NiceLabel Cloud User Guide

Rev-2022-11
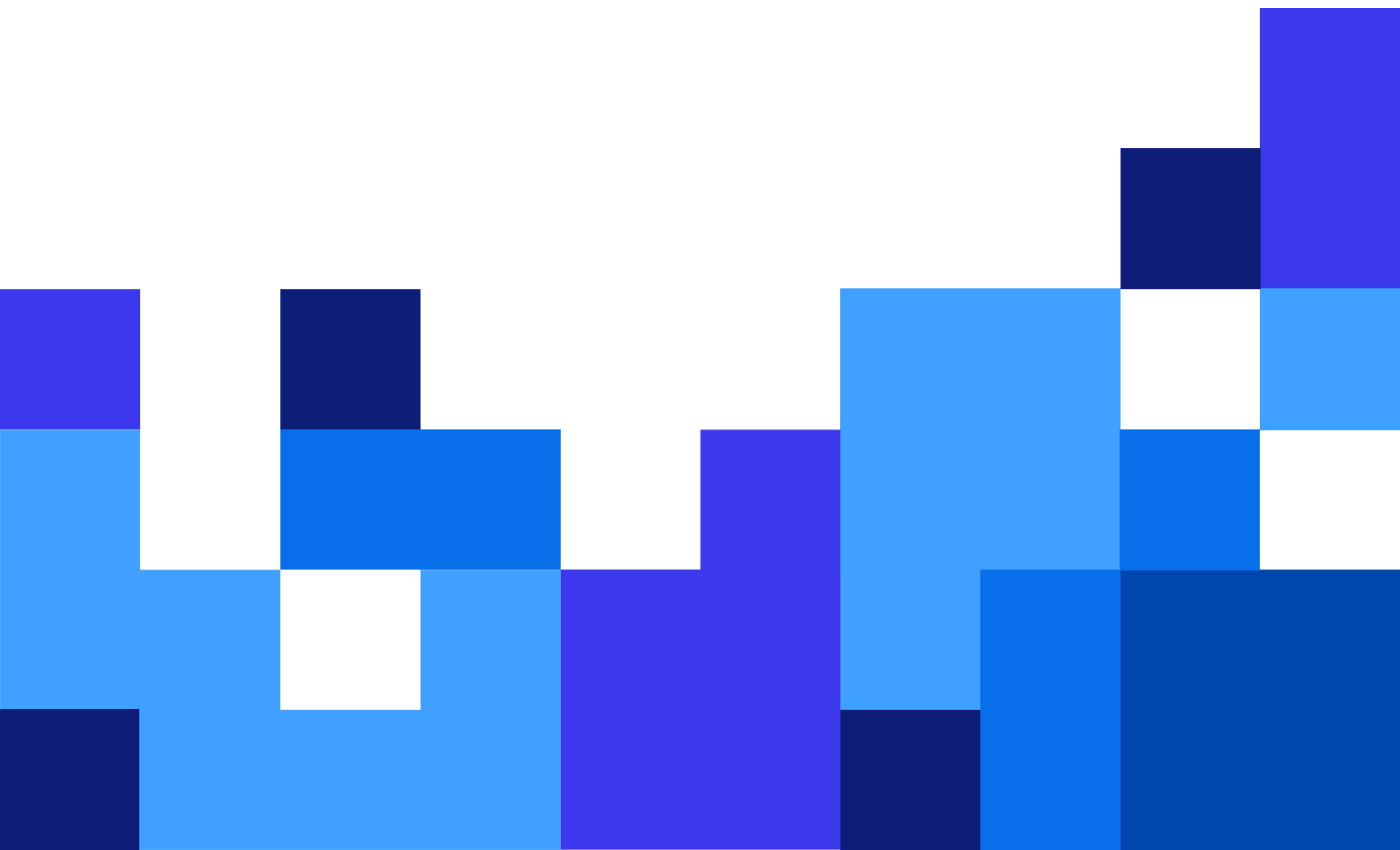
Table of Contents

1. Getting started with NiceLabel Cloud

NiceLabel Cloud from Loftware is our cloud-based Label Management System.

Whether your business is small or large, you can use NiceLabel Cloud to manage and print your labels from anywhere with no IT infrastructure. You access NiceLabel Cloud Control Center from your browser whenever you need to manage and print labels.

NiceLabel Cloud makes it easy for teams large and small to efficiently design new labels, stay up to date with changing laws, avoid product recalls and regulatory fines, and quickly scale your labeling operations.

You design, manage, and print your labels with one system. NiceLabel Cloud is a collection of tools to help your designers, quality assurance engineers, IT managers, and printing operators work together and make sure the correct data prints on the right label at the right time.

Central cloud management of your labeling system provides you lower costs, improved quality assurance, and faster time-to-market without significant hardware investments. You get secure, scalable, standardized labeling for your suppliers and your entire organization.

NiceLabel Cloud provides:

- Reliable IT infrastructure

- Advanced security measures

- Thorough maintenance plans

- Comprehensive disaster recovery processes

Stakeholders in your company, including IT managers, quality assurance managers, operations managers, and others receive benefits from the infrastructure, security, and reliability of NiceLabel Cloud.

For more information, visit https://www.nicelabel.com/label-cloud or read about NiceLabel Control Center.

1.1. Introducing NiceLabel Cloud

https://www.youtube.com/embed/xYlJJTrG8AY

This video introduces you to NiceLabel Cloud. You'll learn the benefits of cloud-hosted label management for businesses big and small, and see a label's journey through NiceLabel Cloud, including:

- Creating a template in Designer.

- Storing and managing label templates in Control Center.

- Implementing versioning and workflows for Quality Assurance in Control Center.

- Integrating your ERP system and automating your printing with Automation.

- Printing your labels with NiceLabel Print and other NiceLabel applications.

1.2. Activating NiceLabel Cloud

https://www.youtube.com/embed/I_uQwtPKr0U

Learn how to activate NiceLabel Cloud, including:

- Where to find NiceLabel Cloud activation information.

- How to sign in to your NiceLabel Cloud account after you activate.

1.3. Adding users

In this video, you will learn:

- Differences between individual users (Guest users) and Organizational directory users.

- How to invite Guest users to NiceLabel Cloud.

- How to accept NiceLabel Cloud invitations as a Guest user.

1.4. Managing Access Roles and Permissions

https://www.youtube.com/embed/bfQ5_jenxGQ

Manage your Access Roles and permissions to secure NiceLabel Cloud. You will learn:

- The relationship between Users, Access Roles, and permissions.

- How to create new Access Roles with permissions.

- How to assign Access Roles to your users.

