

Operational Ledger 3.0

User Guide

TOC

Overview	4
Installing Operational Ledger Add-On 3.1	5
Dependencies	5
Pre-Installation Checklist	6
Installation Steps	6
Installing Operational Ledger Add-On 3.0	9
Dependencies	9
Pre-Installation Checklist	10
Installation Steps	10
Installing Operational Ledger Add-On 3.0.1	13
Dependencies	13
Pre-Installation Checklist	14
Installation Steps	14
Installing Operational Ledger 3.1	17
Dependencies	17
Pre-Installation Checklist	18
Installation Steps	18
Operational Ledger Configuration	21
Legal Entity	21
Creating Legal Entities	21
Legal Entity Accounting Systems	23
Accounting System	23
Creating Accounting Systems	24
Accounting Chart	24

Accounting Scope	26
Creating Accounting Scopes	26
Transaction Type	27
Creating Transaction Types	27
Transaction Value Type	31
Transaction Item Accounting Configuration	33
Transaction Accounting Models	34
General Ledger Transactions	36
OL Operation Transaction	36
Creating Operation Transactions	36
Operation Transaction Values	38
Accounting Entry	39
Creating Accounting Entries	39
Operational Ledger SDK	42
1. FTOS_GL_SetValues_TransactionOperation	42
2. getFinalValue	44
3. getAttributeValue	45
4. isAttributeInvariantDate	45
5. contains	46
6. validateFormulaAttribute	47
7. existsAttributeInEntity	48
8. existsAttributeInTableName	48
9. existsAttributeInSourceEntity	49
10. existsOperationTransactionForRecordId	49
11. existsOperationTransactionValue	50
Glossary	52

Overview

The **FintechOS Operational Ledger** solution is a banking module that enables companies to effectively perform financial transactions and to gather the specific accounting information needed for ledger reports and other financial statements. The data is captured, processed, and displayed on the **FintechOS Portal**, allowing customers to gain insight into overall or specific financial contexts.

Operational Ledger comes with a transaction accounting model that reads information from the transaction and displays it in the right fields. It allows for a registration of transactions, and stores and organizes financial information which is then used in the company's statements, for the creation of a balance or other operations. The double-entry system keeps the data organized for further reports and documents. This solution makes it easy to search for journal entries. It logs along with a company's financial transactions also specific details that enable the system to build ledger entries allowing you to aggregate financial data in a single source of truth for your analysis, reports, or financial statements.

Installing Operational Ledger Add-On 3.1

HINT

This page contains the installation steps for Operational Ledger Add-On **3.1**.
To install the **3.0.1** version, follow the instructions on [this](#) page.
To install the **3.0** version, follow the instructions on [this](#) page.

The Operational Ledger Add-On **v3.1** comes as an add-on to the **Core Banking v3.1** module. This is a process of running two `install_SysPack.bat` files on your environment. Follow the steps described below to perform an automatic installation of the Operational Ledger Add-On.

IMPORTANT!

You must run the script on the machine where FintechOS Studio is installed.
Make sure you have access rights to Studio's database.

Dependencies

In order to install Operational Ledger Add-On 3.1, first you need to install the following:

- **FintechOS Studio** minimum version **v21.2.2.2**
- **Standard SysPack** minimum version **v21.2.2000**
- **Single Customer View (Retail & SME) v3.1**
- **FTOS.Foundation - Project**

- **Banking Product Factory** (Project or Standard Pack) **v3.1**.
- **Core Banking v3.1**

Pre-Installation Checklist

The SysPack has a unique constrain on the FTOS_CB_SystemParameter entity.

If you have already moved data using the **Configuration Data Deployment Package** menu, then you probably have already configured some unique constraints.

Before running the script, make sure you:

1. Disable the constraints that you have created on your environment, allowing the system to create the new one after Operational Ledger 3.1 is imported.
2. Use the new **Configuration Data Definitions** imported with the Operational Ledger Add-On 3.1 file when you export the data.

Installation Steps

1. Download the **Operational Ledger AddOn 3.1** archive file and unzip it.

Operational Ledger AddOn v3.1

Name	Date modified
01.Operation Ledger AddOn v3.1	2/11/2022 5:14 PM
02.Operational Ledger Add On Security R...	2/11/2022 5:14 PM

IMPORTANT!

Each zip file within the **Operational Ledger AddOn v3.1.zip** must be extracted and installed separately, in the given order!

If you decide not to use the default security roles that come with the

package, simply skip the Core Banking Security Roles zip file.

Operational Ledger AddOn v3.1 > 02.Operational Ledger Add On Security Roles v3.1

Name	Size	Date modified
GL Core Banking Security Roles		2/11/2022 5:14 PM
General_Ledger_Security_Roles.zip	33 KB	2/11/2022 12:19 PM

2. Locate the *Operational Ledger AddOn.zip* archive file.

Operational Ledger AddOn v3.1 > 01.Operation Ledger AddOn v3.1

Name	Size	Date modified
GL Core Banking AddOn		2/11/2022 5:14 PM
Operational Ledger AddOn.zip	92 KB	2/11/2022 3:01 PM
README_install_dependencies.txt	1 KB	1/25/2022 3:03 PM

3. Unzip the **Operational Ledger AddOn.zip** archive file.
4. Navigate to the location where you have unzipped the **Operational Ledger Add On** and add the first `install_SysPack.bat` file.
5. Then, locate the **GL Core Banking AddOn** and open it.

Operational Ledger AddOn v3.1 >

Name	Size	Date modified
FtosSysPkgDeployer		1/25/2022 2:52 PM
GL Core Banking AddOn		1/25/2022 2:38 PM
Operational Ledger AddOn.zip	92 KB	1/25/2022 2:38 PM
README_install_dependencies.txt	1 KB	1/25/2022 3:03 PM

6. Add the second `install_SysPack.bat` file to the **CBGLAddOn_SystemParameter_v1000.xml** file.

The `install_SysPack.bat` file allows you to import the data model:

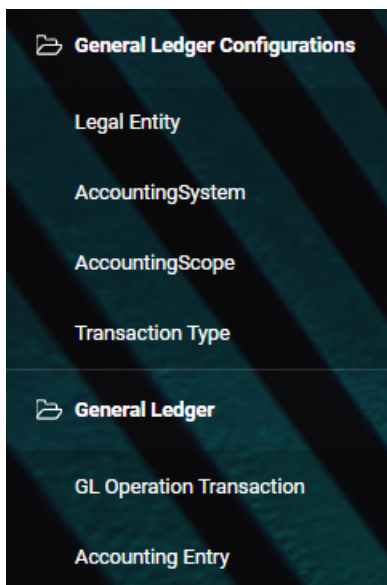
install_SysPack.bat syntax for **Data Model** import

```
FtosSysPkgDeployer.exe -i -s "<StudioLink>" -u
<AdminStudioUser> -p <user_password> -z <DataBaseServer> -v
<DB_user> -k <DB_user_password> -d "<TheNameOfTheDataBase>" -
r "<syspack_path>\*.zip"
```

NOTE

There is no need to unzip the **01.CBGLAddon_SystemParameter_v1000.zip**.

