to applications hosted on Tomcat.

Prerequisite: To enable client authentication, you must first configure SSL (HTTPS) and then configure Tomcat to verify client certificates using a Truststore.

For illustration purposes, we use the example of setting up Tomcat 11.

Step 1: To configure SSL in Tomcat, add the following certificate tag in the **server.xml** file located in the **bin** folder of Tomcat 11.

When enabling HTTPS in Tomcat 11, your connector configuration should be as follows:

```
<Connector port="8443" protocol="org.apache.coyote.http11.Http11NioProtocol"
        SSLEnabled="true">
 <SSLHostConfig>
  <Certificate certificateKeystoreFile="tomcat.p12"
            certificateKeystorePassword="tomcat"
            type="RSA"
            certificateKeystoreType="PKCS12" />
 </SSLHostConfig>
</Connector>
```

Step 2: Enable client authentication and add the following in **server.xml** file. The Truststore path specified in the **server.xml** file must contain all client certificates required to connect to the applications on this Tomcat server.

To enable client authentication, update the connector as follows:

```
<Connector port="8443" protocol="org.apache.coyote.http11.Http11NioProtocol"
        SSLEnabled="true">
 <SSLHostConfig certificateVerification="required"
            truststoreFile="client-truststore.jks"
            truststorePassword="client"
            truststoreType="JKS">
  <Certificate certificateKeystoreFile="tomcat.p12"
            certificateKeystorePassword="tomcat"
            type="RSA"
            certificateKeystoreType="PKCS12" />
 </SSLHostConfig>
</Connector>
```

When using self-sign certificates for client authentication, one needs to manually add the certificate to the trust store. The public key or the full self-signed certificate (usually in .crt format) is imported into the trust file.

In the case of a non-self-signed certificate, the trust file typically contains a list of trusted Certificate Authorities (CAs). Common clients include Agents, Process Studios, and browsers that access the AutomationEdge User Interface (AEUI) or other web applications.

If a client connects without a valid certificate that is not available in the Truststore, a bad certificate error appears.

Configure clients to provide certificates

All clients (Agents, Process Studios, browsers) must provide valid certificates while connecting to Tomcat applications.

To add client certificate to browser:

1. Open **Browser Settings**. For example, Chrome.
2. Go to Privacy & Security, click Security. Scroll down and click **Manage certificates** under the **Advanced** section.
3. In the **Certificate Manager** window, go to the **Your Certificates** tab. Click **Import**, browse, and select the client certificate file (for example, .p12 or .pfx format).

4. Enter the certificate password if prompted, click **OK** to complete the import. Upload Client Certificate.

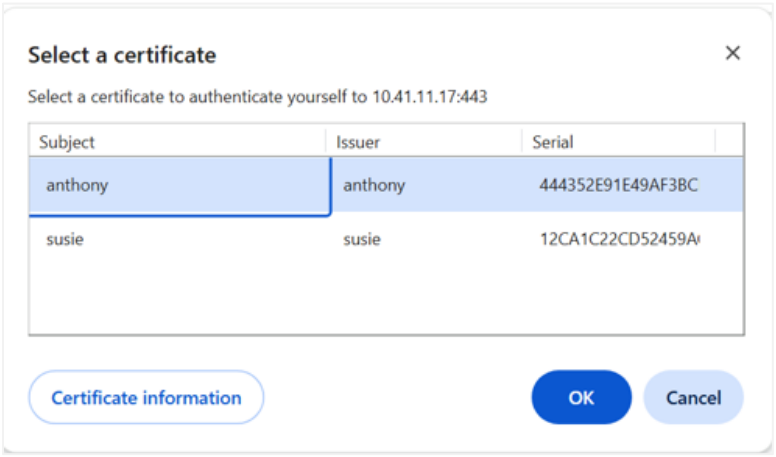
   When you import a client certificate into your browser, the certificate itself (public key) and the private key are imported.

> The **Manage Certificates** section may vary depending on the browser you are using.

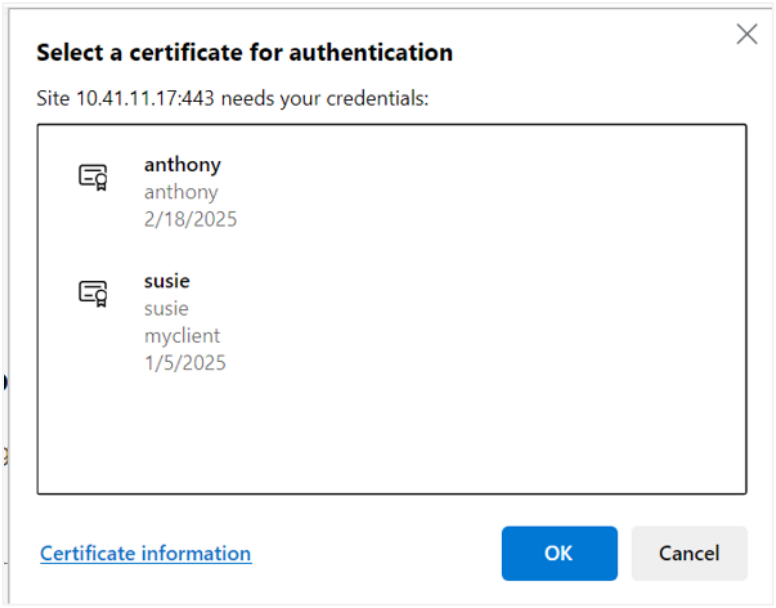
When you visit a Tomcat application, the browser will prompt you to select a certificate. If the certificate is trusted by the Tomcat server (present in its truststore), the connection will be allowed.

Use the same steps for other browsers.