1.5. Downloading and connecting software

You will learn:

- How different NiceLabel Cloud software helps you work.

- How to keep your software updated to the latest version.

- How to connect your software to NiceLabel Cloud.

1.6. Printing your first label

https://www.youtube.com/embed/sMo5XNQVmDM

NiceLabel Cloud helps you print your labels from anywhere. You will learn:

- How to store and share labels.

- How to print your first label from NiceLabel Cloud.

- How to monitor your printing progress.

2. NiceLabel Cloud Infrastructure, security, and Maintenance

NiceLabel Cloud is NiceLabel's cloud-based Label Management System.

Cloud management enables you to reap every benefit of digitally transforming your labeling (including lower costs, improved quality assurance, and faster time-to-market) without significant upfront hardware investments. You get secure, scalable, standardized labeling for your suppliers and your entire organization **with an ROI of less than 6 months**.

NiceLabel Cloud creates these business benefits by providing you with:

- Reliable IT infrastructure

- Advanced security measures

- Thorough maintenance plans

- Comprehensive disaster recovery processes

Stakeholders in your company, including IT managers, quality assurance managers, security officers, and others all see benefits from the infrastructure, security, and reliability of NiceLabel Cloud.

Learn more about NiceLabel Cloud here: https://www.nicelabel.com/label-management-system/label-cloud.

2.1. Infrastructure

2.1.1. Maintaining high availability

NiceLabel develops NiceLabel Cloud on Microsoft's reliable Azure platform, ensuring world-class infrastructure and availability.

Your Service Level Agreement **guarantees high service availability** as defined in your Master Software Subscription and Services Agreement.

NiceLabel Cloud architecture prioritizes high availability and eliminates single points of failure with multiple redundancies. We work in collaboration with Microsoft architects to build NiceLabel Cloud following best practices. Microsoft features NiceLabel Cloud in their global catalogue of Azure-based solutions.

You can find NiceLabel Cloud on Appsource and the Azure Marketplace.

2.1.2. NiceLabel Cloud web availability

NiceLabel builds reliability and redundancy into our cloud-based web architecture.

NiceLabel Cloud websites run on clusters of virtual machines (nodes). Each node runs within different Availability Zones (separate physical locations within Azure data centers). If a node fails, the cluster continues to operate on other nodes.

NiceLabel Cloud runs on several clusters. In the event of a large-scale whole-cluster failure, we can migrate your NiceLabel Cloud accounts to different clusters.

2.1.3. Redundancies and backups

NiceLabel Cloud leverages Azure to provide high availability for your applications and data.

Your data is hosted on redundant database servers to ensure no service interruptions. If one database server fails, another database server takes over (seamless service).

Azure performs database backups and stores copies of each backup in multiple physical locations. We can restore your data from any time in the previous 30 days from server backups. NiceLabel Cloud creates full backups every week, differential backups every 12 hours, and transaction log backups every 5-10 minutes.

2.1.4. Printing offline

Uninterrupted label printing is NiceLabel Cloud's most mission-critical process.

In case of interruptions, NiceLabel Cloud allows you to print from applications on your computers without connecting to NiceLabel Cloud backend in Azure. With system configuration, you can print in offline mode for up to 5 days. NiceLabel Cloud automatically synchronizes your printing records from offline printing.

> **NOTE**
>
> We recommend running NiceLabel Cloud on reliable internet connections. Offline printing requires you to configure your system to rely on your local cache, and is limited to cached labels and locally available data. NiceLabel's professional services team can design or advise you on best practice solution configurations for offline printing.

2.1.5. Assessing and mitigating risk

RISK	MITIGATION
SERVICE OVERLOAD	We monitor performance data. In the event of poor performance, we scale out resources or move you to another web server.

RISK	MITIGATION
SYSTEM DATABASE FAILURE	We geo-replicate your database in another location. In the event of system database failure, your system automatically switches to a geo-replicated copy after 1 hour. Your database also has point-in-time backups for the previous 30 days, so in the event of data corruption, we can restore your data.
USER DATABASE FAILURE	Your database has geo-replicated point-in-time backups for the previous 30 days. In the event of data corruption or loss, we can restore your data.
AUTHENTICATION FAILURE	Your providers (Microsoft, Google) are responsible for service availability. In the event of authentication not working properly (i.e. bugs) NiceLabel cooperates with providers to resolve issues.
WEB SERVER FAILURE	All websites run on multiple nodes (clusters). If one node fails, another takes over.
DATA CENTER FAILURE	Handled by PaaS provider (Microsoft Azure). In the event of data center failures, NiceLabel contacts Microsoft (support request) to resolve issues. If Microsoft doesn't resolve issues or promptly provide estimated restoration times, NiceLabel restores services in another data center.
DNS FAILURE	Microsoft guarantees 100% DNS services availability.

2.2. Security

NiceLabel makes significant efforts to ensure NiceLabel Cloud security.

We implement the latest security standards and perform automatic and manual NiceLabel Cloud security checks. We're committed to providing you trustworthy service while applying policies, technologies, and controls to protect data you entrust to NiceLabel Cloud.

2.2.1. Layered security

NiceLabel keeps your systems and data secure in multiple ways.

Most security breaches don't occur from someone breaking into cloud data centers. Instead, attackers typically exploit cloud application vulnerabilities. To prevent attacks, we combine multiple mitigation strategies and security controls to protect your resources and data.

Our layered security includes:

- Employee education

- Physical security

- Network security

- Web security

- API-based cloud security

- Data encryption

By running on Microsoft Azure, NiceLabel Cloud inherits many platform and infrastructure security approaches and best-practice implementations. Microsoft handles core data center security and inspects dataflows from the internet to help secure your network against intrusions and malware attacks.

We design cloud applications following modern security-conscious programming practices. We use encryption techniques and execute testing procedures to develop code and launch products.

Our development teams complete IT security-related training on software development to strengthen their information security awareness and experience.

2.2.2. Role-based access

NiceLabel Cloud authenticates with Microsoft and Google (Oauth2/OpenID Connect).

We integrate trusted providers into NiceLabel Cloud security to authenticate your user identities and protect your users from attacks. This allows NiceLabel to focus on core features and leaves identification to experts you know.

You can define your users with LDAP directory services or use your Microsoft Office 365 or Active Directory (AD) accounts (available as Azure Active Directory for cloud applications). NiceLabel Cloud does not include authentication mechanisms or custom authentication logic.

2.2.3. Database security

Database separation is essential and ensures you get additional layers of security.

You can only access your own assigned application database. You cannot access application databases directly with management applications or via API. Database ownership prevents other customers from accessing or reading any of your data.