The script start running in your Windows console. Wait for it to finish. If the parameter values were correct, the FintechOS Portal has two new menus, visible after a refresh, the **General Ledger Configurations** and the **General Ledger** menus:



NOTE

The syntax presented here is for information purposes only. Please run the actual `install_SysPack.bat` file.

IMPORTANT!

If you're using **SQL Server Integrated Authentication**, make sure that the Windows

user used for running the script has access to the FTOS database, with read/ write rights. Run the command without the SQL username/ password parameters. If you're using **SQL Server Build In Authentication**, make sure that the SQL Server user has read/ write access to the FTOS database. Run the command with the SQL username/ password parameters.

Installing Operational Ledger Add-On 3.0

HINT

This page contains the installation steps for Operational Ledger Add-On 3.0. To install the 3.0.1 version, follow the instructions on [this](#) page.

The Operational Ledger **v3.0** Add-On comes as an add-on to the **Core Banking v3.0** module. This is a process of running two `install_SysPack.bat` files on your environment. Follow the steps described below to perform an automatic installation of the Operational Ledger.

IMPORTANT!

You must run the script on the machine where FintechOS Studio is installed. Make sure you have access rights to Studio's database.

Dependencies

In order to install Operational Ledger Add-On 3.0, first you need to install the following:

- **FintechOS Studio** minimum version **v21.1.6.1**
- **Standard SysPack** minimum version **v21.1.1005**
- **Single Customer View (Retail & SME) v3.1**

- **FTOS.Foundation - Project**
- **Banking Product Factory** (Project or Standard Pack) **v21.1.8001.**
- **Core Banking v21.1.1001**

Pre-Installation Checklist

The SysPack has a unique constrain on the FTOS_CB_SystemParameter entity.


If you have already moved data using the **Configuration Data Deployment Package** menu, then you probably have already configured some unique constraints.

Before running the script, make sure you:

1. Disable the constraints that you have created on your environment, allowing the system to create the new one after Operational Ledger 3.0 is imported.
2. Use the new **Configuration Data Definitions** imported with the Operational Ledger 3.0 file when you export the data.

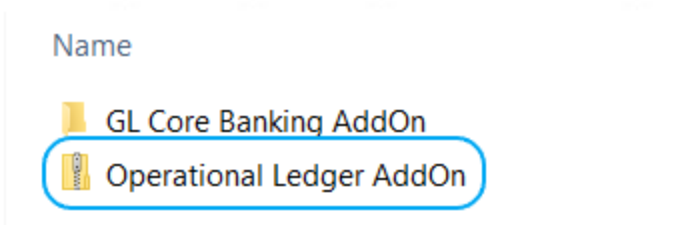
Installation Steps

1. Download the **Operational Ledger Add On 3.0** archive file and unzip it.

 OperationalLedgerAddOn_21.1.1001




2. Locate the *Operational Ledger AddOn* archive file and unzip it.

› OperationalLedgerAddOn_21.1.1001 ›



3. Unzip the **Operational Ledger AddOn.zip** archive file.

4. Navigate to the location where you have unzipped the **Operational Ledger Add On** and add the first `install_SysPack.bat` file.
5. Then, locate the **GL Core Banking AddOn** and open it.

OperationalLedgerAddOn_21.1.1001 > Operational Ledger AddOn >	
Name	Type
 GL Core Banking AddOn	File folder
 GL Core Banking AddOn	XML Document
 ProjectSummary	XML Document

6. Add the second `install_SysPack.bat` file to the **CBGLAddOn_SystemParameter_v1000.xml** file.

The `install_SysPack.bat` file allows you to import the data model:

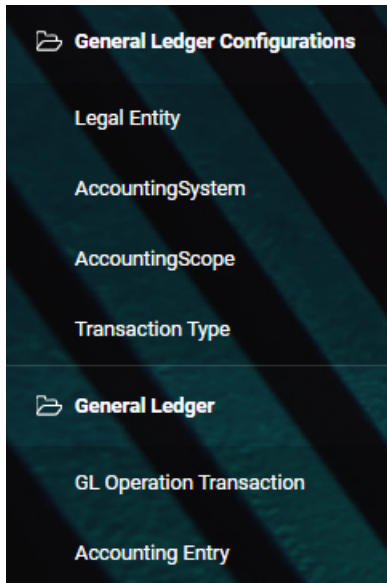
install_SysPack.bat syntax for **Data Model** import

```
FtosSysPkgDeployer.exe -i -s "<StudioLink>" -u
<AdminStudioUser> -p <user_password> -z <DataBaseServer> -v
<DB_user> -k <DB_user_password> -d "<TheNameOfTheDataBase>" -
r "<syspack_path>\*.zip"
```

NOTE

There is no need to unzip the **01.CBGLAddOn_SystemParameter_v1000.zip**.

The script start running in your Windows console. Wait for it to finish. If the parameter values were correct, the FintechOS Portal has two new menus, visible after a refresh, the **General Ledger Configurations** and the **General Ledger** menus:



NOTE

The syntax presented here is for information purposes only. Please run the actual `install_SysPack.bat` file.

IMPORTANT!

If you're using **SQL Server Integrated Authentication**, make sure that the Windows user used for running the script has access to the FTOS database, with read/ write rights. Run the command without the SQL username/ password parameters.

If you're using **SQL Server Build In Authentication**, make sure that the SQL Server user has read/ write access to the FTOS database. Run the command with the SQL username/ password parameters.

Installing Operational Ledger Add-On 3.0.1

HINT

This page contains the installation steps for Operational Ledger Add-On **3.0.1**. To install the 3.0 version, follow the instructions on [this](#) page.

The Operational Ledger Add-On **v3.0.1** comes as an add-on to the **Core Banking v3.0.2** module. This is a process of running two `install_SysPack.bat` files on your environment. Follow the steps described below to perform an automatic installation of the Operational Ledger.