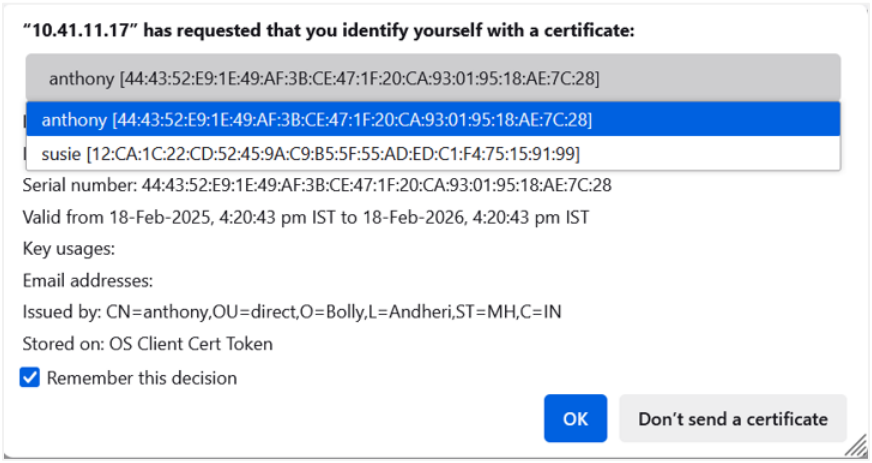
For example, select a certificate-Chrome browser



Select certificate- Edge browser.



Select a certificate – Firefox browser



Add client certificate to Agent and Process Studio

For Agent, you must pass the certificate information using JVM options.

Add the following JVM options:

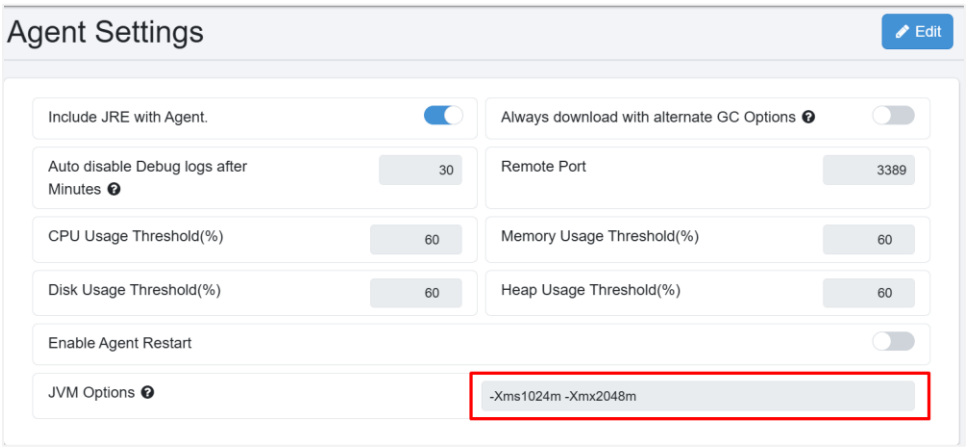
```
-Djavax.net.ssl.keyStore==<path to certificate>

-Djavax.net.ssl.keyStorePassword=<password of client certificate>

-Djavax.net.ssl.keyStoreType=PKCS12
```

Instead of providing a hardcoded path for the client certificate, the user can place the certificate in a folder within the Agent Home and specify the path as follows:

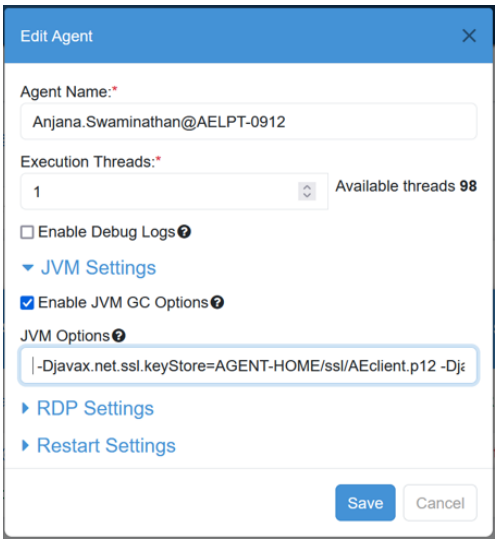
-Djavax.net.ssl.keyStore=AGENT-HOME/ssl/AEclient.p12

For Agent:

You can add these JVM options from the Agent UI in the Agent JVM Options section, available under the Agents tab in Agent Settings on the AEUI. These settings are retained even after the Agent is upgraded.

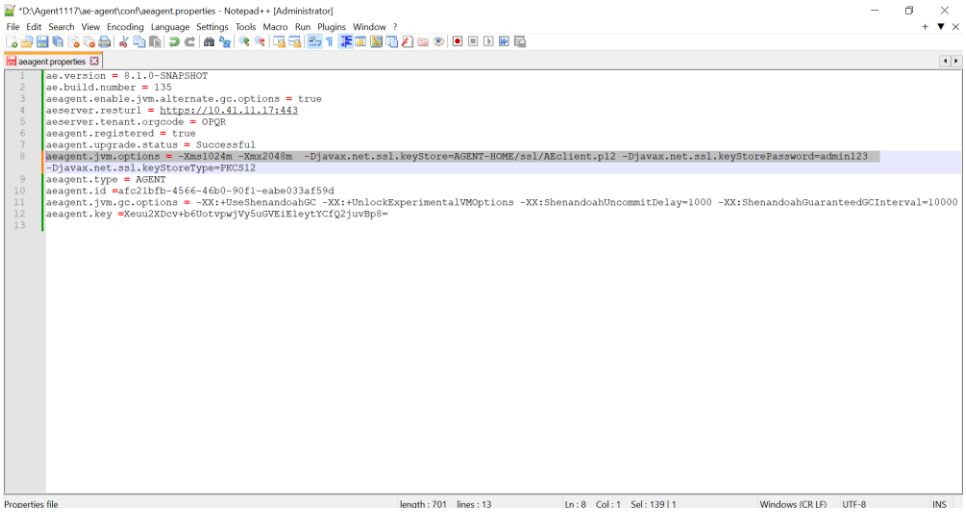


Alternatively, click **Edit Agent** under the **Agent List** on the **Agents** tab, and add the options in the **JVM Options** section.



