Oracle® Retail MICROS Stores2

Functional Document
Stores2 – Data Privacy
Release 1.39.4

March 2018

ORACLE

Oracle® Retail MICROS Stores2 Functional Document Data Privacy, Release 1.39.4

Table of Contents

Note: The rebranding for the latest version of this documentation set is in development as part of post MICROS acquisition activities. References to former MICROS product names may exist throughout this existing documentation set.

Introduction

Purpose

This document describes how to manage the data privacy of customers, vendors, and employees on Stores2, starting from base version 1.39.4.

Privacy requests

Definition

The retailer needs to be able to manage requests received from customers, vendors, and employees related to their personal information.

Requests could be:

- Access data: the requester asks to be informed about the personal data available in the retailer systems;
- Forget data: the requester asks to have all their personal data removed from the retailer systems.

The request will be managed in two different steps:

- Request creation: actor is typically the store's user that creates it;
- Request confirmation: actor is typically the data privacy officer at the headquarters (HQ).

Request creation

General behavior

A new specific function allows the user to submit the requests.

For a new entry, it is required to identify the subject type and then the user is asked to select a customer, a vendor, an employee (both from the company and from shop), a user, a cashier, and an associate.

At the end, the request type and optionally a note should be filled in by the user.

The final result is a submission of the request in a queue table that will be sent to the HQ for processing. Based on the relationship between the employees' tables, when an element is selected, all the other related tables are also included.

The user interface will initially show the list of the requests already submitted. This will allow the user to add, edit, or void a previously submitted entry. When sent (via S2_Exchange), the records will no longer be visible in the list.

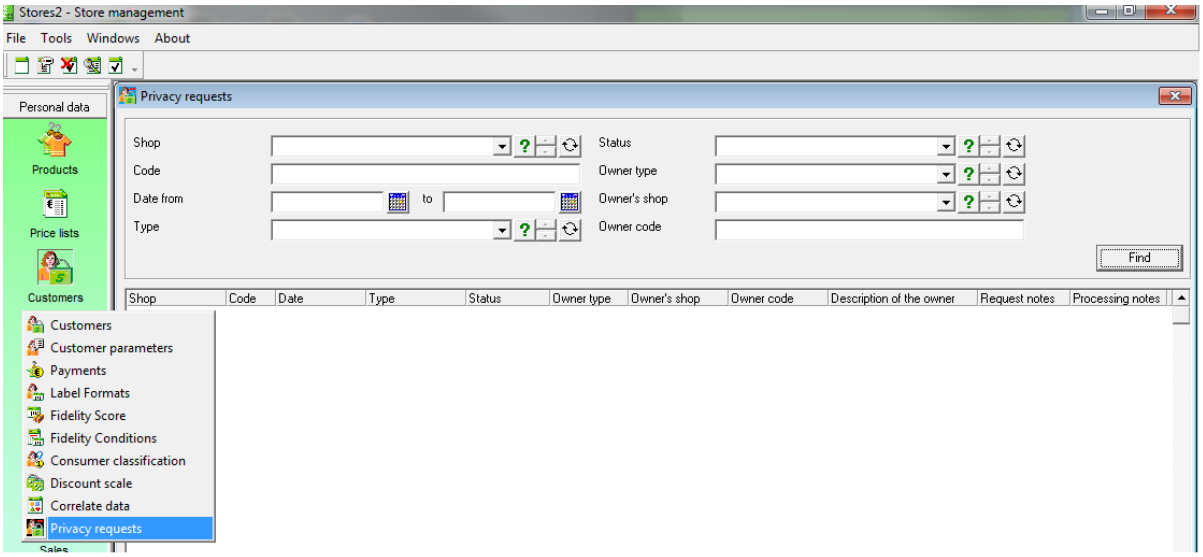
There is also a search form with a list of the requests.

The user has the ability to filter the type of request (access or forget), a range of dates (when the record was created), a list of status (open, rejected, submitted, blocked, completed), selection for the types of the nominatives, and a string for the nominative (to be used for all the types of nominatives).

The list will show the type, date, status, nominative, and user. Data will be ordered by the date/time of submission in ascending order.

Request wizard

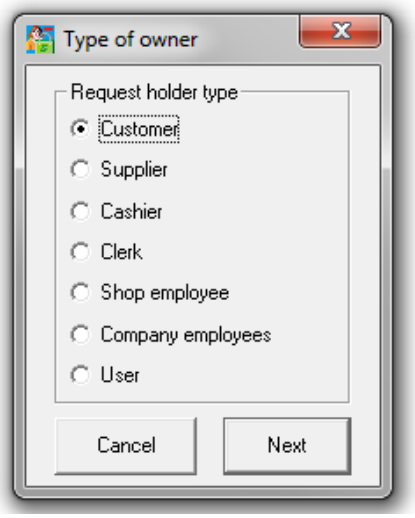
If the user is enabled, from the “Customers” menu, it is possible to access the “Privacy requests” user interface.



It is possible to create a new request by clicking in the blank area with the right mouse button and then selecting “New”.

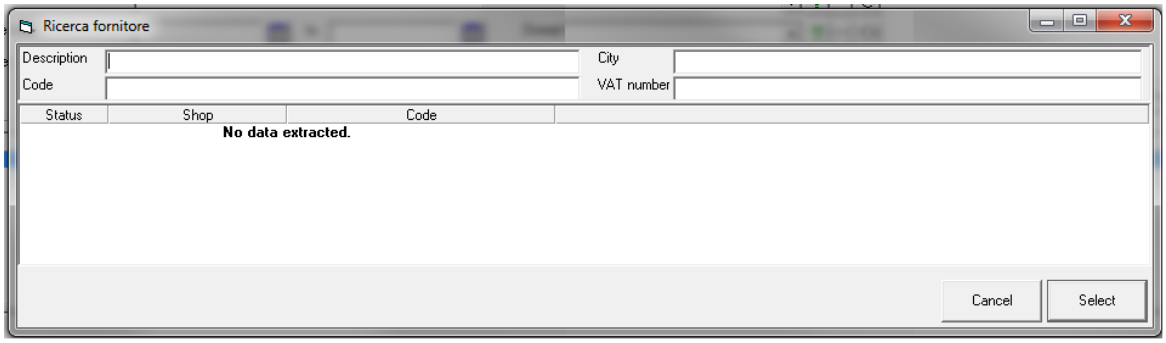
In the first step, the user will be asked to select the right requester type as shown in the image below.