Dependent on your NiceLabel Cloud subscription, you can access user-based cloud databases to store printing data and for daily intermediate master data exports from ERP systems. You don't need user-based cloud database access to run NiceLabel Cloud web applications.

User database access is entirely customer specific. When you claim your user database, we create your first administrative account so you can manage your database and grant user access yourself.

2.2.4. Data encryption

NiceLabel encrypts your data to keep your business safe.

Your data can occupy two states in NiceLabel Cloud-- **data in transit** and **data at rest**. Your data can be exposed to risks in both states. NiceLabel Cloud uses encryption to protect data in transit and at rest from unauthorized access or theft.

Data actively moving between devices or networks across the internet is data **in transit**.

We protect your data in transit from local storage to NiceLabel Cloud storage. We encrypt your data in transit on one end and decrypt it on the other to prevent eavesdropping from unauthorized clients. NiceLabel Cloud uses modern data encryption communication protocols (TLS and HTTPS) for privacy and data integrity.

We encrypt your data when you connect to NiceLabel Cloud with:

- **Browsers.** You can use any modern web browser to interact with our web applications.

- **NiceLabel clients.** All our clients, including Designer, Print, Automation, and Web Client, use secure encrypted channels to request NiceLabel Cloud data and to send back logs and updates.

Data not actively moving between devices or networks across the internet is data **at rest**.

NiceLabel Cloud receives and stores your data in Azure SQL databases unique to you. We follow protective security measures to prevent anyone from accessing, modifying, or stealing your at-rest data:

- Only you have access to product databases you own.

- Your Azure SQL databases use transparent data encryption (TDE). TDE gives you real-time database encryption and decryption using AES 256 encryption algorithms.

2.2.5. API security

NiceLabel software uses Azure APIs for secure data exchanges and inter-application communication.

- **Service Bus:** the communication system between mutually interacting software applications in service-oriented architecture. We use Service Bus to communicate with your on-premise infrastructure, either through cloud-connected IoT printers or cloud triggers (running in NiceLabel Automation). Service Bus creates outbound connections from your backend to NiceLabel Cloud and makes it possible to call your backend from the cloud.

- **Azure functions.** Our published APIs for NiceLabel Cloud (Cloud Print API and Cloud Trigger API) call Azure functions for additional processing, which in return call the correct Service Bus endpoints. For example, when you execute "print" in Cloud Print API, NiceLabel Cloud generates a print job, knows where your IoT cloud printer is, and delivers your print job to your printer. We have traffic limits in place to prevent the abuse of APIs.

2.2.6. Health monitoring

NiceLabel continuously monitors your hosted system health with Azure Insight.

Insight automatically detects performance anomalies and includes powerful analytics tools to help us diagnose issues and improve products by understanding how our customers use NiceLabel Cloud.

We use Insight to:

- Monitor abnormal traffic and respond quickly to possible threats.

- Detect and respond to higher demands for services.

- Continuously improve performance and stability

2.2.7. Testing

NiceLabel tests all code extensively to ensure safety and high quality.

Commonly exploited software vulnerabilities include defects, bugs, and logic flaws. Our development team strives to produce quality code through best practice techniques, including:

- Pair programming

- Recurring code reviews

- Adhering to secure code standards

- Running multiple tests

Our general policy is to automatically test everything we can. We perform continuous regression testing for each release throughout the lifecycle of our software to ensure industry-grade quality standards.

In addition to our experienced internal testing teams, we continuously contract third-party security assessment specialists to make sure our software is safe, secure, and ready for you to use.

2.2.8. Internal testing

NiceLabel development teams design and execute an expansive array of manual and automated tests for each new software build.

We increase the number of tests and the number of test team members for final testing before releases. Any security flaws we detect results in writing new automated tests to prevent problems from appearing again.

2.2.9. Third-party security assessments and penetration testing

NiceLabel contracts third-party IT security specialists for major and minor releases.

Our security experts access our software like our customers, but use their expertise to assess our web and desktop applications to identify exploitable vulnerabilities. Testing involves building custom threat profiles to uncover security vulnerabilities specific to our applications and web technology.

Our third-party security testers use the OWASP Testing Guide for test execution and verification. The Open Web Application Security Project (OWASP) is a worldwide free and open community focused on improving the security of application software.

2.2.10. Acceptance in regulated environments

NiceLabel testing is compliant and trusted by customers in regulated industries.

NiceLabel Cloud customers in regulated industries including healthcare, pharmaceuticals, medical device manufacturing, food & beverage, and others rely on test results we provide.

We will also work with you to perform tests with your own testing tools and procedures.

2.3. Maintenance and updates

NiceLabel keeps you up to date.

Our controlled process updates NiceLabel Cloud automatically, providing you seamless hassle-free updates with **no platform downtime**.

We update with major product releases every two years, minor releases twice per year, and service releases in between (as needed):

RELEASE TYPE	EXAMPLE	DESCRIPTION
MAJOR	NiceLabel **10**	Major product releases include significant feature updates, performance improvements, and bug fixes.
MINOR	NiceLabel 10.**1**	Minor releases include feature updates, performance improvements, and bug fixes.
SERVICE	NiceLabel 10.1.**1**	Service releases include bug fixes.

For full update schedules and more details, read our Product Lifecycle Policy.

2.3.1. Updating NiceLabel Cloud

NiceLabel provides new NiceLabel Cloud deployments for every software release.

We prepare and test each new deployment before we update your subscription, allowing you to upgrade seamlessly without downtime. We update your subscription by simply moving you from your previous deployment version to your new one. If you experience any unexpected problems, we return you to your previous version while we solve any problems.

We have transparent update plans and clearly-defined procedures in place for all release types (major, minor, and service). You receive email notifications before any NiceLabel Cloud version updates, allowing you to prepare for, test, and track every upgrade.

2.3.2. Update timeline for NiceLabel Cloud releases

> **NOTE**
>
> **Release days** are when major, minor, or service releases become publicly available.