When you configure JVM settings from the user interface, they are saved in the **aeagent.properties** file located in the AGENT_HOME/conf directory.

After the client certificate details are added to the agent properties file, the agent includes the client certificate information each time agent communicates with the server.



For Process Studio:

Add the jvm options to the setenv.bat file which is in the **bin** folder of Tomcat.

Ensure that this file is not overwritten during future upgrades.

It is recommended to always take a backup of this file before performing an upgrade.

For example, for client auth configuration in set-process-studio-env.bat see the following:

```
set OPT=-Djavax.net.ssl.keyStore=..\ssl\AEclient.p12 -
Djavax.net.ssl.keyStorePassword=admin123 -Djavax.net.ssl.keyStoreType=PKCS12
```

After applying the settings, the Agent or Process Studio will present the client certificate to the Tomcat server. If the certificate is valid and trusted, register and start agent.

Setup nginx with client authentication

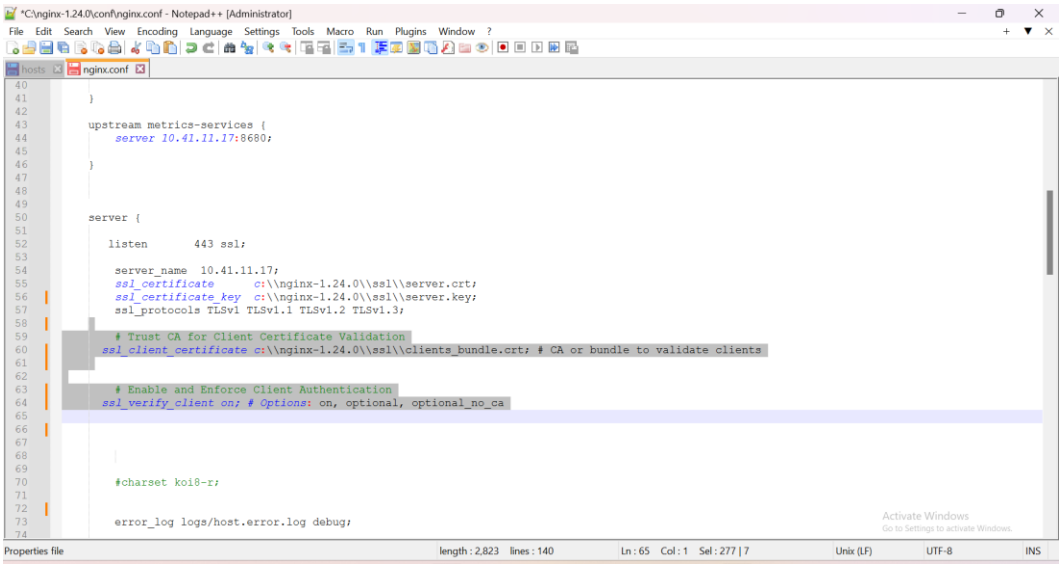
The ssl_client_certificate in nginx specifies the location of the trusted Certificate Authorities (CAs) that NGINX uses to verify client certificates.

If you are using self-signed certificates, this CA list typically consists of a bundle of trusted public client certificates.

To create a bundle of client certificates:

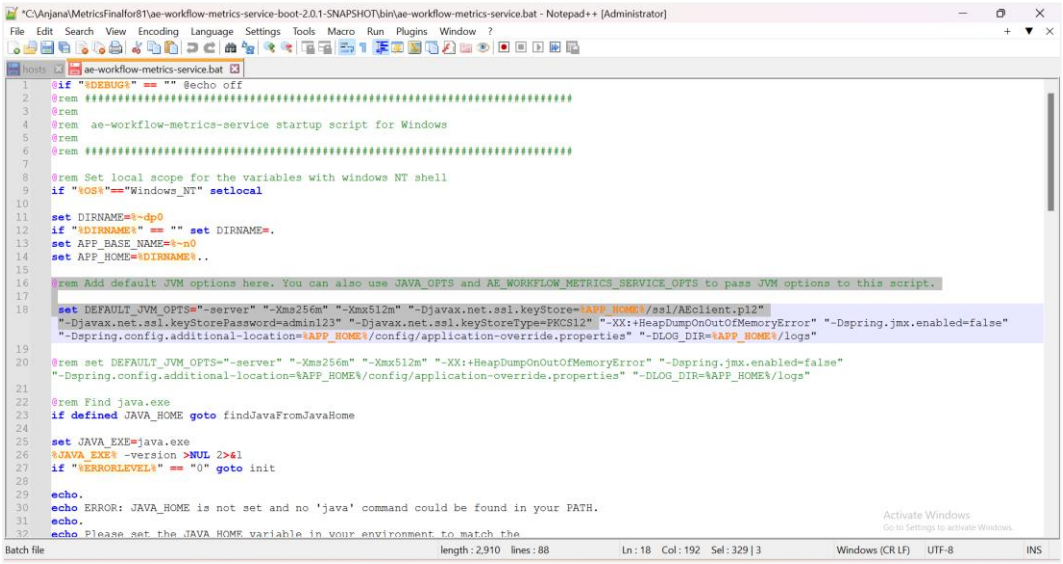
```
type client1.crt client2.crt client3.crt > clients_bundle.crt
```

The ssl_client_certificate directive accepts only a single file. If you have multiple client certificate files (.crt), you must concatenate them into a single file.