From the shop, only the “Customer” option will be available. From the HQ, all options can be used.

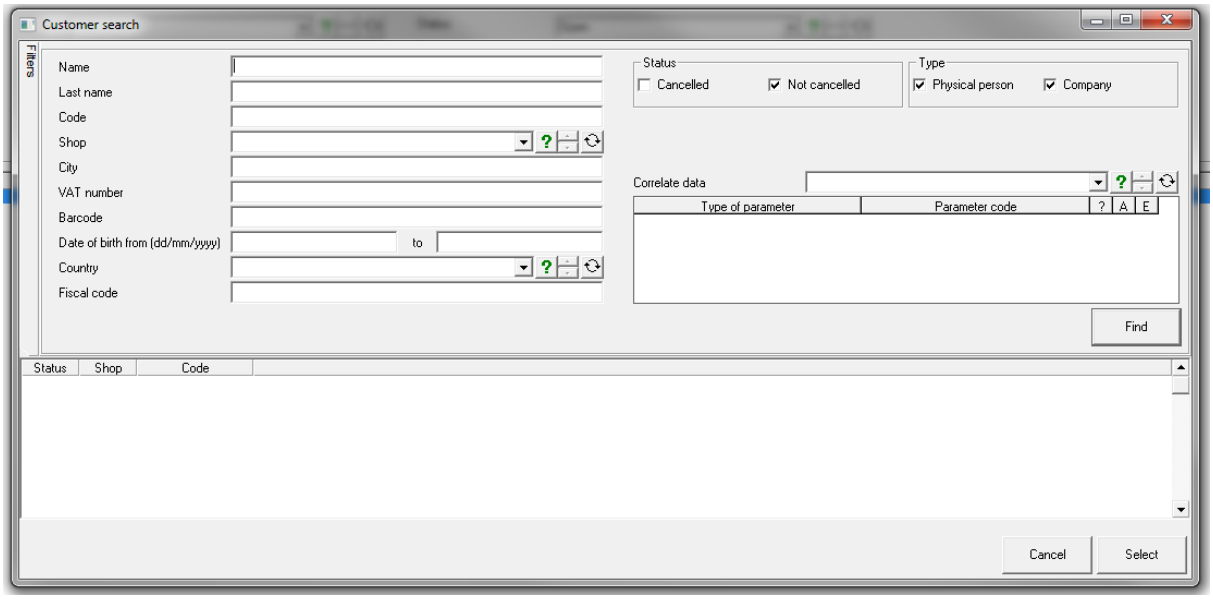


The second step will be the research of the requestor; each kind of requestor has its own research.

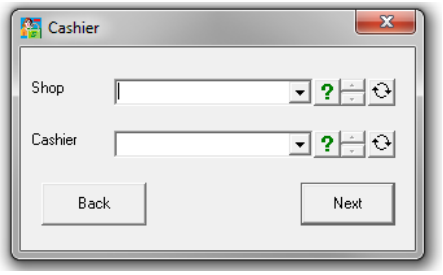
Vendor:



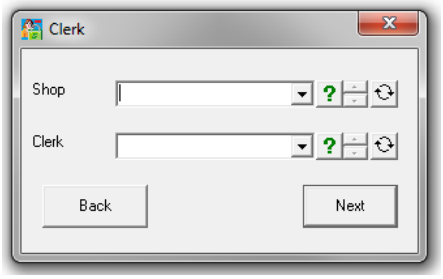
Customer:



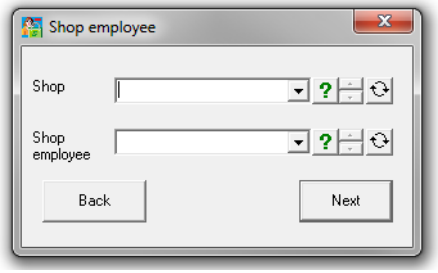
Cashier:



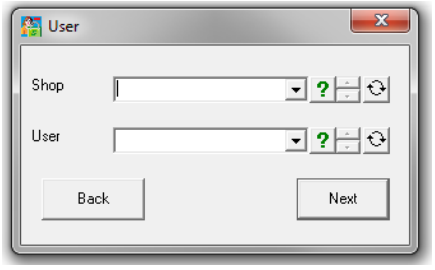
Clerk:



Shop employee :

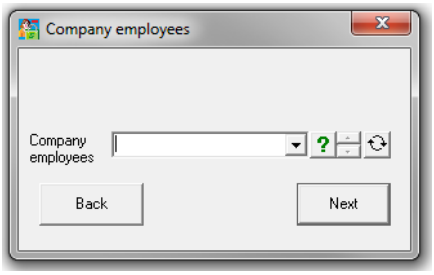


User:



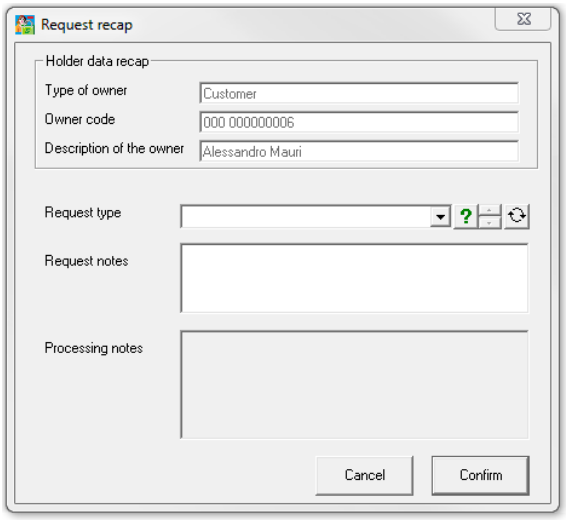
Privacy requests

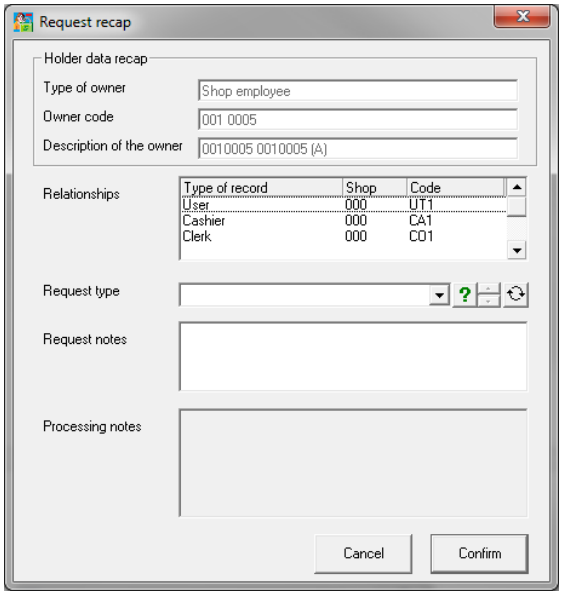
Company employee:



After the selection of the requester type, the program will provide the last step, where the application will show a recap of the request and ask the user to select the request type (access data or forget data) and, optionally, to fill in some notes.