IMPORTANT!

You must run the script on the machine where FintechOS Studio is installed. Make sure you have access rights to Studio's database.

Dependencies

In order to install Operational Ledger Add-On 3.0.1, first you need to install the following:

- **FintechOS Studio** minimum version **v21.2.2**
- **Standard SysPack** minimum version **v21.1.2000**
- **Single Customer View (Retail & SME) v3.1**
- **FTOS.Foundation - Project**
- **Banking Product Factory (Project or Standard Pack) v3.0.2.**
- **Core Banking v3.0.2**

Pre-Installation Checklist

The SysPack has a unique constrain on the FTOS_CB_SystemParameter entity.

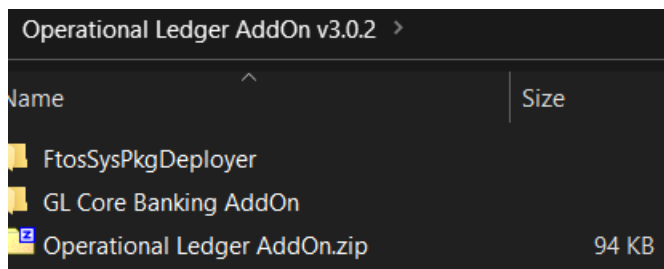
If you have already moved data using the **Configuration Data Deployment Package** menu, then you probably have already configured some unique constraints.

Before running the script, make sure you:

1. Disable the constraints that you have created on your environment, allowing the system to create the new one after Operational Ledger 3.0.1 is imported.
2. Use the new **Configuration Data Definitions** imported with the Operational Ledger 3.0.1 file when you export the data.

Installation Steps

1. Download the **Operational Ledger Add On 3.0.1** archive file and unzip it.
2. Locate the *Operational Ledger AddOn* archive file and unzip it.



3. Unzip the **Operational Ledger AddOn.zip** archive file.
4. Navigate to the location where you have unzipped the **Operational Ledger Add On** and add the first `install_SysPack.bat` file.
5. Then, locate the **GL Core Banking AddOn** and open it.
6. Add the second `install_SysPack.bat` file to the **CBGLAddOn_SystemParameter_v1000.xml** file.

The `install_SysPack.bat` file allows you to import the data model:

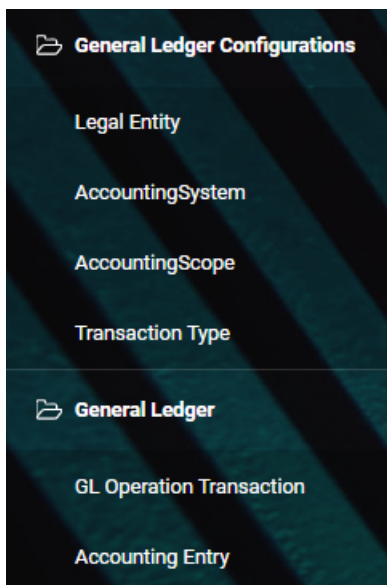
install_SysPack.bat syntax for **Data Model** import

```
FtosSysPkgDeployer.exe -i -s "<StudioLink>" -u
<AdminStudioUser> -p <user_password> -z <DataBaseServer> -v
<DB_user> -k <DB_user_password> -d "<TheNameOfTheDataBase>" -
r "<syspack_path>\*.zip"
```

NOTE

There is no need to unzip the **01.CBGLAddon_SystemParameter_v1000.zip**.

The script start running in your Windows console. Wait for it to finish. If the parameter values were correct, the FintechOS Portal has two new menus, visible after a refresh, the **General Ledger Configurations** and the **General Ledger** menus:



NOTE

The syntax presented here is for information purposes only. Please run the actual `install_SysPack.bat` file.

IMPORTANT!

If you're using **SQL Server Integrated Authentication**, make sure that the Windows

user used for running the script has access to the FTOS database, with read/ write rights. Run the command without the SQL username/ password parameters.

If you're using **SQL Server Build In Authentication**, make sure that the SQL Server user has read/ write access to the FTOS database. Run the command with the SQL username/ password parameters.

Installing Operational Ledger

3.1

HINT

This page contains the installation steps for Operational Ledger **3.1**.
To install the Operation Ledger **Add-On 3.1** version, follow the instructions on [this](#) page.

The Operational Ledger **v3.1** comes as a stand-alone product that can be installed to be used with other Lighthouse or Northstar products. This is a process of running two `install_SysPack.bat` files on your environment. Follow the steps described below to perform an automatic installation of the Operational Ledger.

IMPORTANT!

You must run the script on the machine where FintechOS Studio is installed.
Make sure you have access rights to Studio's database.

Dependencies

In order to install Operational Ledger 3.1, first you need to install the following:

- **FintechOS Studio** minimum version **v21.2.2.2**
- **Standard SysPack** minimum version **v21.2.2000**
- **Banking Product Factory** (Project or Standard Pack) **v3.1**.

Pre-Installation Checklist

The SysPack has a unique constrain on the FTOS_CB_SystemParameter entity.

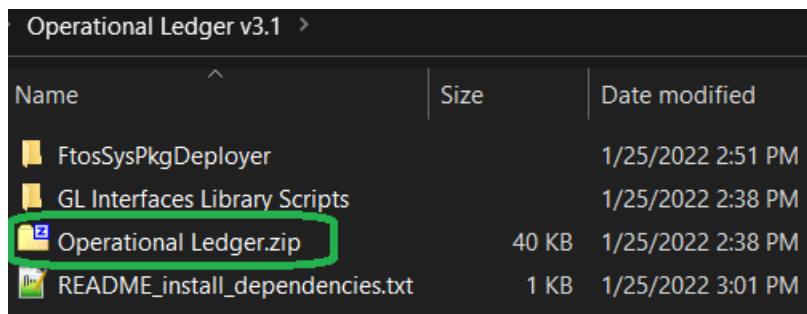
If you have already moved data using the **Configuration Data Deployment Package** menu, then you probably have already configured some unique constraints.

Before running the script, make sure you:

1. Disable the constraints that you have created on your environment, allowing the system to create the new one after Operational Ledger 3.1 is imported.
2. Use the new **Configuration Data Definitions** imported with the Operational Ledger 3.1 file when you export the data.

Installation Steps

1. Download the **Operational Ledger 3.1** archive file and unzip it.
2. Locate the *Operational Ledger.zip* archive file.