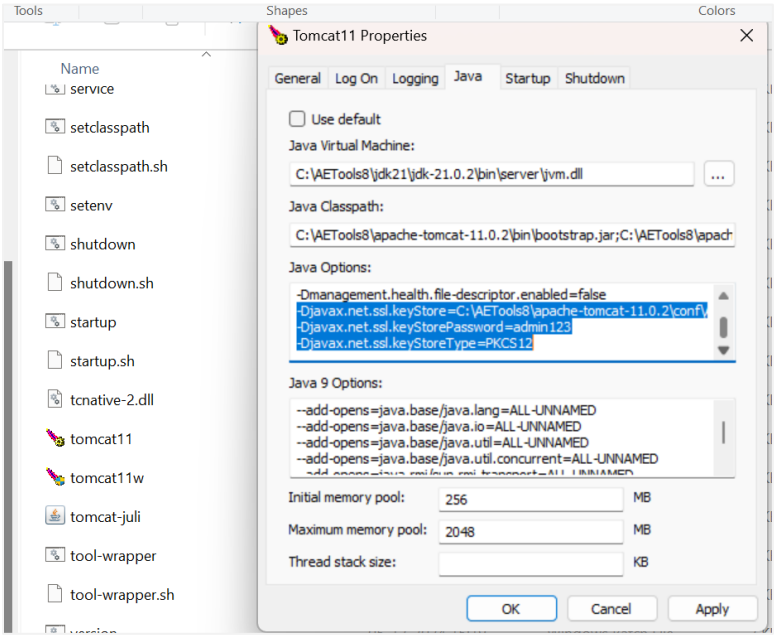
Add the client certificate details to the DEFAULT_JVM_OPTS variable in the ae-workflow-metrics-service.bat file.

This enables the Metrics Service to include the client certificate when communicating with the NGINX server.

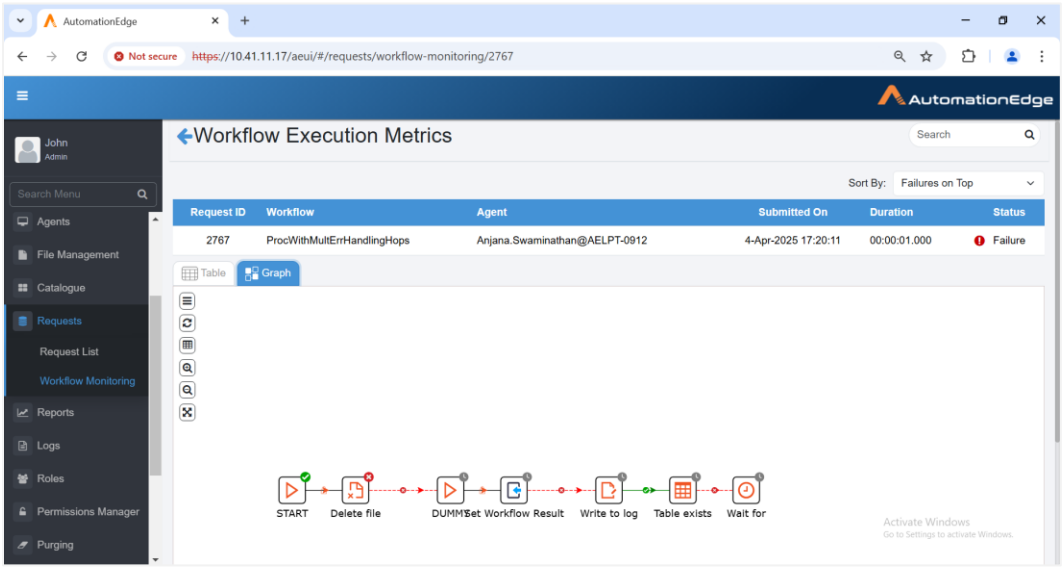


You must specify the client authentication details in the Tomcat service configuration to retrieve workflow metrics data on the **Workflow Monitoring** page.

In this scenario, tomcat acts as a client to the NGINX server and must send the client certificate during communication.



Once all the required client authentication configurations are done in the Agent, Metrics Service, NGINX, and Tomcat, the metrics data is successfully retrieved and displayed in AEUI.



Security settings for ActiveMQ

For configuring Apache ActiveMQ security settings, see the following links:

- https://activemq.apache.org/components/classic/documentation/how-do-i-use-ssl
- https://activemq.apache.org/components/classic/documentation/security
- https://activemq.apache.org/components/classic/documentation/encrypted-password
- https://activemq.apache.org/components/classic/documentation/using-activemq-classic

After completion of security hardening activities configure the ActiveMQ authentication setup for AE.

ActiveMQ authentication for AE

ActiveMQ authentication is not enabled by default for AE. Configure the following settings in the AE engine.

- Simple authentication plugin is used to set authentication for ActiveMQ. Encrypt the password and store it safely in configuration files. Following is the encrypt password command:

  $bin/activemq encrypt –password activemq –input mypassword

- Where the password you want to encrypt is passed with the input argument, the password arguments is a secret used by the encryption.
- You get an encrypted password in the output. For example, **Encrypted password:** *K6lz7RKJg9DzGiX8eRbvUw==*
- In the **<ActiveMQ home>/conf/credentials-enc.properties** file, add the following: *aeuser.password=ENC(K6lz7RKJg9DzGiX8eRbvUw==)*

- You need to edit the **<ActiveMQ home>/conf/activemq.xml** file.
- You need to enable scheduler support to true. For example:

```
<broker xmlns="http://activemq.apache.org/schema/core" brokerName="localhost"
dataDirectory="${activemq.data}" useJmx="true" schedulerSupport="true">
```

  - Delete the following bean definition:

```
<bean class="org.springframework.beans.factory.config.PropertyPlaceholderConfigurer">
<property name="locations">
    <value>file:${activemq.conf}/credentials.properties</value>
</property>
</bean>
```