For cashiers, associates, shop employees, and users, relationships will be shown.





After the selection of a button, the application will request a confirmation from the user; in case of missing notes, an additional request will be prompted to the user.

If the request was inserted from the shop, it will be visible in the HQ after the data transfer via S2_Exchange (generally twice a day).

Once the request is sent to the HQ, it will no longer be visible from the shop.

Requests confirmation

User interface

From the same user interface shown above, it is possible to process requests.

Requests can be processed only in HQ and only from the users enabled.

The user has the capability to filter for:

- The store where the request was created;
- The ID of the request;

- The range of dates (when the record was created);
- The request type;
- The request status (open, rejected, submitted, blocked, completed);
- The type of the requester;
- The shop and code of the requester.

The list will show:

- The store where the request was created;
- The ID of the request;
- The range of dates (when the record was created);
- The request type;
- The request status;
- The type of the requester;
- The shop and code of the requester;
- The description of the requester;
- Request notes;
- Processing notes.

Data will be ordered by the date/time of submission in ascending order.

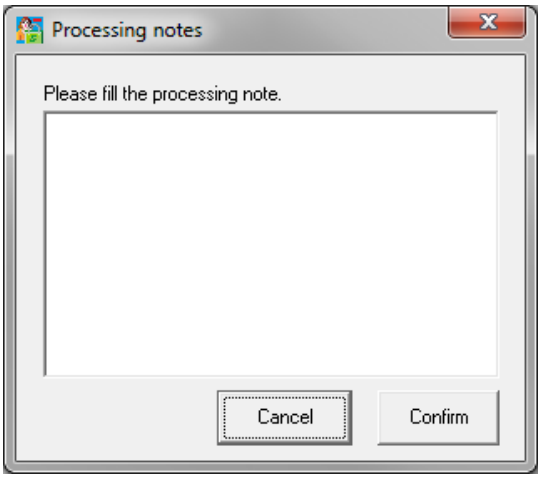
The user will have the available tools:

- Property\Check: display details;
- Reject the request: only applicable if the status is “Open” or “Blocked”
- Process the request: only applicable if the status is “Open”, “Rejected”, or “Blocked”

Multiple selection is allowed.

Reject the request

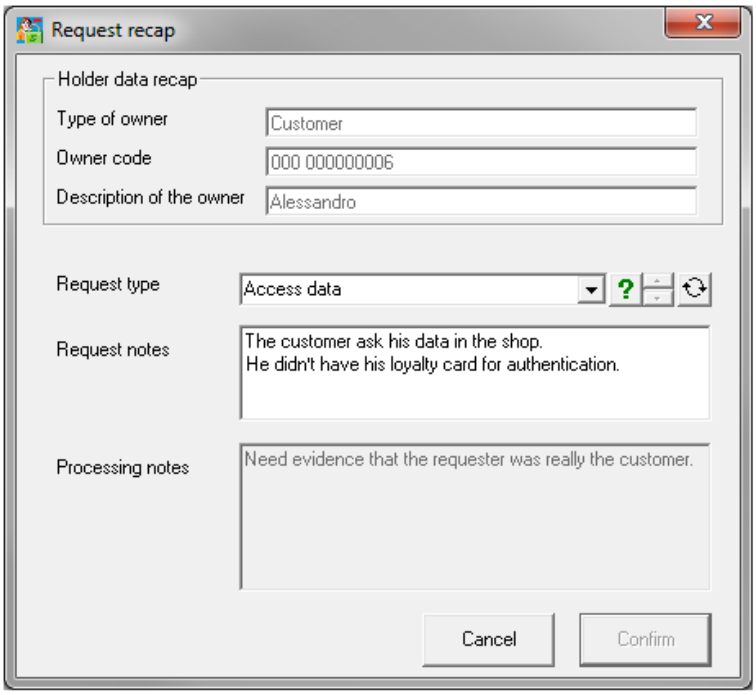
It is possible that the data privacy officer decides to reject a request. In this case, an additional optional note will prompted to the user.



It will be possible to process a rejected request.

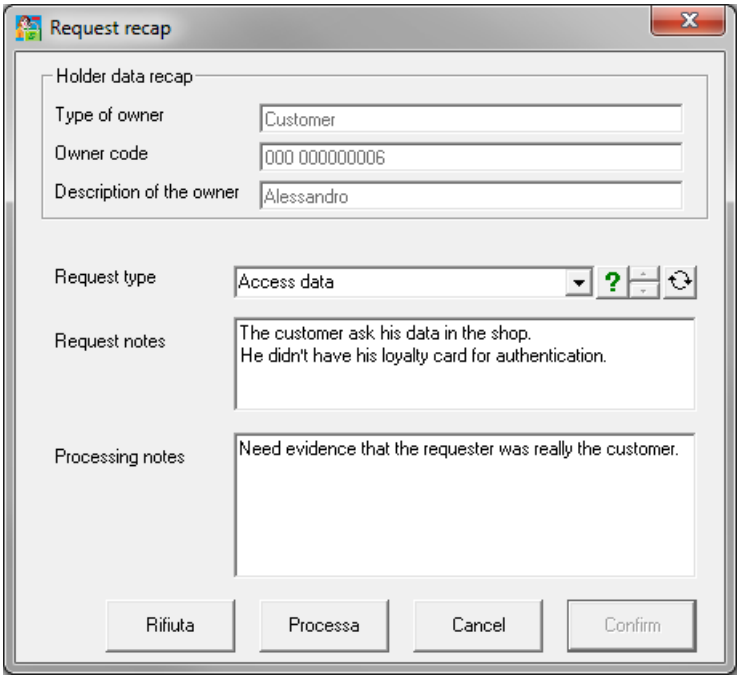
Property on the request

It is possible to open the request detail. No further actions are allowed. This should be used in the shop.