3. Unzip the **Operational Ledger.zip** archive file.
4. Navigate to the location where you have unzipped **Operational Ledger** and add the first `install_SysPack.bat` file.
5. Then, locate the **GL Interfaces Library Scripts** and open it.

Operational Ledger v3.1 >		
Name	Size	Date modified
FtosSysPkgDeployer		1/25/2022 2:51 PM
GL Interfaces Library Scripts		1/25/2022 2:38 PM
Operational Ledger.zip	40 KB	1/25/2022 2:38 PM
README_install_dependencies.txt	1 KB	1/25/2022 3:01 PM

6. Add the second `install_SysPack.bat` file to the **CBGLAddon_SystemParameter_v1000.xml** file.

The `install_SysPack.bat` file allows you to import the data model:

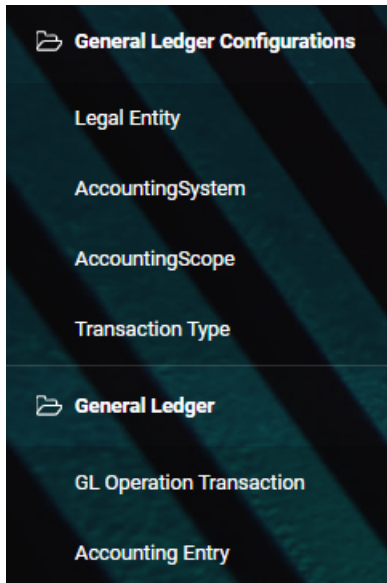
install_SysPack.bat syntax for **Data Model** import

```
FtosSysPkgDeployer.exe -i -s "<StudioLink>" -u
<AdminStudioUser> -p <user_password> -z <DataBaseServer> -v
<DB_user> -k <DB_user_password> -d "<TheNameOfTheDataBase>" -
r "<syspack_path>\*.zip"
```

NOTE

There is no need to unzip the **01.CBGLAddon_SystemParameter_v1000.zip**.

The script start running in your Windows console. Wait for it to finish. If the parameter values were correct, the FintechOS Portal has two new menus, visible after a refresh, the **General Ledger Configurations** and the **General Ledger** menus:



NOTE

The syntax presented here is for information purposes only. Please run the actual `install_SysPack.bat` file.

IMPORTANT!

If you're using **SQL Server Integrated Authentication**, make sure that the Windows user used for running the script has access to the FTOS database, with read/ write rights. Run the command without the SQL username/ password parameters.

If you're using **SQL Server Build In Authentication**, make sure that the SQL Server user has read/ write access to the FTOS database. Run the command with the SQL username/ password parameters.

Operational Ledger Configuration

The **Operational Ledger** holds legal entities with their respective accounting systems and their transactions. The inserted transaction information is organized by types into accounts, inside the chosen account chart. This way, it helps keep track of revenues, expenses, owners' equity, assets, liabilities.

In order to set up a **Operational Ledger**, a number of configurations must be made. These configurations are done from the **FintechOS Portal**. The configurations are detailed in the pages below:

- [Legal Entity](#)
- [Accounting System](#)
- [Accounting Scope](#)
- [Transaction Type](#)

Legal Entity

The legal entity is the company for which the user operates the accounting. A legal entity can have multiple accounting systems.

Creating Legal Entities

1. Log into the **FintechOS Portal**.

2. In the main menu, expand **General Ledger Configurations** and select **Legal Entity**¹. The **Legal Entities List** page opens.
3. At the top-right corner of the page, click the **Insert** button to create a new legal entity.
4. Fill in the following fields:

ADD LEGAL ENTITY

LEGAL ENTITY

Name

Status

Valid From Date

Valid Until Date

Field	Required	Type	Description
Name	No	Text	The name of the entity.
Status	No	Lookup	The status of the entity. Select from the following statuses or create a new one: <ul style="list-style-type: none"> • Draft • Active.
Valid From Date	No	Date	The date from which the entity is valid.
Valid Until Date	No	Date	The date from which the entity is no longer valid.

5. Click **Save and reload**. The **Legal Entity Accounting Systems** section is displayed.

¹An association, corporation, partnership, proprietorship, trust, or individual that has legal standing in the eyes of law. A legal entity has legal capacity to enter into agreements or contracts, assume obligations, incur and pay debts, sue and be sued in its own right, and to be held responsible for its actions.

Legal Entity Accounting Systems

This section stores data about the accounting systems used by the legal entity.

1. To add a new legal entity accounting system, click the **Insert** button under the **Legal Entity Accounting Systems** section. The **Add Legal Entity Accounting Systems** page opens.

2. Fill in the following fields:

ADD LEGAL ENTITY ACCOUNTING SYSTEM

LEGAL ENTITY ACCOUNTING SYSTEM

Legal Entity

DocTest
↓
✎

Accounting Standards

Test
↓
✎

Chart Of Accounts

0002
↓
✎

Field	Required	Type	Description
Legal Entity	No	Text	The name of the legal entity.
Accounting Standards	No	Text	The accounting standards used by the system.
Chart Of Accounts	No	Lookup	The accounting systems used.

2. Click **Save and close**.

HINT

Users can add, delete, or export Legal Entity Accounting Systems tables.

Accounting System

An **accounting system**¹ represents the system used for collecting and organizing the financial data. For each legal entity there is a corresponding system. An entity can have more accounting systems.

¹The system used to manage the income, expenses, and other financial activities of a business.

Creating Accounting Systems

1. Log into the **FintechOS Portal**.
2. In the main menu, expand **General Ledger Configurations** and select **Accounting System**. The **Accounting Systems List** page opens.
3. At the top-right corner of the page, click the **Insert** button to create an accounting system.
4. Fill in the following fields:

ADD ACCOUNTING SYSTEM

ACCOUNTING SYSTEM

Code

0000

Name

DocTest

Accounting Reference Currency

EUR

Field	Required	Type	Description
Code	No	Text	It is the code of the accounting system.
Name	No	Text	It is the official name of the accounting system.
Accounting Reference Currency	No	Option set	It contains every currency possible with its code and symbol. Choose the currency for this specific system.

5. Click **Save and reload**. The **Accounting Chart** section is displayed.

Accounting Chart

This section stores data regarding the accounts.

1. To add a new transaction item accounting configuration, click the **Insert** button under the **Accounting Chart** section. The **Accounting Chart** page opens.