  - Add the following bean definitions:

```
<bean id="environmentVariablesConfiguration"
class="org.jasypt.encryption.pbe.config.EnvironmentStringPBEConfig">
        <property name="algorithm" value="PBEWithMD5AndDES" />
<property name="passwordEnvName" value="ACTIVEMQ_ENCRYPTION_PASSWORD"/>
</bean>
<bean id="configurationEncryptor" class="org.jasypt.encryption.pbe.StandardPBEStringEncryptor">
        <property name="config" ref="environmentVariablesConfiguration" />
</bean>
<bean id="propertyConfigurer" class="org.jasypt.spring4.properties.EncryptablePropertyPlaceholderConfigurer">
<constructor-arg ref="configurationEncryptor" />
<property name="location" value="file:${activemq.conf}/credentials-enc.properties"/>
</bean>
```

  - Add **<simpleAuthenticationPlugin>** tag under broker element in the **activemq.xml** file.

```
<plugins>
        <simpleAuthenticationPlugin anonymousAccessAllowed="false">
        <users>
                <authenticationUser username="admin" password="${aeuser.password}"
groups="users,admins"/>
      </users>
   </simpleAuthenticationPlugin>
</plugins>
```

- Set up ACTIVEMQ_ENCRYPTION_PASSWORD.

After you have configured the authentication for ActiveMQ, set the ACTIVEMQ_ENCRYPTION_PASSWORD environment variable.

- To set environment variable on **Windows** machine:
  a. Under **System Variables**, add **New** variable.
  b. Enter the variable name as **ACTIVEMQ_ENCRYPTION_PASSWORD** and enter *activemq* as the variable value. For example, **ACTIVEMQ_ENCRYPTION_PASSWORD**=*activemq*.
- To set environment variable on **Linux** machine:

Ensure you have the root privilege to create a file under **/etc/profile.d**.

i. Create new file under directory **/etc/profile.d** which will set the environment variable. Following is the syntax for creating a file:

$cd /etc/profile.d/

$sudo touch automationedge.sh

Creating the new file enables all users using Linux machines to access the environment variable with ease.

ii. Edit **automationedge.sh** and add the following:

export ACTIVEMQ_ENCRYPTION_PASSWORD=activemq

iii. Save the file and restart the machine.

- To edit **ae.properties** file in **<AE home>**.
  i. Open **ae.properties** file and find properties **mq.username** and **mq.password**.

mq.username and **mq.password** properties are empty by default.

ii. Enter the following details under **ActiveMQ credentials**.

#ActiveMQ credentials

mq.username=aeuser

mq.password=mypassword

- Restart services for ActiveMQ and Tomcat.

ActiveMQ: Enable JMX with authentication

To enable JMX follow the instructions at
https://activemq.apache.org/components/classic/documentation/jmx

You must ensure that only the owner has read and write permissions on the JMX file, since it contains the passwords in clear text. For security reasons, the system checks that the file is only readable by the owner and exists with an error if it is not. Therefore, in a multiple-user environment, you should store the password file in private location such as your home directory.

- **Linux**

  chmod 600 jmx.password

- **Windows**

  https://docs.oracle.com/javase/1.5.0/docs/guide/management/security-windows.html

AE with SSL

AE supports secure URL. As AE is hosted on Apache Tomcat, modify the configuration to secure Tomcat server with TLS.

You don't need to modify the configuration, if you have already completed configuring Tomcat for TLS (HTTPS).

Migrate from previous release

In the chapter, you will learn about migrating AutomationEdge from release *7.*.** to the current release.

For migration scenarios from releases prior to *7.*.** to release *7.*.*, see the **AutomationEdge_R7.7.3_Installation guide**.

The chapter includes the following topics:

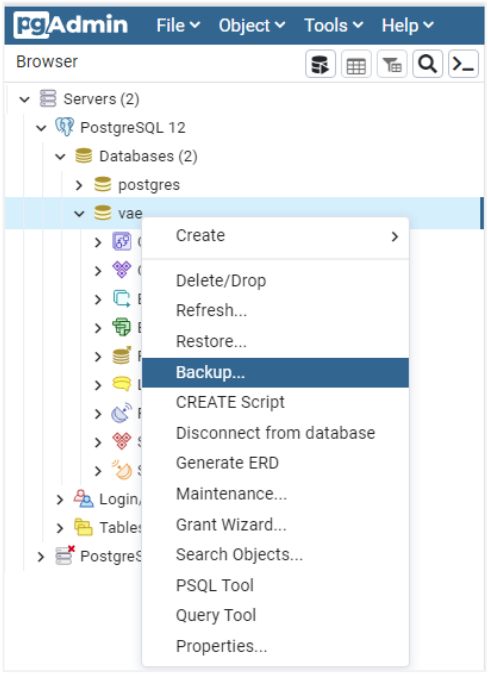
Pre-migration activities

Before you start the migration process, ensure that you:

- Take the backup of your database. For example, if your database is **PostgreSQL**, then:

  Sign in to the database with your username and password, and then go to the **database → right-click and select Backup**.

  Save the database backup in a location of your choice.



- Ensure that none of the **Requests** are in *New* or *Execution Started* status.

If a workflow is in *Execution Started* state, then we recommend that the execution is completed before you start with the migration.

i. Stop Tomcat before proceeding with the software installations.

ii. Upgrade the following software:

- Upgrade to Java version 21.

  Setup the environment with OpenJDK 21 JRE for Tomcat/ActiveMQ for AutomationEdge R8.1.1 onwards. For details, see Install Java.

- Upgrade the database version. For installation details, see Install database.

- Upgrade the Apache ActiveMQ version.

  If ActiveMQ is using RDBMS storage, follow the steps as mentioned in ActiveMQ about adding 4 jars from https://mvnrepository.com/.

- Start ActiveMQ and test by storing sample message from the admin console.