Check the request

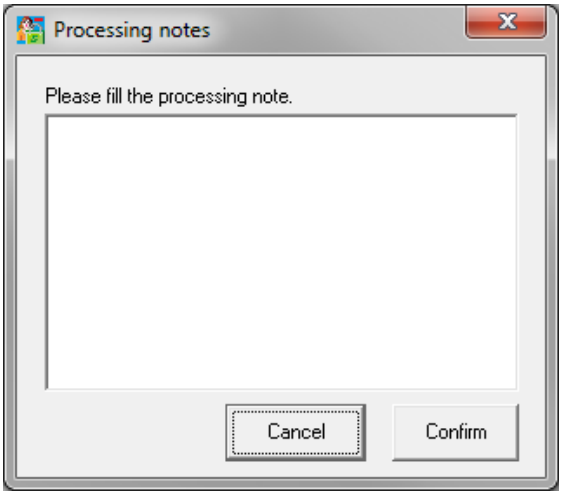
It is possible that the data privacy officer decides to see the request details.



In the check process, it is possible to process or reject the request.

Process the request

It is possible that the data privacy officer decides to process a request. In this case, an additional optional note will be prompted to the user.



During the process of a request:

If the request is to access data, the application will provide data to the API; in case of success, Stores2 will save the data in a JSON file on the hard disk in the defined path and will open (or bring to front) the folder.

If the request is to forget data, the application will ask the API to check if it is possible to forget requester's data.

- In case of a positive response:
  - The script for tokenization, if existing, will be executed;
  - The API will be requested to forget the requester;
  - The requester will be marked as forgotten and associated to the token (if provided).
- In case of a negative response, the user will be warned that it is not possible to forget the requester.

Forgotten requesters will not be found in researches.

Tokens and flag of the requestor forgotten will be exported to the ERP system from Host00.

If the prerequisites were not reached, the status of the request will be updated as "Blocked".

Configuration

Data privacy API

Definition

The purpose of the API is to allow third-party systems of the retailer to process rights requests on Stores2.

The same interface is used from Stores2 for processing requests.

The API is a WAR application (datapriv.war) and can be hosted on Oracle WebLogic or other hosts such as Tomcat or Jetty.

Tomcat

The following steps are used to host the API (it is not mandatory to follow this procedure):

- Download and unzip tomcat in the desired folder.

- Set up the TLS as in https://tomcat.apache.org/tomcat-8.0-doc/ssl-howto.html
- Add sqljdbc42.jar and sqljdbc41.jar in the Tomcat\lib folder.

Environment Variables

Define the following Windows environment variables (be sure to set the right path instead of the demonstration ones).

```
CATALINA_HOME = C:\Program Files (x86)\apache-tomcat-9.0.2
```

```
JAVA_JRE = C:\Program Files (x86)\Java\jre1.8.0_161
```

Users

Change tomcat-users.xml.

The API will be usable just for users with the DATAPRIV_ADMIN role, so it is mandatory to define it.

```
  <role rolename="tomcat"/>
  <role rolename="manager-gui"/>
  <role rolename="DATAPRIV_ADMIN"/>
  <user username="theDPadmin" password=" thepassword" roles="DATAPRIV_ADMIN"/>
  <user username="tomcat" password=" thepassword" roles="tomcat"/>
  <user username="themanager" password="thepassword" roles="manager-gui"/>
```

Then it is possible to deploy datapriv.war.

Wallet

Oracle wallet is an encrypted file where it is possible to store credentials in a secure way. The Datapriv API will get the credentials from the Oracle wallet defined.

Define the wallet location in catalina.properties file:

```
oracle.net.wallet_location = C:/Program Files (x86)/apache-tomcat-9.0.2/wallet
```

After the war has been deployed, it is required to create the wallet file for credential storing.

The following sample shows how to create the wallet file using oraclepki library included in the datapriv web application (obviously, paths shall be the correct ones).

Wallet Creation:

```
java -classpath "C:\Program Files (x86)\apache-tomcat-9.0.2\webapps\datapriv\WEB-
INF\lib\oraclepki-12.2.1.2.0.jar";"C:\Program Files (x86)\apache-tomcat-
9.0.2\webapps\datapriv\WEB-INF\lib\osdt_cert-3.1.0.jar";"C:\Program Files
(x86)\apache-tomcat-9.0.2\webapps\datapriv\WEB-INF\lib\osdt_core-3.1.0.jar"
oracle.security.pki.OracleSecretStoreTextUI -wrl "C:\Program Files (x86)\apache-
tomcat-9.0.2\wallet" -create
```

A complex password will be requested twice.

Save database connection into the wallet:

```
java -classpath "C:\Program Files (x86)\apache-tomcat-9.0.2\webapps\datapriv\WEB-
INF\lib\oraclepki-12.2.1.2.0.jar";"C:\Program Files (x86)\apache-tomcat-
9.0.2\webapps\datapriv\WEB-INF\lib\osdt_cert-3.1.0.jar";"C:\Program Files
(x86)\apache-tomcat-9.0.2\webapps\datapriv\WEB-INF\lib\osdt_core-3.1.0.jar"
oracle.security.pki.OracleSecretStoreTextUI -wrl "C:\Program Files (x86)\apache-
tomcat-9.0.2\wallet" -createCredential
jdbc:sqlserver://localhost:1433;databasename=DB_00000 sqluser password4sqluser
```

Config files

The API has four config files located in the datapriv\WEB-INF\classes\stores2\ folder:

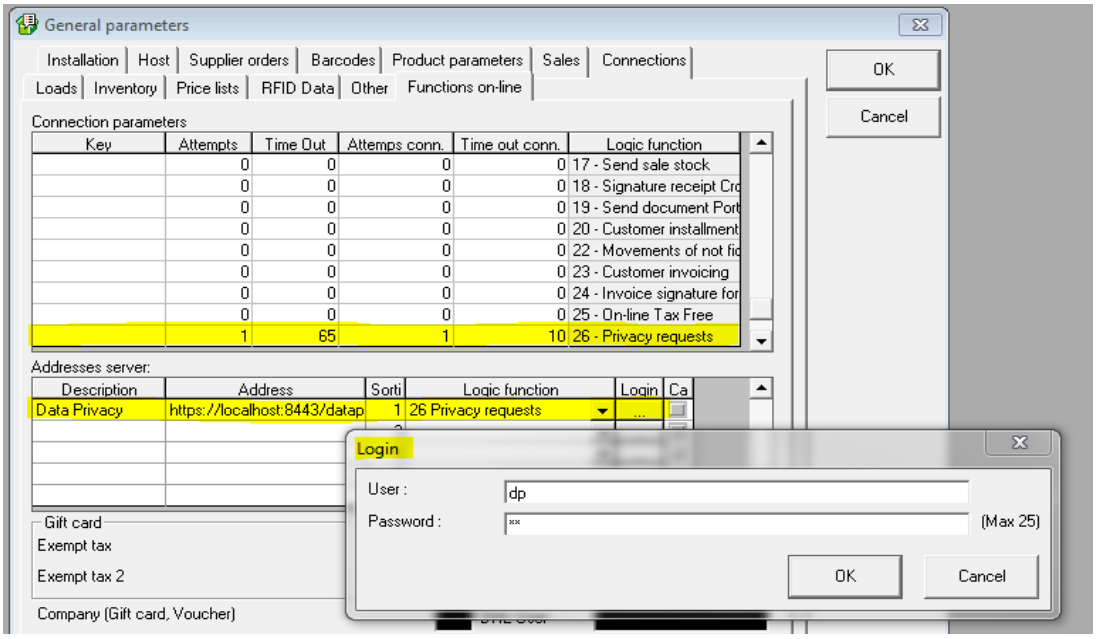
- DP-Global.xml: contains main configurations as the database connection.
  The “datasource-url” context-parm shall be equal to the credential stored in the wallet (`jdbc:sqlserver://localhost:1433;databasename=DB_00000` in the example in the paragraph above).
- DP-Get.xml: contains queries used to access data. It could be implemented from the retailer or system integrators; queryGroup types and input parameters are fixed.
- DP-ValidateForget.xml: contains queries used to validate data forget. It could be implemented from the retailer or system integrators; queryGroup types and input parameters are fixed. When any query returns a result set, this will indicate that the validation has failed and no further queries will be run.
- DP-Forget.xml: contains queries used to delete data. It could be implemented from the retailer or system integrators; queryGroup types and input parameters are fixed.

Stores2

API connection

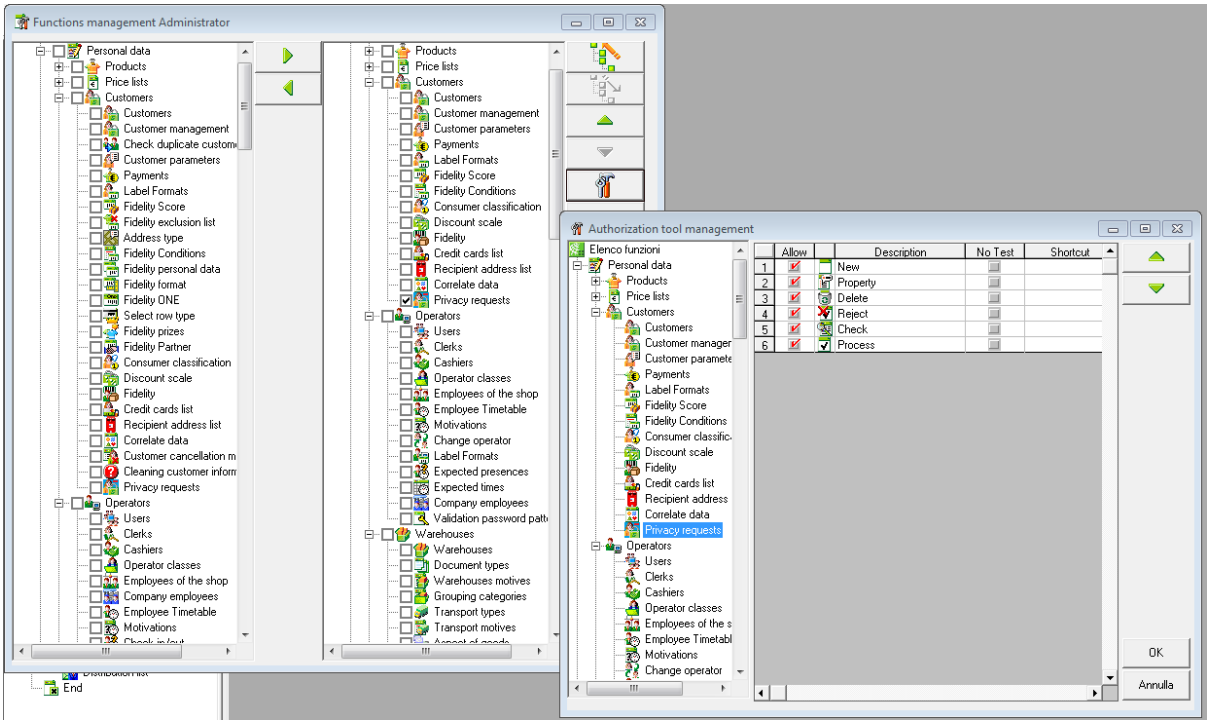
A new on-line function has been defined for the configuration of communication parameters with the API.

It is required to insert the API address (for example: https://localhost:8443/datapriv/rest/privatedata/) and the login of a user who has the DATAPRIV_ADMIN role in the webserver.