TIME	EVENT	DESCRIPTION
RELEASE DAY	Email notification	You get an email notification saying we will start with the NiceLabel Cloud upgrade process. The email includes information about the account that will be updated and the time of the update.
RELEASE + 7 DAYS	Email notification	You get an email notification saying your NiceLabel Cloud Essentials/Business sandbox/Compliance sandbox environment will be updated to the latest version in the next 24 hours.
	NiceLabel Cloud Essentials	We update your NiceLabel Cloud Essentials account to the latest version. You have no production environment downtime, we simply move your account.
	NiceLabel Cloud Business: Sandbox environment updates	We update your sandbox environment to the latest version. You have no sandbox environment downtime, we simply move your account. This procedure allows you to see how your production environment will handle the update.
	NiceLabel Cloud Compliance: Sandbox environments updates	We update your sandbox environment to the latest version. You have no sandbox environment downtime, we simply move your account. This procedure allows you to see how your production environment will handle the update and allows you an assessment of your system validation.
RELEASE + 14 DAYS	Email notification	You get an email notification saying your BusinessNiceLabel Cloud sandbox environment will be updated to the latest version in the next 24 hours.
	NiceLabel CloudBusiness: without Sandbox environment	We update your production environment to the latest version. You have no production environment downtime, we simply move your account.
RELEASE + 28 DAYS	Email notification	You get an email notification saying your NiceLabel Cloud Business production environment will be updated to the latest version in the next 24 hours.

TIME	EVENT	DESCRIPTION
	NiceLabel Cloud Business: Production environment updates	We update your production environment to the latest version. You have no production environment downtime, we simply move your account.
RELEASE + 3 MONTHS	Email notification	You get a third and final email notification saying your NiceLabel Cloud Compliance production environment will be updated to the latest version in the next 24 hours.
	NiceLabel Cloud Compliance: Production environment updates	We update your production environment to the latest version. You have no production environment downtime, we simply move your account.

NiceLabel Cloud Compliance release cycle

NiceLabel Cloud is updated once a year. Before the production environment is upgraded, we provide a three-month testing period. During the testing period, users can contact NiceLabel technical support to test and resolve possible open issues before the production environment is upgraded.

2.3.3. Updating privately hosted software

You can host NiceLabel Cloud modules on your private infrastructure with your subscription:
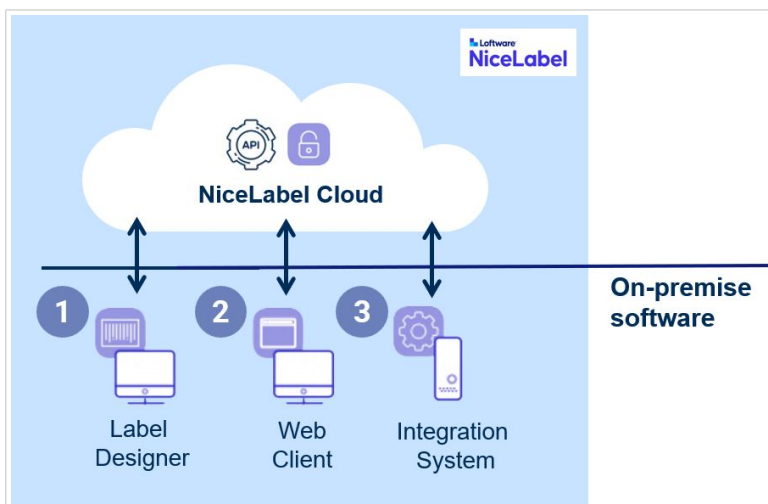
1. Your **Label Designer** designs labels and configures your printing application.

2. Your **Web Printing Client** runs your printing application through the web.

3. Your **Integration System** integrates your printing.

Your privately hosted software always links seamlessly with your NiceLabel Cloud subscription.

We regularly update NiceLabel Cloud, but your **privately hosted software does not update automatically**. Our up-to-date NiceLabel Cloud backend still runs older versions of your privately hosted software.

Until you update your privately hosted software, you may not be able to use some new NiceLabel Cloud features and functionality. To get every benefit from your NiceLabel Cloud subscription, periodically update your privately hosted software as we release newer versions of NiceLabel Cloud. Schedule your updates to fit your maintenance and production requirements.

2.4. Disaster recovery

NiceLabel Cloud is stable and secure.

But if things go wrong, NiceLabel has comprehensive disaster recovery plans in place. Our teams work hard to minimize your downtime and help you get back to business as usual as quickly as possible.

2.4.1. Terms and definitions

NAME	DEFINITION
INCIDENT	A situation that might be, or could lead to, a disruption, loss, or disaster.
DISASTER	Any condition that results in a prolonged inability to access or use NiceLabel Cloud. A disaster requires recovery action to restore normal operation.
INCIDENT RESPONSE TEAM	Includes members of our support and application development teams who respond to incident support requests from customers. Incident response team members receive alerts from our monitoring system. Incident response teams resolve incidents or escalate incidents to disasters.
PROVISIONING TEAM	Includes application development team members responsible for managing NiceLabel Cloud. Besides regular management, our provisioning team supports our incident response team in resolving incidents.
DISASTER RECOVERY TEAM	Assembles in the event of a disaster scenario to recover the service from the disaster. Includes provisioning team members.
DISASTER RECOVERY PROCESS MANAGEMENT TEAM	Monitors, reviews, and makes changes to disaster recovery processes to ensure effectiveness. This team is not directly involved in disaster response, but reviews each disaster scenario to improve processes.

2.4.2. Flow chart

Disaster recovery flow chart showing Incident Response Team, Provisioning Team, Disaster Recovery Team, and Disaster Recovery Board swimlanes.
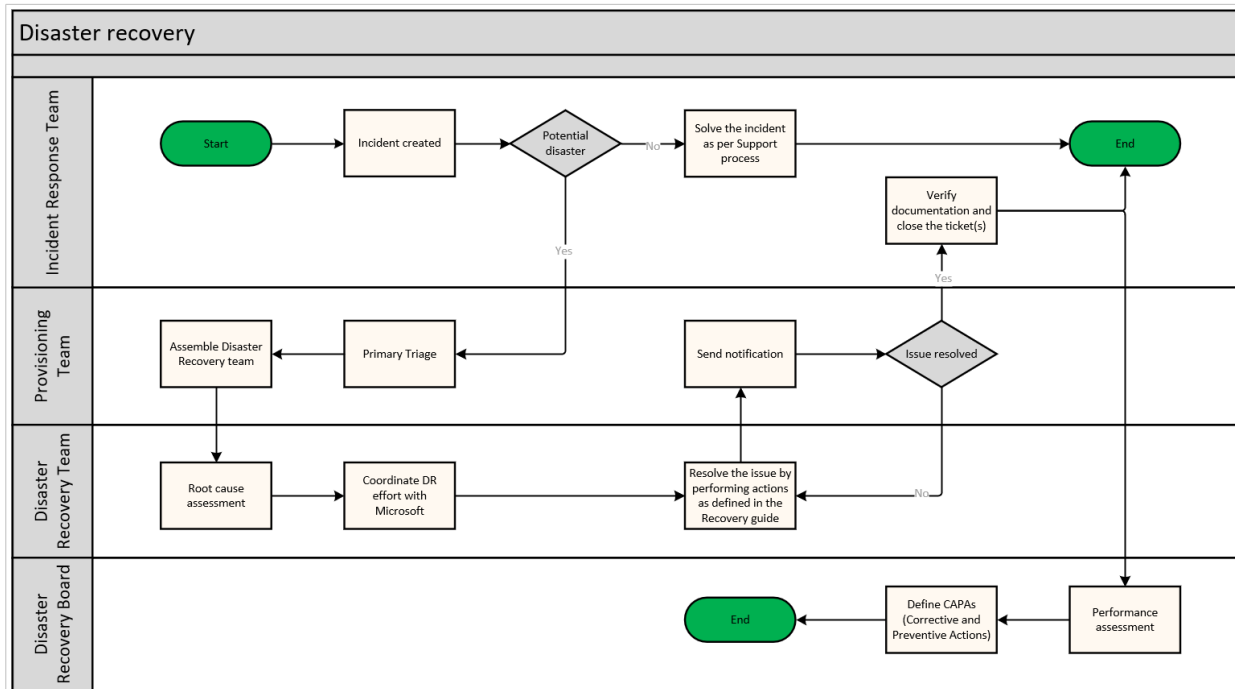
2.4.3. Incident start