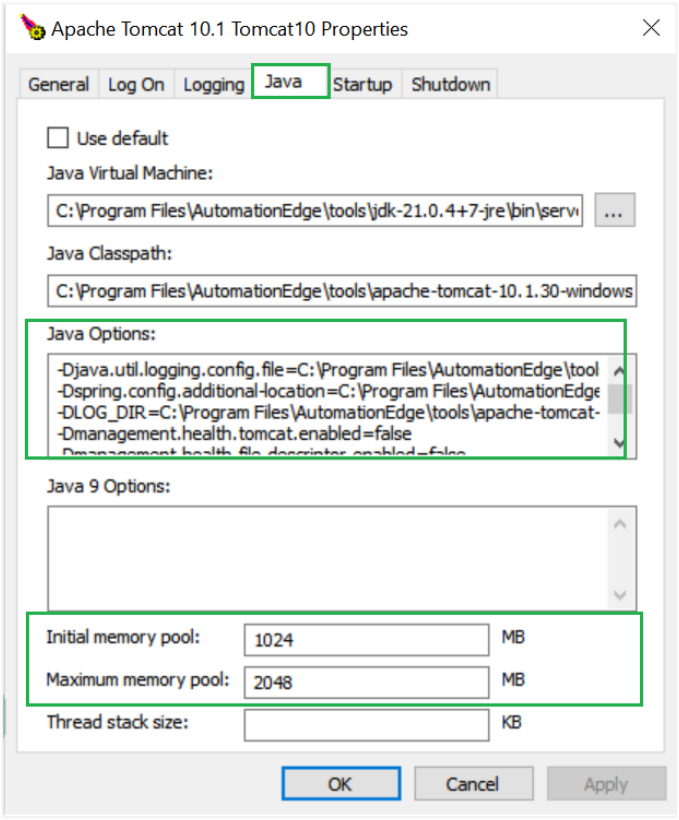
- Upgrade the Apache Tomcat version. For installation details, see Install Apache Tomcat.

  If Tomcat services have not been configured for JRE 21 perform the following:

  Tomcat Services need to be either recreated or reconfigured after migration to JRE 21 version.

  Following are the steps to reconfigure Tomcat:

  - Stop Tomcat if not already stopped.
  - Locate and open execute Tomcat Monitor Application <TomcatServiceName>w.exe available at <Tomcat Home\bin>. For example, **tomcar11w.exe** or **AE-Tomcat-11w.exe** in case AE is installed using the AE installer in **E:\tools\apache-tomcat-11\bin**.
  - It opens the Tomcat Monitor Application.
  - Following dialog appears, change the Java Virtual Machine Path to point to the JRE 21 jvm.dll available in **<JRE_HOME>/bin/server**.



iii. Pre-migration activities for Agents:

- **Approach 1:** Agents can be bundled with JRE 21 from the AutomationEdge Server.

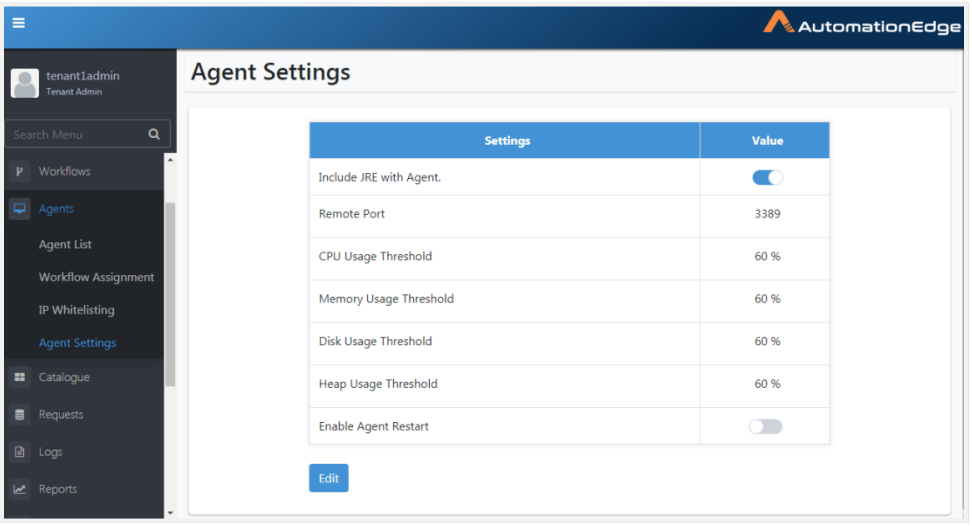
  OR

- **Approach 2:** Agents can use system JRE 21.

We recommend you follow **Approach 1**, where the bundled JRE in Agent is automatically upgraded from JRE binaries on AE server in future upgrades as well.

- Following are the actions so that Agents are bundled with JRE 21 from AutomationEdge server:

  i. In the AE UI, navigate to **Agents à Agent Settings** page. On the page toggle the **Include JRE with Agent** switch to enable the bundling of JRE with the Agent.



When the system migrates to 8.1.1, Agents will automatically upgrade and migrate to JRE21 supplied by AE.

ii. Stopping Agents:

- For AutomationEdge releases prior to 5.4.0, stop all Agents manually as a first step of migration.
- For Release 5.4.0 onwards Stopping Agents is optional - provided 'Include JRE with Agent' (option under **Agents à Agent Settings**) is enabled.
- **Artefacts:** Make sure you have all the artefacts of the latest AE release and a valid AE license.