2. Fill in the following fields:

AccountingChart

Accounting System

DocTest

reportingCode

doctest

Reporting Group Name

DocTest

Account Code

doctest

Name

Doc Test

Balance Type

BalanceSheet

Type

Liabilities

Field	Required	Type	Description
Accounting System	No	Lookup	The accounting system of the chart ¹ .
Account code	No	Text	The code of the account.
reportingCode	No	Text	The code of the reporting.
Type	No	Option set	<p>The type of the account. The following options are available:</p> <ul style="list-style-type: none"> • [none] • Assets • Expenses • Revenues • Liabilities • Others.
Balance Type	No	Option set	<p>The balance type of the account. The following options are available:</p> <ul style="list-style-type: none"> • BalanceSheet • P&L • OffBalanceSheet
Reporting Group Name	No	Text	The name of the reporting group.
Name	No	Text	The name of the account.

¹An index of all the financial accounts in the general ledger of a company.

2. Click **Save and close**.

HINT

Users can add, delete, or export accounting charts.

Accounting Scope

This menu introduces the scope for accounting a financial transaction.

Creating Accounting Scopes

1. Log into the **FintechOS Portal**.
2. In the main menu, expand the **General Ledger Configurations** and select **Accounting Scope**. The **Accounting Scopes List** page opens.
3. At the top-right corner of the page, click the **Insert** button to add a new accounting scope.
4. Fill in the following field:

ADD ACCOUNTING SCOPE**ACCOUNTING SCOPE**

Name

Field	Required	Type	Description
Name	No	Text	The name of the accounting scope.

5. Click **Save and reload**.

From here, users can export the transactions displayed in the **Banking Product GL Accounts** and the **Transaction Item Accounting Configuration** sections.

NOTE

The **Transaction Item Accounting Configuration** section displays the configuration of each transaction. For more details on creating such transaction configurations, see the [Transaction Item Accounting Configuration](#) chapter.

Transaction Type

The **Transaction Type** menu holds the accounting model and the item configuration. It is used for inserting the financial data transactions.

For using transaction types in conjunction with Core Banking, see [Transaction Types Used in Core Banking](#).

HINT

Users can insert, delete, or export **Transaction Types** tables.

Creating Transaction Types

1. Log into the **FintechOS Portal**.
2. Expand the **General Ledger Configurations** menu and select **Transaction Type**. The **Transaction Operation Types List** page opens.
3. At the top-right corner of the page, click the **Insert** button to create a new transaction type.
4. Fill in the following fields:

EDIT TRANSACTION TYPE

TRANSACTION TYPE

Name

Withdraw

Transaction Code

W

Generates Accounting Entry

☒

Generate New Contract Version

☐

Real Time Process

☐

PURGE

To Be Purged

☒

Master Purge Entity

FTOS_CB_ContractEvent

Purge Number of Days

0

Is Automatic Transaction

☐

Is System Transaction

☐

Transaction Operation Type

PaymentOut

SourceEntityId

FTOS_CB_ContractEvent

Commission Type

Field	Required	Type	Description
Name	Yes	Text	The name of the transaction.
Is Automatic Transaction	No	Bool	<p>If true, then the transaction is automatic.</p> <div> <p>IMPORTANT!</p> <p>If a transaction type is marked as an automatic transaction (Is Automatic Transaction = True), then that transaction type cannot be selected in the Events page when closing contract events.</p> </div>
Is System Transaction	No	Bool	If true, then this is a system transaction. It is used when there's a need of a transaction for the sole purpose of generating accounting entries (accruals, provisions).
Transaction Code	Yes	Text	The code of the transaction.

Field	Required	Type	Description
Transaction Operation Type	Yes	Lookup	<p>The type of the transaction. The following options are available:</p> <ul style="list-style-type: none"> • Disbursement • Payment In • Payment Out • Recover Debt • Repayment Contract -
Generates Accounting Entry	No	Bool	If true, it generates recordings in the FTOS_GL_AccountingEntryentity.
realTimeProcess	No	Bool	If true, it allows real-time processing for transactions made on bank accounts. For more information, see the Bank Account Transaction Queue page.
To Be Purged	No	Bool	If true, it allows the possibility to purge or archive certain records in draft status.
Master Purge Entity	Yes	Lookup	<p>The master purge entity under which the draft records are.</p> <div> <p>NOTE</p> <p>This field is displayed only when To Be Purged = True.</p> </div>

Field	Required	Type	Description
Purge Number of Days	Yes	Whole Number	<p>The default number of calendar days that a record can be kept in Draft status before it is purged. For additional information, see the Core Banking System Parameters page.</p> <div> NOTE This field is displayed only when To Be Purged = True. </div>
Source Entity ID	Yes	Lookup	<p>The source entity from where the values are taken when generating accounting entries.</p> <div> NOTE This field is displayed only when Generates Accounting Entry = True. </div>
Generate New Contract Version	No	Bool	If true, a new contract version is generated.
Commission Type	No	Lookup	The commission type applicable to transactions of this type.

5. Click **Save and reload**.

IMPORTANT!

If a transaction type is marked to automatically generate accounting entries (Generates Accounting Entry = True), then the following sections are

displayed: **Transaction Value Type**, **Transaction Item Accounting Configuration**, and **Transaction Accounting Models**.

Transaction Value Type

Transaction value types are defined as header items or detail items. Header items are the general details of a transaction (for example date, customer, currency, and so on). The detail items are grouped into numeric or text information.

In this section, users can create and determine the values calculated for each transaction. The additional data from here is used in the **Transaction Accounting Models** section.

- 1. To add a new transaction value type, click the **Insert** button under the **Transaction Value Type** section. The **Add Transaction Value Type** page opens.
- 2. Fill in the following fields:

ADD TRANSACTION VALUE TYPE

TRANSACTION VALUE TYPE

Value Type Name

DocTest

Transaction Type

Disbursement

Is Header

☐

Type

Numeric

Value Type Attribute

eventValue

Formula

eventValue * 2

Field	Required	Type	Description
Value Type Name	No	Text	The name of the value type.
Transaction Type	No	Lookup	The transaction value type.
Type	No	Option Set	The type of the transaction. The following options are available: <ul style="list-style-type: none">[none]NumericText

Field	Required	Type	Description
Value Type Attribute	No	Lookup	The value of a specific attribute from the source entity. It is a list of all the attributes defined in the SourceEntityId field from the FTOS_GL_TransactionType entity.
Is Header	No	Bool	When selected, it defines the header items of the transaction.
Formula	No	Text	Supports only basic math operations: addition (+), subtraction (-), multiplication (*), and division (/). Input a specific formula based on the Value Type Attribute chosen.