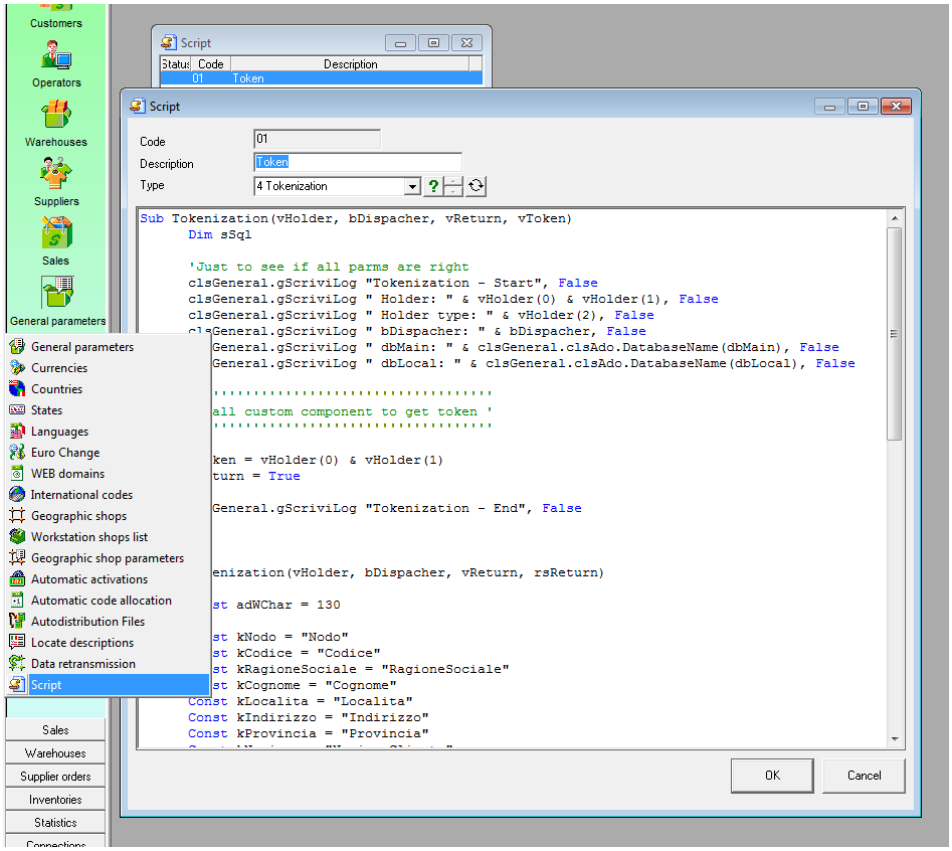
In user profiles, it is possible to make the new user interface visible and to define operations allowed as standard of Stores2.

Privacy requests



Support for tokenization

In general parameters, Script, a script for “Tokenization” could be defined.



The following test script can be used for tokenization and detokenization. Please consider that it is just for demonstration purposes.

```
Sub Tokenization(vHolder, bDispacher, vReturn, vToken)
	Dim sSql

	'Just to see if all parms are right
	clsGeneral.gScriviLog "Tokenization - Start", False
	clsGeneral.gScriviLog "  Holder: " & vHolder(0) & vHolder(1), False
	clsGeneral.gScriviLog "  Holder type: " & vHolder(2), False
	clsGeneral.gScriviLog "  bDispacher: " & bDispacher, False
	clsGeneral.gScriviLog "  dbMain: " & clsGeneral.clsAdo.DatabaseName(dbMain), False
	clsGeneral.gScriviLog "  dbLocal: " & clsGeneral.clsAdo.DatabaseName(dbLocal), False

	''''''''''''''''''''''''''''''''''''''''
	' Call custom component to get token '
	''''''''''''''''''''''''''''''''''''''''

	vToken = vHolder(0) & vHolder(1)
	vReturn = True

	clsGeneral.gScriviLog "Tokenization - End", False

End Sub

Sub Detokenization(vHolder, bDispacher, vReturn)

	Const adWChar = 130

	'Fields ad those expected in the Detokenization of ExportToSaftPT
	Const kNodo = "Nodo"
	Const kCodice = "Codice"
	Const kRagioneSociale = "RagioneSociale"
	Const kCognome = "Cognome"
	Const kLocalita = "Localita"
	Const kIndirizzo = "Indirizzo"
	Const kProvincia = "Provincia"
	Const kCAP = "CAP"
	Const kNazione = "NazioneCliente"
	Const kNazioneISO = "NazioneClienteISO"
	Const kCodiceFiscale = "CodiceFiscale"
	Const kPartitaIVA = "PartitaIVA"

	'Just to see if all parms are right
	clsGeneral.gScriviLog "Detokenization - Start", False
	clsGeneral.gScriviLog "  Holder: " & vHolder(0) & vHolder(1), False
	clsGeneral.gScriviLog "  Holder type: " & vHolder(2), False
	clsGeneral.gScriviLog "  Holder token: " & vHolder(3), False
	clsGeneral.gScriviLog "  bDispacher: " & bDispacher, False
	clsGeneral.gScriviLog "  dbMain: " & clsGeneral.clsAdo.DatabaseName(dbMain), False
	clsGeneral.gScriviLog "  dbLocal: " & clsGeneral.clsAdo.DatabaseName(dbLocal), False

	''''''''''''''''''''''''''''''''''''''''''''''''''''
	' Call custom component to get data from token '
	''''''''''''''''''''''''''''''''''''''''''''''''''''

	'Sample data just for test purposes
	rsCliente.Fields.Append kNodo, adWChar , 3
	rsCliente.Fields.Append kCodice, adWChar, 20
	rsCliente.Fields.Append kRagioneSociale, adWChar, 50
	rsCliente.Fields.Append kCognome, adWChar, 50
	rsCliente.Fields.Append kLocalita, adWChar, 50
	rsCliente.Fields.Append kIndirizzo, adWChar, 50
	rsCliente.Fields.Append kProvincia, adWChar, 50
	rsCliente.Fields.Append kCAP, adWChar, 50
	rsCliente.Fields.Append kNazione, adWChar, 50
	rsCliente.Fields.Append kNazioneISO, adWChar, 50
	rsCliente.Fields.Append kCodiceFiscale, adWChar, 50
	rsCliente.Fields.Append kPartitaIVA, adWChar, 50

	'open the recorset
	rsCliente.Open

	'add the new record --> in the real world, data are obtained from 3rd party system
	rsCliente.AddNew Array(kNodo, kCodice, kRagioneSociale, kCognome, kLocalita, _
			kIndirizzo, kProvincia, kCAP, kNazione, kNazioneISO, kCodiceFiscale, _
			kPartitaIVA), _
```

```
                    Array("000", "12345678", "Mario", "Rossi", "Milano", _
                    "Via Montenapoleone 369", "MI", "12345", "Italy", "IT", _
                    "RSSMRA1234567890", "0202020202020202" )

        vReturn = True

        clsGeneral.gScriviLog "Detokenization - End", False

End Sub
```

Restricted access to data

Minimization

Definition

In Stores2, it was already possible to minimize customers' data shown to users in different locations.