Incidents begin when our incident response team receives information about issues affecting NiceLabel Cloud.

This information may come from:

- Monitoring system alerts

- Customer support requests (phone or email)

- Other events indicating potential NiceLabel Cloud problems

We track incidents with support tickets following standard support procedures.

2.4.4. Incident response

Incident response teams handle incidents. Response includes:

1. Incident assessment (reviewing alerts, customer reports).

2. Decision point. After investigation, teams decide whether or not to escalate incidents to disasters.

    a. Incident response teams consult with the provisioning team as needed.

    b. If incidents do not require disaster responses, teams resolve incidents according to standard support processes.

Incident handling and response times follow standard support procedures determined by your SLA level.

2.4.5. Escalation to disaster

Incident response teams contact the provisioning team to trigger disaster recovery responses. Our provisioning team assembles a disaster recovery team to oversee disaster recovery processes.

2.4.6. Disaster recovery

Teams log all status updates in our internal system to ensure visibility for all teams involved. Teams add the [Disaster] keyword to all related support requests to organize disaster logs.

Our disaster recovery team analyzes problems and determines next steps by following our established disaster recovery procedures:

1. Identify the scale, impact, and root cause of the problem.

2. If the problem is due to underlying Azure cloud infrastructure, make sure Microsoft is solving the problem:

    a. Check Microsoft notifications in the Service Health section.

    b. Open support tickets as needed.

    c. Monitor Microsoft's progress.
        If Microsoft resolves the problem in a timely manner, no additional recovery action is required.

3. If Microsoft does not solve the problem, begin recovery procedures following our recovery guide.

While the disaster recovery team progresses through disaster recovery, we provide affected users with status updates and estimated resolution times.

Following recovery, the disaster recovery team analyses root causes of outages and recommends improvements you can make to prevent future incidents. Your affected users receive reports including service credit notes when applicable.

2.4.7. Recovery times

We're committed to restoring service as soon as possible. Recovery times may vary according to the nature and the scale of the problem. NiceLabel works with Microsoft to resolve issues related to underlying services provided by Microsoft Azure.

2.4.8. Process reviews

Our disaster recovery process management team reviews our recovery processes:

- After each disaster scenario

- Periodically (at least once per year)

- As needed (during planned enhancements, or if deficiencies are found outside the scope of periodic testing)

Our disaster recovery process management team determines if our processes require changes and may delegate implementation to our provisioning team. We notify affected teams about all changes.

2.4.9. Periodic tests

Our disaster recovery process management team periodically tests our disaster recovery processes to ensure correct execution and measure effectiveness. Our teams schedule and perform periodic tests at least once per year following established plans. Teams analyze test results during process reviews.

2.5. Additional resources

Loftware delivers the following documentation on request.

> **NOTE**
> Some documents are subject to signed non-disclosure or Software Maintenance Agreements.

1. **Consensus Assessment Initiative Questionnaire (CAIQ).** Shows compliance with CSA® (Cloud Security Alliance) best practices. CSA STAR™ (CSA Security, Trust, Assurance and Risk) is the industry's most powerful program for security assurance in the cloud. STAR™ encompasses key principles of transparency, rigorous auditing, and standards harmonization.

2. **Microsoft CAIQ document.** NiceLabel Cloud runs on Microsoft Azure and automatically inherits Azure security specifications: https://cloudsecurityalliance.org/star/registry/microsoft/.

3. **Traceability matrixes. Features lists including test cases.** NiceLabel Control Center feature specifications. Automated NiceLabel Cloud product testing ensures there are no software flaws or bugs in released software.

4. **Internal team testing reports.** Showing tests we performed and total numbers of tests ran for each software release.

5. **Penetration testing reports.** Assessments from our third-party security company showing tests performed for NiceLabel Cloud, test time intervals, any vulnerabilities, and their conclusions on software security.

6. **ISO 9001:2015 certificate.** Loftware along with Euro Plus d.o.o., has implemented and maintains a management system that meets the requirements of ISO 9001:2015 standards: https://www.nicelabel.com/resources/files/doc/resources/ISO9001-certificate.pdf.

3. NiceLabel Cloud Data Integration

NiceLabel Cloud gives you multiple ways to print labels with data from your external business applications.

Depending on your data security access policies, you can:

- **Create links to your databases from NiceLabel software.** Connect databases to your labels and forms. Use custom user interfaces you create in NiceLabel PowerForms to print labels.

- **Integrate NiceLabel printing into your business systems.** Initiate label printing from your existing applications. NiceLabel Automation acts as your background print engine for processing data and label images.

3.1. Linking to databases

Business applications you interact with daily organize and store updates, changes, and new information in databases. There are many kinds of databases, but our customers typically use **relational databases** for label data. Relational databases range from simple file-based storage like Microsoft Access or Microsoft Excel files to more robust high-performance SQL databases.

NiceLabel software can read data from any kind of database. You only need:

- Database connections (database drivers).
- Database permissions (usernames and passwords).

Our software gives you multiple ways to interact with your database data. You can:

- Use your data as-is.
- Generate custom views to filter data from single tables.
- Merge data from multiple tables into datasets.
- Create visual database views with graphic design tools or SQL statements.

You can use datasets fields as data sources for label and form objects to create **dynamic objects**. Dynamic objects update the content of labels you print with different items in your datasets.