3. Click **Save and close**.

When a transaction value type is marked as a header item, (Is Header = True), the transaction values are set into the attribute values of the FTOS_GL_OperationTransaction entity. If the **Value Type Name** field is not an attribute of that entity, then the following error is displayed:

Name should be an attribute name from header table FTOS_GL_OperationTransaction.

A json with default values is sent when using the function for setting the operation transaction values. The json has the following form:

```
[
  {
    attributeName: 'DescriptionText',
    value: 'Disburse 1500'
  },
  {
    attributeName: 'ProvisionAmount',
```

```

    value: '15.00'
  }
]

```

The `json` checks if there is any default value for the `attributeName`, from the **Value Type Name** field. If no values are returned, the **Formula** field is checked. When neither field returned any values, the source entity of the attribute from the **Value Type Attribute** field is checked.

Transaction Item Accounting Configuration

The **Transaction Item Accounting Configuration** section holds the configuration of each transaction. It helps define an account from the Accounting Chart.

This section holds the configuration for an item with the respective chart account. It represents the listing of the names of the accounts for the company inserted in the **Legal Entity** menu.

1. To add a new transaction item accounting configuration, click the **Insert** button under the **Transaction Item Accounting Configuration** section. The **Add Transaction Item Accounting Config** page opens.
2. Fill in the following fields:

ADD TRANSACTION ITEM ACCOUNTING CONFIG

TRANSACTION ITEM ACCOUNTING CONFIG

Accounting System

DocTest

Accounting Scope

DocTest

Chart Account

Doc Test

Operational Item

Repayment Fee

Currency

EUR

Take From Product

Field	Required	Type	Description
Accounting System	Yes	Lookup	The accounting system.
Accounting Scope	Yes	Lookup	The accounting scope.
Chart Account	Yes	Lookup	The accounting chart.
Operational Item	No	Lookup	The item of operations.
Currency	No	Lookup	The currency of the accounting entry line.

Field	Required	Type	Description
Take From Product	No	Bool	If true, then the configurations for each transaction are inherited from the banking product level.

3. Click **Save and close**.

Transaction Accounting Models

This section holds the accounting models, all the rules used in order to generate accounting entries for each transaction.

The details from the **Debit Account Rule** and the **Credit Account Rule** are defined by the information from the **Transaction Item Accounting Configuration** section. All other details are defined by the information from the **Transaction Value Type** section.

1. To add a new transaction accounting model, click the **Insert** button under the **Transaction Accounting Models** section. The **Add Transaction Accounting Model** page opens.

2. Fill in the following fields:

ADD TRANSACTION ACCOUNTING MODEL

TRANSACTION ACCOUNTING MODEL

Transaction Type

DocTest

Accounting System

DocTest

Line Condition

DocTest

Debit Account Rule

DocTest

Credit Account Rule

DocTest

Debit Customer Rule

SourcePartnerId

Credit Customer Rule

DestinationPartnerId

Accounting Entry Value Rule

DocTest

currencyRule

CurrencyId

Item Rule

ItemId

DescriptionRule

DescriptionText

EntityId Rule

DocTest

Transaction Id Rule

DocTest

Transaction Detail Rule

ContractId

Transaction Value Type

Field	Required	Type	Description
Transaction Type	No	Lookup	The transaction type. It is auto-filled.
Accounting System	No	Lookup	The accounting system.

Field	Required	Type	Description
Line Condition	No	Text	The condition applied in order to post the accounting entry line.
Debit Account Rule	No	Text	The accounting entry value of the debit account. It is auto-filled.
Debit Customer Rule	No	Text	The rule to save the partner transaction in the debit-credit relationship. It is auto-filled.
Accounting Entry Value Rule	No	Text	The posted accounting entry value.
Item Rule	No	Text	The transaction item of the accounting entry line. It is auto-filled.
EntityId Rule	No	Text	The internal status of the record.
Transaction Detail Rule	No	Text	The rule to identify and post the ID of the operational transaction detail.
Credit Account Rule	No	Text	The credit account of the accounting entry line. It is auto-filled.
Credit Customer Rule	No	Text	It is auto-filled by the destination partner ID.
currencyRule	No	Text	The accounting entry line currency. It is auto-filled.
DescriptionRule	No	Text	The description of the generated accounting entry. It is auto-filled.
Transaction Id Rule	No	Text	The related contract ID of the transaction.
Transaction Value Type	No	Lookup	The value type of the transaction. It is defined in the Transaction Value Type section.

3. Click **Save and close**.

General Ledger Transactions

The **Operational Ledger** comes with a number of menus for viewing and managing transactions, customizing transaction values, and so on. It captures ledger details for each financial transaction in order to create ledger records according to the accounting system implemented. It also allows customers to feed the same transaction data into different accounting systems and make financial reports. The menus are presented in the below pages:

- [GL Operation Transaction](#)
- [Accounting Entry](#)

OL Operation Transaction

This menu allows users to search for existing transactions or create new ones. It holds the header items of a transaction. In the data model, the entity FTOS_GL_OperationTransaction stores all the transaction data from the ledger. All the transaction value types are the details saved into the **Operation Transaction Value** section.

Creating Operation Transactions

1. Log into the **FintechOS Portal**.
2. In the main menu, expand **General Ledger Transactions** and select **GL Operation Transaction**. The **GL Operation Transaction List** page opens.
3. At the top-right corner of the page, click the **Insert** button to create an operation transaction.
4. Fill in the following fields:

ADD GL OPERATION TRANSACTION

GL OPERATION TRANSACTION

Transaction Type

DocTest

Currency

EUR

Accounting Date

16/07/2021

Transaction Date

23/07/2021

Product

doctest

Item

doctest

Source Partner

DocTest

Destination Partner

DocTest

Analytic Debit Code

doctest

Analytic Credit Code

doctest

Field	Required	Type	Description
Transaction Type	No	Lookup	The type of the transaction.
Accounting Date	No	InvariantDate	The accounting date.
Product	No	Text	The product of the transaction.
Source Partner	No	Lookup	The source partner.
Analytic Debit Code	No	Text	The analytic debit code.
Currency	No	Lookup	The currency of the accounting entry line.
Transaction Date	Yes	InvariantDate	The transaction date.
Item	No	Text	The transaction item.
Destination Partner	No	Lookup	The destination partner.
Analytic Credit Code	No	Text	The analytic credit code.

5. Click **Save and reload**. The **Operation Transaction Values** and the **Accounting Entries** sections are displayed.