From version 1.39.4, the privacy engine has been boosted and allows the configuration also for the user profile (which overrides the location one).

Also, entries related to vendors, cashiers, associates, shop employees, company employees, and users are configurable by location or user profile.

It is also possible to define scripts with the scope of increasing the filters of data extracted.

Configuration

New configurable forms

In table "Elenchi", new configurable forms are added:

- 46 - Scheda Fornitori
- 47 - Scheda Cassieri
- 48 - Scheda Commessi
- 49 - Scheda Dipendenti Nodo
- 50 - Scheda Dipendenti Azienda
- 51 - Scheda Utenti
- 52 - Elenco Fornitori
- 53 - Elenco Cassieri
- 54 - Elenco Commessi
- 55 - Elenco Dipendenti Nodo
- 56 - Elenco Dipendenti Azienda
- 57 - Elenco Utenti
- 58 - Elenco Indirizzi
- 59 - Login Utenti

For forms above from 52 to 58 and for the customers list form, it is possible to define a custom script "on list", with the scope of adding additional filters in the data extraction. A simple sample could be the following one.

```
Sub Main(vReturn, vWhere)
      'Add new clause in the "WHERE" condition
      vWhere = "Clienti.Nodo <> '000' "
End Sub
```

In table "ElenchiFiltriCampi", the list of the components of each form is saved.

For each form of the list above, it is possible to hide or make mandatory some information. The position of fields cannot be changed.

Override for user profile

What was mentioned in the paragraph above could be also configured for the user profile, by inserting in the tables “ElenchiColonneOperatore” and “ElenchiFiltriCampiOperatore” records related of overridden components.

Security

Definition

By default, it will not be possible to create users, employees, and cashiers without a strong password (length equal or more than 8 digits, with at least 1 lowercase, 1 uppercase, 1 number, and 1 special character), because unprotected environments should not give access to personal data to someone not allowed to access it.

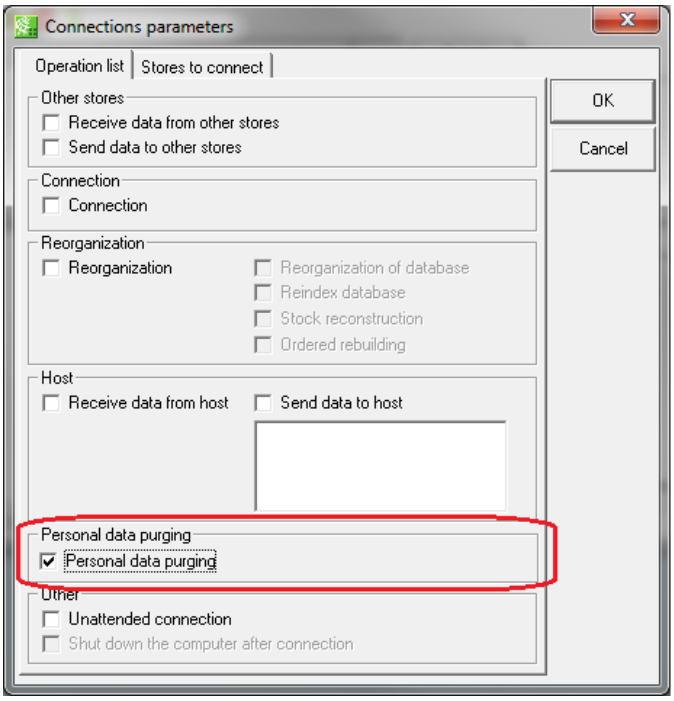
It is possible to override the base rule with custom rules in the Personal data\Operators\Validation password patterns.

Purging

Definition

The application will be able to clean data using custom rules.

If the location is enabled, in the connection properties of S2_Exchange, it will be possible to schedule a data purging after the connection.



Purging rules should be simply customized from the retailer.

The purging operation will be composed from three steps:

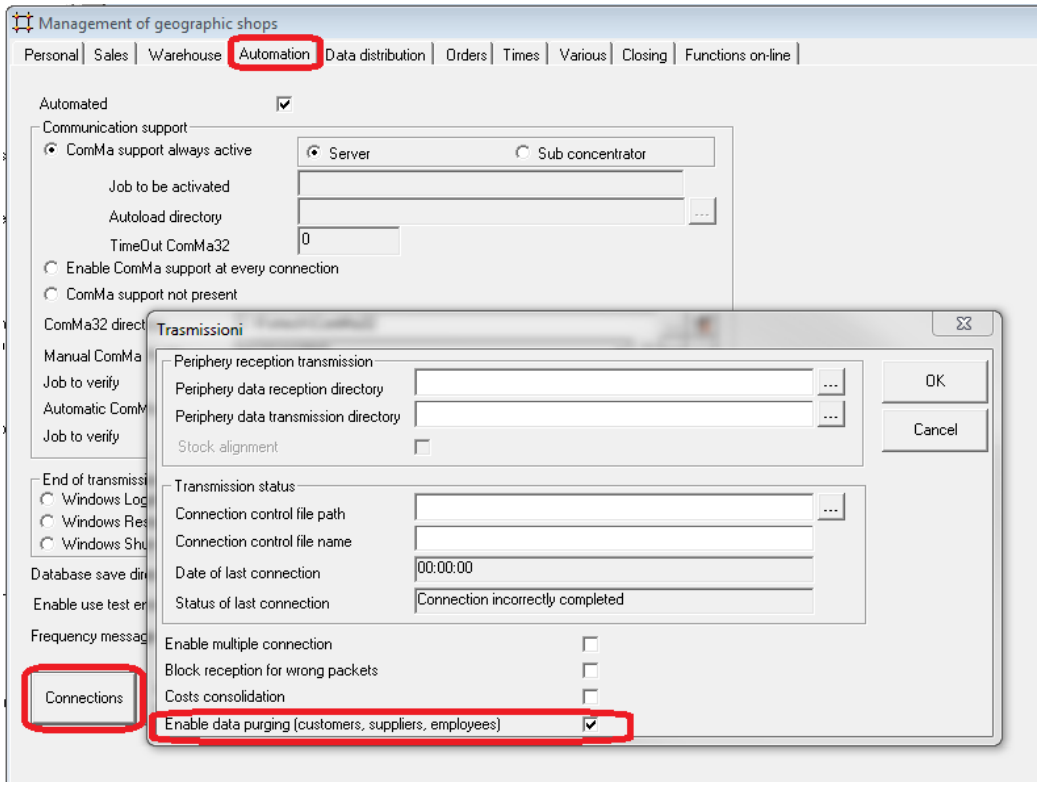
- Get data that can be purged with sp_canPurgeList.
- For each record:
  - Call the tokenization script (if defined);
  - Purge data with sp_dataPurging.