> **NOTE**
> We recommend creating single database connections for your label printing solutions and configurations, and not for your individual label templates.

3.1.1. Linking to file databases

For simple projects, **file databases** are often easier to use than professional SQL databases. Perhaps you store product information in Microsoft Access databases or Microsoft Excel spreadsheets, or export data

from your business systems into file databases. Both are common practices, but be cautious: using file databases may indicate your company does not have a single source of truth for your data.

NiceLabel Cloud stores your labeling data in your cloud **Document Management System (DMS)**. Your DMS typically stores label templates and images, but your DMS can also store your file databases.

The downside to using file databases is **read-only file access**. You cannot write data to file databases stored in DMS. To update your databases with new data, you must re-upload your files.

Alternatively, you can configure your labeling solutions to load data from **locally available file databases**. By storing Microsoft Access and Microsoft Excel files in local or network-attached storage, you can access your files in read-write mode and modify files directly from NiceLabel software.

3.1.2. Linking NiceLabel Cloud Azure SQL user databases

NiceLabel Cloud Business (and above) subscriptions come with **Microsoft Azure SQL database** access. Microsoft SQL databases are hosted in Microsoft Azure environments. Your NiceLabel Cloud edition determines your database size.

Your Azure SQL database is separate from product databases where NiceLabel Cloud stores your subscription labeling assets and printing history. You have full Azure SQL user database access and can independently manage your database with Microsoft SQL Server Management Studio. Use your Azure SQL database for your own storage needs, including storing product information.

Your company security policies may not grant NiceLabel direct access to your business system databases, but NiceLabel can access your Azure SQL database. **Our customers often export their business system master data into their Azure SQL databases to make this data available to NiceLabel software.**

3.1.3. Linking to external databases

NiceLabel Cloud applications can also access data from other databases if you have the required **database drivers** installed on your computers running NiceLabel software. For example, to use NiceLabel Web printing, install database drivers on your server (if you want your server to initiate database connections) or on your computer (if you want your computer to initiate connections).

NiceLabel works with any Windows database drivers (like ODBC and OLEDB drivers).

3.2. Direct integrations

Cloud-based business systems driving your company processes typically raise one label printing question: **How can your cloud software print labels on your on-premise printers?**

The answer is NiceLabel. Our cloud software communicates with both your cloud business systems and your on-premise printers. We provide information, samples, and configuration steps to enable you to print labels from your cloud-based applications.

You receive sample files to explore various direct integration options:

- NiceLabel Automation configuration (MISX)

- Sample label (NLBL)

- Cloud system mockup website (AngularJS framework). The mockup website demonstrates integration options we explain below.

3.2.1. Cloud trigger printing

Cloud trigger printing, also known as **server-based proxy printing**, allows your cloud applications to communicate with server-based proxies (NiceLabel Automation) which process and print to local print queues available on your servers. **No firewall modification is necessary** to allow data from your cloud applications to pass to NiceLabel Automation. You send your data to APIs in NiceLabel Cloud, which forward your data to NiceLabel Automation securely using Azure Service Bus.

BENEFITS

- **Supports all printer brands/models.**

- **Provides data enrichment** (merging data from various external systems on the same label).

- **Returns label previews**.

REQUIREMENTS

- On-premise installation of NiceLabel Automation (integration system).

- Printer drivers for all printers installed on your server with NiceLabel Automation.

- Network-attached printers. Printing to locally connected printers (like USB printers) is only possible when they are shared in a network.
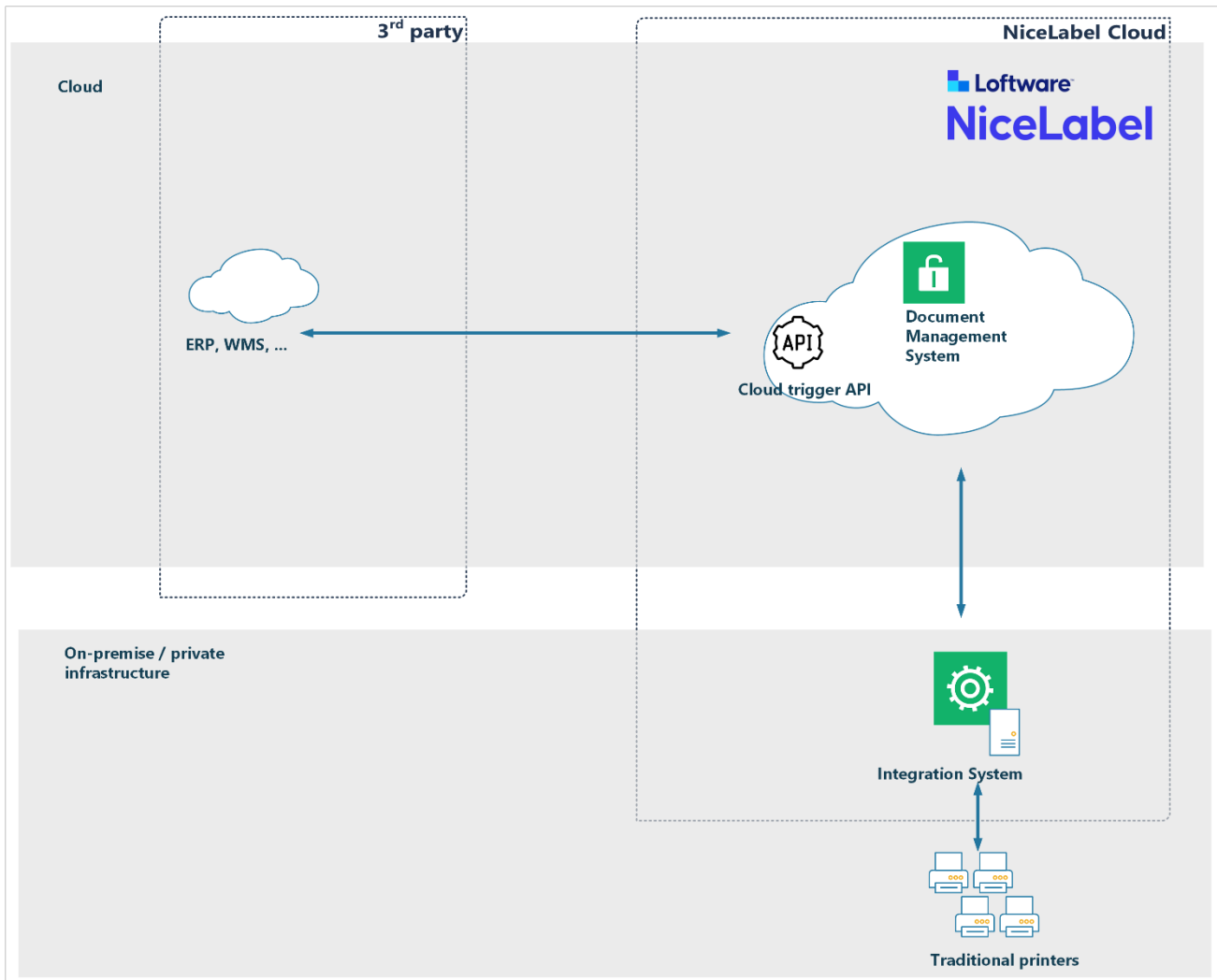
3.2.2. Operation

Cloud trigger integrates NiceLabel Cloud with your existing cloud-based business systems so you can print labels on your local printers. Modern IoT printers communicate directly with the cloud, but legacy printers do not. To print labels on legacy printers, install our on-premise integration system with your printers.