NOTE

To automatically generate accounting entries, click the **Generate Accounting Entries**

button. The accounting entries generated are shown in the **Accounting Entries** section.

Operation Transaction Values

This section allows users to customize the details of each transaction. It holds the detail items of a transaction.

- 1. To add a new operation transaction value, click the **Insert** button under the **Operation Transaction Values** section. The **Add Operation Transaction Value** page opens.
- 2. Fill in the following fields:

ADD OPERATION TRANSACTION VALUE

OPERATION TRANSACTION VALUE

Name

DocTest

Transaction Value Type

DocTest

Value

0

Operation Transaction

2325 C EUR

Currency

EUR

Text

DocTest

Field	Required	Type	Description
Name	No	Text	The name of the transaction value.
Transaction Value Type	No	Lookup	The value type of the transaction.
Value	No	Text	The value of the transaction.
Operation Transaction	No	Lookup	The operation transaction ID.
Currency	No	Lookup	The currency of the accounting entry line.
Text	No	Text	The description of the transaction.

- 2. Click **Save and reload**. The **Accounting Entries** section is displayed.

NOTE

The **Accounting Entries** section holds the accounting entries of an operation transaction value. From here, users can add, delete, or export accounting entries. For more information on creating accounting entries, see [Accounting Entry](#).

Accounting Entry

This menu contains the accounting entries generated from the operation transaction and the operation transaction value records.

Creating Accounting Entries

1. Log into the **FintechOS Portal**.
2. In the main menu, expand **General Ledger Transactions** and select **Accounting Entry**. The **Accounting Entries List** page opens.
3. At the top-right corner of the page, click the **Insert** button to create an accounting entry.
4. Fill in the following fields:

ADD ACCOUNTING ENTRY

ACCOUNTING ENTRY	
Accounting Date 19/07/2021	Accounting Value 0
Debit Partner DocTest	Credit Partner DocTest
Debit Account Doc Test	Credit Account Doc Test
Analytic Debit Account Code 0000	Analytic Credit Account Code 0000
Currency EUR	Exchange Rate 0
Equivalent Value 0	Item TransactionFee
Description DocTest	

Field	Required	Type	Description
Accounting Date	No	Date	The date of the accounting entry line.
Accounting Value	No	Numeric	The value of the entry.
Debit Partner	No	Lookup	The debit partner.
Credit Partner	No	Lookup	The credit partner.
Debit Account	No	Lookup	The debit account
Credit Account	No	Lookup	The credit account.
Analytic Debit Account Code	No	Numeric	The code of the analytic debit account.
Analytic Credit Account Code	No	Numeric	The code of the analytic credit account.
Currency	No	Option set	The currency of the accounting entry line.
Exchange rate	No	Numeric	The exchange rate of the accounting entry line.
Equivalent Value	No	Numeric	The value after the exchange rate has been applied.
Item	No	Lookup	The accounting registration item.
Description	No	Text	The description of the accounting entry.

5. Click **Save and close**.

HINT

Users can add, delete, or export accounting entries.

Operational Ledger SDK

The FTOS_GL_OperationTransactionHelper library server automation script and the FTOS_GL_GetAttributesForTransactionValueType server automation script list and endpoint have been implemented to help generate accounting entries in the Operational Ledger.

The functions available for the implemented library and script are presented below. These functions allow Operational Ledger to convert the transaction value type details into another entity that contains the actual transaction values. Then, based on certain rules, the generated accounting entries are inputted on accounting models. For more information on accounting models, see [Transaction Accounting Models](#).

FTOS_GL_OperationTransactionHelper Library

1. FTOS_GL_SetValues_TransactionOperation

Description:

Based on the transactionTypeName function, it retrieves all the information needed from the FTOS_GL_TransactionType and the FTOS_GL_TransactionValueType entities.

Iterates through the transaction value types list and creates an object with the needed properties for each of the below entities. If a line for the same record ID is found, it updates it, if not, a new one is inserted.

- FTOS_GL_OperationTransaction
- FTOS_GL_OperationTransactionValue

For both objects, the value is retrieved by calling the getFinalValue function.

Input:

- **recordId** - the record ID for which the transaction operation is logged
- **transactionTypeName** - the transaction type (for example Loan Contract, Repayment, Disbursement, Interest Capitalization)
- **defaultValuesJson** - a stringified json containing a list of objects with the following form:

```
[
  {
    attributeName: 'DescriptionText',
    value: 'Disburse 1500'
  },
  {
    attributeName: 'ProvisionAmount',
    value: '15.00'
  }
]
```

Output:

No values are returned. The outcome of this function creates the FTOS_GL_OperationTransaction and the FTOS_GL_OperationTransactionValue entities.

CallExample:

```
var operationTransactionHelperLibrary = importLibrary('FTOS_GL_OperationTransactionHelper');
var eventId = '64f1c182-d329-4d94-892e-c3cf4556d186';
var defaultJsonValues= [
  {
    attributeName: 'DescriptionText',
    value: 'Disburse 1500'
  },
  {
    attributeName: 'ProvisionAmount',
    value: '15.00'
  }
];
operationTransactionHelperLibrary.FTOS_GL_SetValues_
TransactionOperation(eventId, 'Disbursement', JSON.stringify(
defaultJsonValues));
```

2. getFinalValue

Description:

The `json` checks if there is any default value for the `attributeName`, from the **Value Type Name** field. If no values are returned, the **Formula** field is checked. When neither field returned any values, the source entity of the attribute from the **Value Type Attribute** field is checked.

Input:

- **defaultValues** - json object containing a list of objects with the following form:

```
{
  attributeName: 'DescriptionText',
  value: 'Disburse 1500'
},
{
  attributeName: 'ProvisionAmount',
  value: '15.00'
}
]
```

- **recordId** - the record ID of the retrieved value
- **entityId** - the ID of the entity for which the value is retrieved (source entity ID)
- **valueTypeName** - the value type name defined in the [Transaction Value Type](#) section, used to check for default values
- **valueTypeAttributeName** - the name of the attribute from the source entity
- **valueTypeFormula** - the value type formula, used for getting the value

Output: the value of a specific transaction value type

Call example:

```
{
  attributeName: 'DescriptionText',
```



```

    value: 'Disburse 1500'
  },
  {
    attributeName: 'ProvisionAmount',
    value: '15.00'
  }
]

```

3. getAttributeValue

Description: a general fetch is made in order to retrieve the attribute value.