A log will be written in case of failure.

Configuration

Store

In the store configuration, the new function needs to be enabled.



Scheduled task

In the scheduled tasks of Stores2, an additional parameter is required to activate the data purge: /PURGEDATA

Query config

On the Stores2 database, two new stored procedures have been defined:

- sp_canPurgeList: The stored procedure returns a list of codes of customers, vendors, cashiers, associates, shop employees, company employees, and users that could be forgotten\anonymized.
  The procedure is composed from a structured list of queries in “UNION” and could be implemented by the retailer or the system integrator adding custom rules or modifying existing ones. The structure of the return table is fixed.
- sp_dataPurging: The stored procedure performs the anonymization of a record starting from the key:
  - @type: type of record to be updated (customer = 0; supplier = 1; cashier = 2; salesAssistant = 3; storeemp = 4; companyemp = 5; user = 6);
  - @store;
  - @code;
  - @userUPD: code of the user who is performing the update.

The return value will be a record with ret_code and ret_desc fields; in case of no errors, the result will be 0 – OK.

Limitation

The function is not supported in the Access database.

In case of database rebuilding from the HQ, the stored procedure on the rebuilt database will be the ones taken from the HQ database. So in case of a different implementation of stored procedures between HQ and shops, those stored procedure shall be modified after the rebuilding.

Data encryption

Definition

Stores2 exchanges data between HQ, stores, and ERP systems.

The data should be:

- Zipped access databases (Stores2 shop ←→ Stores2 HQ);
- Plan text files (Stores2 HQ ←→ ERP);
- Direct on the SQL Server database (Stores2 HQ ←→ ERP).

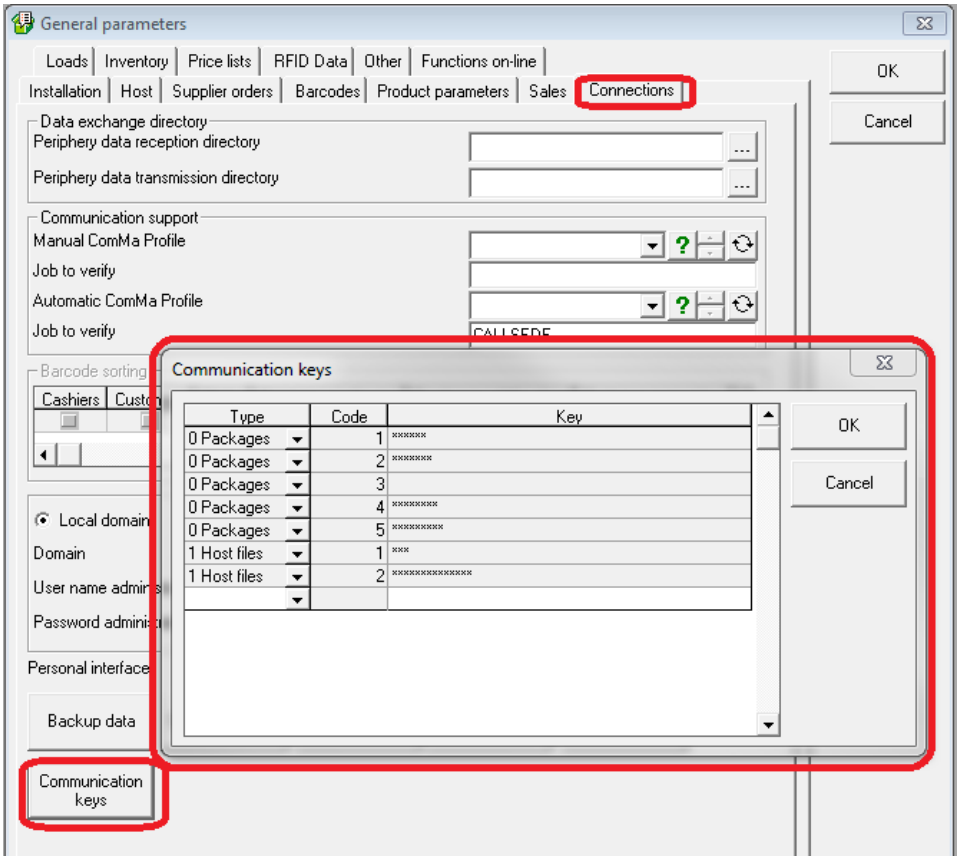
The access to the last one is already restricted using database privileges, while data in files should be potentially inaccessible from not allowed people.

There is now the possibility to define (only in HQ, as global settings) two different encryption keys for Stores2 packages and for host files.

If a password has been defined, it will be activated after a global automatic data extraction for shops. It should grant to have shops always be aware about the password of packages sent from HQ.

Configuration

In general settings\connections, a new button “Communication keys” has been created.



Log

Log files have been reviewed and in the case of personal data logged (few times customer name), personal data has been removed from the log files.

Consent

Definition

While inserting the personal data of a customer, it is required that the customer sign a consent report which contains information related to the usage of customer's data.

It is possible to set up the program for showing the consent report during the saving of customer data.

In any case, during the saving of customer data, Stores2 will ask the user if the customer has signed the consent report and will save the reply in the customer record.

Configuration

An example of the consent report can be seen in the Attachments tab. It is an example and needs to be implemented from the retailer.

The report needs to be associated as customer detail print.

General notes

Recommendations for retailers

- To review the list of the mandatory fields (customers, employees, and vendors);
- To review the list of the managed fields (customers, employees, and vendors);
- To set up appropriately the consent data and reports:
    - This includes that, for example, if there is a specific consent for the data analysis and the customer did not provide this access, it is up to the retailer to configure the reports in order to exclude those customers;
- To set up correctly the access of personal information for the user profile;
- To set up the database connection with the user and strong password and not with the windows user;
- To set up appropriately the datapriv app;
- To set up appropriately the data purging;
- To set up the packages encryption;
- To use encryption in Stores2WebService.
- To use strong password everywhere.