Cloud-based business systems do not communicate with on-premise software and devices. Firewalls protect your local networks from unsolicited requests from the internet.

Your cloud-based business systems produce outputs you send to our NiceLabel cloud-hosted **Cloud trigger APIs** (proxies). Our APIs relay outputs to your on-premise NiceLabel Automation for processing using secure **Azure Service Bus** messages (Microsoft cloud messaging service). You send messages as HTTPS requests using GET or POST methods.

> **NOTE**
>
> Structures of payloads you send to APIs must match your NiceLabel Automation configurations. You can configure Automation to adapt to standard XML/JSON messages your cloud system can already generate.

Your on-premise integration system merges label templates from NiceLabel Cloud DMS (Document Management System) with data received from your cloud business systems to print labels on your local printers.

Cloud Trigger transparently and securely integrates your local label printing with applications that communicate over the open internet located on different networks than your printers.

Workflow:

1. **Send data** from your cloud-based system to NiceLabel API in NiceLabel Cloud (HTTPS requests).

2. Your on-premise integration system **receives data** and processes requests following configurations you define in NiceLabel Automation configuration files.

3. **Your labels print** on local printers. Status responses get send back to your cloud-based system. Responses can also contain label previews or other custom data you need.

3.2.3. Technical details

You make NiceLabel APIs print requests to:

https://labelcloudapi.onnicelabel.com/Trigger/v1/CloudTrigger/{triggerId}

You provide:

- `{triggerId}`: Your Cloud Trigger "unique identifier" (defined in your Automation configuration).

- Custom request header to provide your user credentials.

    - Set Ocp-Api-Subscription-Key to your subscription key.

- Label data in either query strings (GET method) or message bodies (POST method).

The integration system processes the label data you provide and prints your labels. You can choose data input formats including CSV, XML, JSON, etc. You must create matching Automation trigger configurations to parse and process data you send into the trigger.

This integration pack example sends JSON messages to NiceLabel Cloud APIs:

https://labelcloudapi.onnicelabel.com/api/CloudTrigger/Api-CloudIntegrationDemo-Print

Sample outbound JSON message:

```
{
    "FilePath": "/folder/label.nlbl",
    "FileVersion": "",
    "Quantity": "1",
    "Printer": "",
    "PreviewFormat": "PNG",
    "Variables": [
        {
```

```
        "Product_name": "Syringe",
        "GTIN": "00311234567901",
        "LOT": "ABC123",
        "BestBefore": "10.05.22",
        "SSCC": "03831234560000001",
        "Count": "10"
    }
  ]
}
```

This JSON message tells NiceLabel Automation to:

- Create a PNG preview of label.nlbl.

- Use key-value pairs from the Variables object.

This example of NiceLabel Automation configuration works with JSON. However, you can create custom NiceLabel Automation configurations for your specific situation. Custom configurations adapt to your existing application data structures.

For more information about configuring and running integration samples, read **Section 3.3, "Appendix A: Integration bundle"**.

For more information about API definitions, read **Section 3.4, "Appendix B: Cloud trigger API"**.

3.2.4. IoT printing with Cloud API

IoT printing with Cloud APIs, also known as **direct printing**, allows your cloud applications to print directly to your cloud-enabled printers. IoT printing with Cloud APIs **does not require any local NiceLabel footprint or any locally installed NiceLabel software** (like NiceLabel Automation or printer drivers).

BENEFITS

- No local NiceLabel footprints.

REQUIREMENTS

- Cloud-enabled printers which can connect to the NiceLabel Cloud service. For example, you can print with any Zebra Link-OS printer or SATO CL4NX / CL6NX printers.

  > **NOTE**
  >
  > Some Zebra printers with **Link-OS Basic** don't support Cloud Print (for example, Zebra ZD230). Check your printer's specification for the printer operating system. See also the list of Zebra DNA printers. **Print DNA Basic** printers don't support Cloud Print.

- Reliable internet connections.
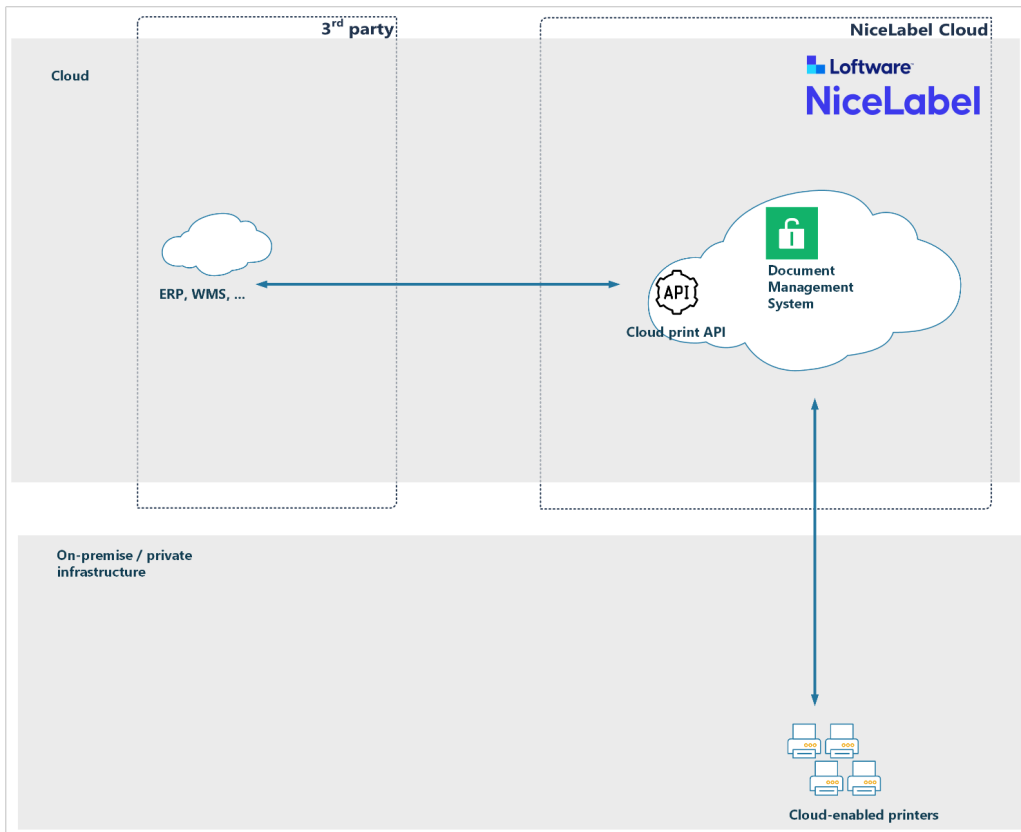
3.2.5. Operation