Input:

- **recordId** - the record ID of the retrieved value
- **entityId** - the ID of the entity for which the value is retrieved
- **attributeName** - the name of the attribute from the entity for which the value is retrieved

Output: the attribute value

CallExample:

```

var defaultValue = ;
var recordId = '631def4a-c0ab-4c3f-9a23-f5ffc4488f13';
var sourceEntityId= '631def4a-c0ab-4c3f-9a23-f5ffc4488f13';
var valueTypeName = 'ProvisionAmount';
var valueTypeAttributeName = '';
var formula = 'availableValue * 2';
var valueToBeSaved = getFinalValue(defaultValue, recordId,
sourceEntityId, valueTypeName , valueTypeAttributeName,
formula );

```

4. isAttributeInvariantDate

Description: it checks if the attribute value is an invariant date or not.

Input:

- **attributeId** - the attribute ID

Output: true or false.

Call example:

```
var recordId = '631def4a-c0ab-4c3f-9a23-f5ffc4488f13';
var returnedValue = getAttributeValue(recordId, 'FTOS_CB_Contract', 'availableAmount')
```

5. contains

Description: iterates through an array and if the condition is satisfied, the object is returned, otherwise no results are returned.

Input:

- **arr** - an array of objects
- **key** - the parameter name of the object
- **val** - the value on which the condition is made

Output: if the array contains an object with that specific value for that key, it returns the object, otherwise no output is returned.

Call example:

```
var defaultJsonValues= [
  {
    attributeName: 'DescriptionText',
    value: 'Disburse 1500'
  },
  {
    attributeName: 'ProvisionAmount',
    value: '15.00'
  }
];
```

```
var value = contains(defaultJsonValues, 'attributeName',
  'ProvisionAmount');
//value will be
//{
//  attributeName: 'ProvisionAmount',
//  value: '15.00'
//}
```

6. validateFormulaAttribute

Description: using a regex pattern `/[A-Za-z]+/gm`, it matches all the attribute names and it iterates through all the findings to check if it's an attribute of the source entity from the transaction type ID using the `existsAttributeInSourceEntity` function.

Input:

- **formula** - a field which validates basic math operation formulas with the attributes of a source entity
- **transactionTypeId** - the ID of the transaction type; the formula is validated against the source entity ID of the transaction type

Output: if the attribute inputted in the formula field is not part of the entity, then following error is displayed:

Name should be an attribute name from header table FTOS_GL_ OperationTransaction.

Call example:

```
var formula = 'availableValue * 2'
var transactionTypeId = '631def4a-c0ab-4c3f-9a23-f5ffc4488f13';
validateFormulaAttribute(formula, transactionTypeId);
```

7. existsAttributeInEntity

Description: a fluent query is done on the attribute table conditioning on the entity ID and the attribute name.

Input:

- **attributeName** - attribute name that is checked
- **entityId** - entity ID on which the attribute is checked

Output: true or false.

Call example:

```
var blnExists = existsAttributeInEntity('currencyId',  
    '631def4a-c0ab-4c3f-9a23-f5ffc4488f13');
```

8. existsAttributeInTableName

Description: a fluent query is done on the attribute and the entity table conditioning on the entity table name and the attribute name.

Input:

- **attributeName** - attribute name that is checked
- **entityTableName** - table name of the entity on which the attribute is checked

Output: true or false.

Call example:

```
var blnExists = existsAttributeInTableName('currencyId',  
    'FTOS_CB_Contract');
```

9. existsAttributeInSourceEntity

Description: based on the `transactionTypeId` function, the source entity ID is retrieved from the `FTOS_GL_TransactionType`. Then, a fluent query is done on the attribute and the entity tables, conditioning on the source entity ID and the attribute name.

Input:

- **attributeToCheck** - attribute name that is checked
- **transactionTypeId** - the ID of the transaction type, used to retrieve the source entity ID

Output: true or false

Call example:

```
var blnExists = existsAttributeInSourceEntity('currencyId',
'631def4a-c0ab-4c3f-9a23-f5ffc4488f13');
```

10. existsOperationTransactionForRecordId

Description: a fluent query that searches in the `FTOS_GL_OperationTransaction` table for record IDs that exist in the **EntityID** column lines.

Input:

- **recordId** - the record ID that checks if an operation transaction exists for it

Output: the operation ID is returned if found. If not, no results are displayed.

Call example:

```
var blnExists = existsAttributeInSourceEntity('currencyId',
'631def4a-c0ab-4c3f-9a23-f5ffc4488f13');
```

11. existsOperationTransactionValue

Description a fluent query, that searches in the FTOS_GL_OperationTransactionValue table for a record with a specific operation transaction ID and transaction value type ID.

Input:

- **operationTransactionId** - operation transaction ID
- **transactionValueType** - transaction value type ID

Output: the operation transaction value ID is returned. If not, no results are displayed.

Call example:

```
var result = existsOperationTransactionForRecordId
('631def4a-c0ab-4c3f-9a23-f5ffc4488f13');
```

FTOS_GL_GetAttributesForTransactionValueType Server Script and Endpoint

Description: It retrieves a list of attributes conditioned by the source entity ID from the transaction type. If the attribute value type is numeric, only numeric or whole number type attributes are displayed. If not, all attributes are retrieved.

Input:

- **attributeValueTypeId** - the attribute value type ID which is either numeric or text
- **transactionTypeId** - the transaction type ID that is used for retrieving the source entity

Output: a list of attributes {attributeid: '', attributeType: '', displayName: '', name: ''}

Call example:

```
var result = ebs.callActionByNameAsync("FTOS_GL_  
GetAttributesForTransactionValueType",  
{attributeValueTypeId: '', transactionTypeId:''})  
    .then(function(res){  
        });
```

Glossary

A

Accounting

The measurement, processing, and communication of financial and non financial information about economic entities such as businesses and corporations.

Accounting chart

An index of all the financial accounts in the general ledger of a company.

Accounting system

The system used to manage the income, expenses, and other financial activities of a business.

C

Circulation

The circulation of money/ transactions in a given period.

Credit

An entry recording of a received sum. It has a positive value.

D

Debit

An entry recording of an owed sum. It has a negative value.

I

IBAN

International Bank Account Number

L

Ledger

A book in which the monetary transactions of a business are posted in the form of debits and credits.

Legal Entity

An association, corporation, partnership, proprietorship, trust, or individual that has legal standing in the eyes of law. A legal entity has legal capacity to enter into agreements or contracts, assume obligations, incur and pay debts, sue and be sued in its own right, and to be held responsible for its actions.

T

Transaction

An instance of buying or selling something.