AutomationEdge Server migration

War Files Deployment

Backups

You must take backup of the following:

- Backup AutomationEdge database. See the AutomationEdge_R8.1.1_Postgre_SQL_Backup guide for Postgres database backup.
- Backup AutomationEdge home.
- Backup AutomationEdge internal configuration files from <Tomcat home>/webapps/aeengine/WEB-INF/classes, especially if they were modified to suit the customer environment.
    - restvalidation.json

Delete files

- *Delete vae.war/aeengine.war*, *vaeinterface.war/remedyforceinterface.war* , aeintegrationservice.war and *aeui.war* from <Tomcat home>/webapps directory.
- Delete *remedyforceinterface.war* (if present. Relevant only if *remedyforceinterface.war* is deployed) from <Tomcat home>/webapps directory.
- Delete directories vae/aeengine, vaeinterface/remedyforceinterface and aeui from <Tomcat home>/webapps

Copy Files

Once all the war files and the corresponding directories are deleted, copy the new war files ***aeengine.war*** and ***aeui.war*** to <Tomcat home>/webapps

Start Tomcat

Start the Tomcat Service so that the war files are extracted in **webapps** directory.

Update Files

Update the following internal properties files in **<Tomcat home>/webapps/aeengine/WEB-INF/classes**.

- application.properties
- restvalidation.json
- configuration.properties

Do not replace these files from the backup. It may result in unstable application deployment.

Restart Tomcat

Restart the Tomcat service if there are any updates to the internal properties files above to make the changes effective.

Security Hardening

If settings are not done previously, for details, see Security Hardening.

Post Migration Steps

In the section, we discuss the post migration steps required with respect to - License, Agents, Plugins & Workflows and Workflow Schedules.

License

In case you do not have a compatible license, sign in as an *Admin* and upload and apply the AE license, compatible with the current release to use license features like Advanced Agents, Turbo Agents, and Assisted Agents.

Agents

Post migrations steps for Agents are required only if the Pre-migration activities for Agents are not performed.

In case of Agent upgradation failure, use any of the following two options.

We recommend **Option 1**.

Option 1 (Agent is bundled with JRE from AE server - preferred option)

Complete the following settings in AE UI, if **Include JRE with Agent** is not enabled:

- In the AE UI, navigate to **Settings Agent Settings** and toggle the **Include JRE with the Agent** switch.
- If case Agents throw an error during upgradation, then set the following property-
    - Open the application.properties file present under AGENT_HOME/conf folder
    - Change the property agent.upgrade.status value to Successful

Option 2 (Agent uses System JRE)

We recommend you follow **Option 1**, where the bundled JRE in Agent is automatically upgraded from JRE binaries on AE server in future upgrades as well.

If "Include JRE with Agent" is disabled before and even after AE Server migration, then Agent can use System JRE.

Complete the following steps:

- Setup JRE 21 on the Agent machines manually. For details, see Install Java.
- Open the application.properties file present under AGENT_HOME/conf folder.
- Check agent.upgrade.status. If the value is other than Successful, change the value to Successful.

Start Agents

Any stopped Agents prior to upgrade should be started.

Post Agent Startup

- All Agents start as Standard Agents. You may edit the Agents to Advanced or Turbo Agents. You may change Agent Mode depending on availability in license subscribed. However, Agent Mode cannot be changed when an Agent is in Unknown state.

- Agents acquire Tenant level Proxy settings if any during upgrades, discarding the Agent level proxy settings. The required changes need to be done manually.
- If you are running Agent as Administrator during restart after upgrade, it does not start Agent as Administrator. If you wish to run the Agent as Administrator, once all the Agents are upgraded you need to shut the Agent and restart Agent as Administrator.

Plugins & Workflows

- As a part of a migration to this Release 8.1.1, all the Process Studio workflows are made inactive.
- This is due to framework changes. If server is updated such that plugins do not match supporting framework versions on AE server, all the workflows using such plugins are disabled. User will be able to enable these workflows after new plugins are uploaded.
    - User should upload the new zip of Plugins 4.0 or higher provided in the release package. Until the new plugins are uploaded onto the server, user will not be able to make the workflows active again. Once new plugins are uploaded the workflows can be enabled*.
    - Login with System Administrator and upload Plugins zip files.
- Now activate all workflows.

When uploading PS-plugins using zip, you may get errors for **OOTB**, **Advanced Rest Client**, or **WebGUI** plugins, stating *Duplicate step or process entry name*.

Workaround:

Plugins distribution contains the following zip files:

- Data-plugins-4.0.zip
- IT-plugins-4.0.zip
- ITSM-plugins-4.0.zip
- MachineLearning-plugins-4.0.zip
- PS-plugins-4.0.zip
- Script-plugins-4.0.zip
- Cloud-Storage-4.0.zip

The PS-plugins-4.0.zip contains several jars, including:

- gui-automation-4.0.jar
- advanced-rest-client-4.0.jar

For PS-plugins.zip you might get errors while uploading. There are two possible workarounds as follows. You may follow any one of these.

- Upload two new plugin jars followed by PS-plugins.zip
    - Upload gui-automation plugin Jar (gui-automation-4.0.jar) independently
    - Then upload advanced rest client plugin jar (advanced-rest-client-4.0.jar)
    - Then use the PS-plugins-4.0.zip to update rest of the plugins
    - Upload PS-plugins-4.0.zip again and again, until all plugin specific errors are resolved. Typically, you might have to upload it thrice.

Schedules

Repeat interval values for minutes from Release 5.4.0 onwards, is restricted to 1, 2, 3, 4, 5, 10, 15, 20, 30, 45. As a part of AE Server migration, Schedules having minute's interval other than the allowed values mentioned previously are disabled. Users need to update the interval with valid values after migration and then enable such schedules.

Database Drivers

We ship JDBC jars for all three databases we support, that is, PostgreSQL, Oracle, and MSSQL Server with the application.

APPENDICES

Appendix 1: AE file system

The appendix lists some important files in AE file system.

Following are the components of the AE Tools directory:

1. **Active MQ HOME**
   i. You can mention, AMQ_HOME/data/kahadb is a storage directory for AMQ messages.
   ii. Active MQ HOME \conf\activemq.xml is the Active MQ configurations file.
   iii. Active MQ HOME\data directory contains the log files:
      - activemq.log
      - audit.log
2. **Apache Tomcat Home**
   i. Apache Tomcat Home\logs directory contains automationedge.log file.
   ii. Apache Tomcat Home\webapps directory is used to deploy:
      - aeangine.war
      - aeui.war
   iii. Apache Tomcat Home\webapps\aeui directory has configuration file to set AE UI properties: aeui-config.properties
   iv. Apache Tomcat Home\webapps\aeengine\WEB-INF\classes contains aeengine internal properties files.
      - hibernate.properties
      - log4j2-spring.xml
3. **AutomationEdge Home directory**

   Directory path: <preferred_drive>:/AutomationEdge/aehome

   AutomationEdge Home

   - **workflow**: This folder stores the workflow supporting zip files.
   - **AE_TEMP**: This directory is the temporary storage for temporary files or resources used by AutomationEdge, which are shared across Multiple AE Servers in HA. (Note: If resources are not shared, then a local temp folder should be used instead of one under AE_HOME).
   - **conf**: This folder contains the following configuration files:
      - ae.properties
      - database.properties

- **files**: This folder stores the run time input and output files to workflows
- **psplugins**: This folder contains all the plugins uploaded to AutomationEdge server.

4. AutomationEdge Enable Debug Logging

Set the log level values for AE Server, Agent, and Integration Service by modifying log-level details in the respective files.

You need not restart Tomcat service or agent.

i. AE Server

Apache Tomcat Home/webapps/aeengine/WEB-INF/classes/log4j2-spring.xml

ii. Agent

Agent Home/conf/log4j2.xml

After saving the modified files, DEBUG logs appear in the log4j2.xml file.

Appendix 2: AE installation on CentOS

In the appendix, you will learn about installation on Red Hat Enterprise Linux (RHEL) based Linux distributions, such as CentOS, Fedora, and OEL.

The steps and commands were performed on CentOS7.

Environment check

Perform an environment check for the following:

- Hostname-IP address resolution
- Firewall settings
- Steps and commands performed in CentOS

Prerequisites

Install and configure the following software:

- Java
- PostgreSQL with empty **vae** database
- Apache Tomcat
- Apache ActiveMQ

Make necessary changes to set up the CentOS environment.

AE installation

For AE installation, create a directory structure and database for AE, and deploy Automation engine and UI war files on Tomcat server.

Create AE directory structure

Create the top-level directory, AE Base Directory in which all AE software and components are installed. You need to create the following directories and setup AE_HOME environment variable:

AutomationEdgeBase <preferred_drive>:/AutomationEdge.
For example, /home/<username>/Automationedge
AutomationEdge Tools <preferred_drive>:/AutomationEdge/tools.

For example, /home/<username>/Automationedge/tools

AutomationEdge Home <preferred_drive>:/AutomationEdge/aehome.

For example, /home/<username>/Automationedge/aehome

Set environment variable

AE_HOME= ”<path for the AE working directory>”.

For example, AE_HOME=/home/<username>/Automationedge/aehome

War files deployment

Following are the steps to deploy war files:

1. Delete all out-of-the-box apps directories from Tomcat, **/tools/apache-tomcat-11/webapps**.
2. Copy the following .war files in the **/tools/apache-tomcat-11.0 /webapps** directory.
   - **aeengine.war**
   - **aeui.war**
3. Restart the Tomcat service to extract **.war** files in the **webapps** directory, and database is populated with objects.
4. If Postgres DB user password is different, change it in file **database.properties**.

Appendix 3: AE installation on Ubuntu

In the appendix, you will learn about installation on Debian based Linux distributions, such as Ubuntu, OpenSuse, and Kali.

The steps and commands were performed on Ubuntu 18.

Environment check

Perform an environment check for the following:

- Hostname-IP address resolution
- Firewall settings
- Steps and commands performed in Ubuntu 18

Prerequisites

Install and configure the following software:

- Java
- PostgreSQL with empty **vae** database
- Apache Tomcat
- Apache ActiveMQ

Make necessary changes to set up the Ubuntu environment.

AE installation

For AE installation, create a directory structure and database for AE, and deploy Automation engine and UI war files on Tomcat server.

Create AE directory structure

Create the top-level directory, AE Base Directory in which all AE software and components are installed. You need to create the following directories and setup AE_HOME environment variable:

AutomationEdgeBase <preferred_drive>:/AutomationEdge.
For example, /home/<username>/Automationedge
AutomationEdge Tools <preferred_drive>:/AutomationEdge/tools.
For example, /home/<username>/Automationedge/tools

AutomationEdge Home <preferred_drive>:/AutomationEdge/aehome.

For example, /home/<username>/Automationedge/aehome

Set environment variable

AE_HOME= "<path for the AE working directory>".

For example, AE_HOME=/home/<username>/Automationedge/aehome

War files deployment

Perform the following steps to deploy war files:

1. Use the following command to change the directory to **webapps**:

   ```
   cd "/home/<user_name>/AE_Tools/apache-tomcat-11.0/webapps"
   ```

2. Use the following command to delete all out-of-the-box apps directories or files or folders from the **webapps** directory.

   ```
   rm -rf *
   ```

3. Copy AE war files to the **webapps** directory.

   Use WinSCP tool if you are copying from Windows to Linux.

4. Use following command to rename **aeengine.war** as follows:

   ```
   mv aeengine-8.1.1.war aeengine.war
   ```

5. Use following command to rename **aeui.war** as follows:

   ```
   mv aeui-8.1.1.war aeui.war
   ```

6. Ensure the following are running:

   i. ActiveMQ: Start using super user credentials.

   ii. PostgreSQL service.

   If Postgres DB user password is different, change it in file **/tools/automationedge/conf/database.properties**.

   iii. Tomcat: Start using super user credentials.

   Restart the Tomcat service so that .war files get extracted in **webapps** directory, and database is populated with objects.

==========END OF DOCUMENT==========