Cloud printers are smart label printers that connect to our NiceLabel Cloud print service and receive print jobs. Cloud printing lets you **print from any applications or devices** to cloud-connected printers, regardless of printer locations. Cloud printing also **eliminates the need for printer drivers**. Your Cloud Print service creates and delivers print jobs to your target printers via the cloud.

The diagram above shows cloud printing architecture. Printers register and connect to our NiceLabel Cloud print service. When you submit print requests, your cloud service merges label templates from DMS with your received data to create matching print jobs (for example, ZPL for Zebra printers, SBPL for SATO printers, etc.). Your cloud service delivers your print jobs over the internet to requested printers.

3.2.6. Technical details

Cloud Print APIs become available when you enable the Cloud Print add-on for your NiceLabel Cloud subscription. Cloud Print APIs expose multiple HTTP REST methods for print integration. To use these methods, **register your cloud-enabled printers in NiceLabel Cloud** so they are visible to the service.

For example, the GET method "Printers" retrieves a list of all your registered printers and their live statuses. The POST method "Print" prints your selected labels from DMS to your cloud printers with your cloud application data.

You can provide API payloads as JSON or as XML data.

APIs process your provided label data and print your labels. Your labels always load from DMS.

You make NiceLabel API Print requests to:

https://labelcloudapi.onnicelabel.com/Print/v1/Print/{printerName}

You provide:

- `{printerName}`: Your cloud-connected printer name (defined at printer registration). For your list of registered printers, execute the **GetPrinters method**.

- Custom request header to provide your user credentials.

    - Set Ocp-Api-Subscription-Key to your subscription key.

- Label data in either JSON or XML payloads.

Your JSON payloads can look like this:

```
{
        "FilePath": "/folder/GS1-128.nlbl",
        "FileVersion": "",
        "Quantity": 1,
        "Variables": [
            {
                "Product_name": "Syringe",
                "GTIN": "00311234567901",
                "LOT": "ABC123",
                "BestBefore": "10.05.22",
                "SSCC": "03831234560000001",
                "Count": "10"
            },
        ],
        "PrinterSettings": ""
    }
```

For more information about API definitions, visit our Developer Portal.

For more information about NiceLabel Cloud printer registration and Developer Portal integration, read Cloud Printers.


3.2.7. Thin client printing

With thin client printing, also known as **client-based proxy printing**, your cloud applications communicate with local proxies (NiceLabel Web Clients) which process and print labels to local print queues available on your computers.

BENEFITS

- **Supports all printer brands/models**.

- **Performs data enrichment** (merging data from various external systems on the same label).

- **Processes locally** on your computers.

- **Prints to all local or network printers** with drivers installed.

- **Displays label preview**s in your Web client.

REQUIREMENTS

- Requires local NiceLabel footprint– you install small thin clients on each computer.
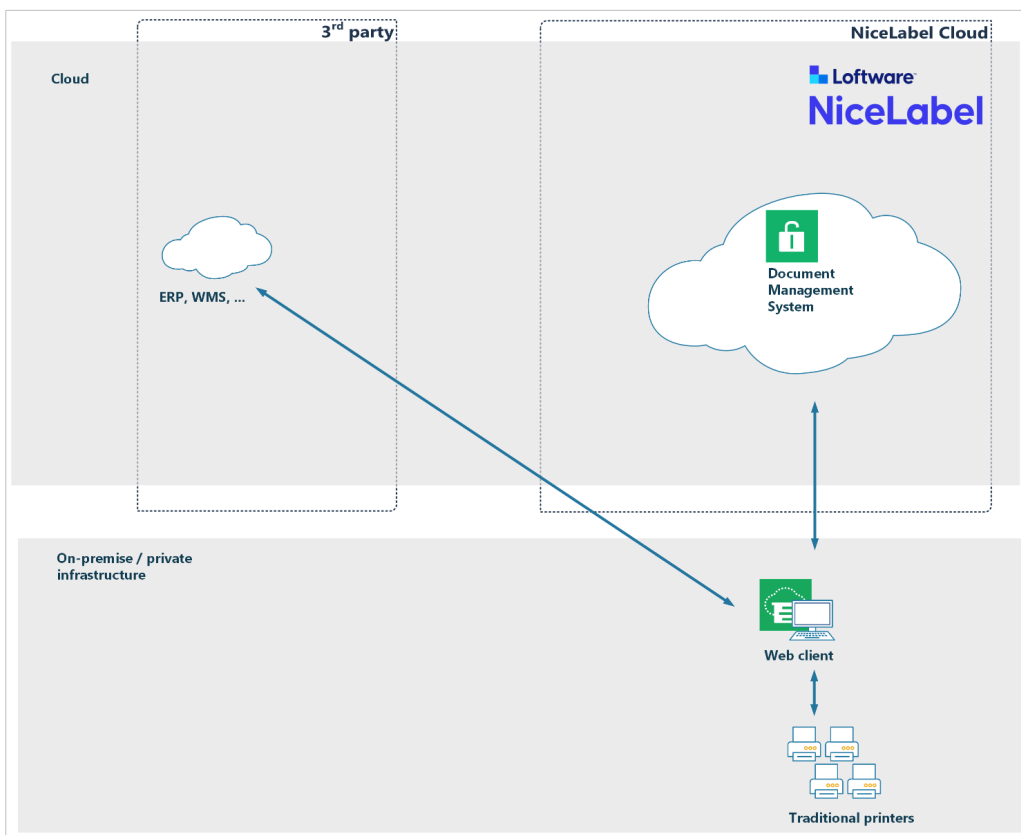
3.2.8. Operation

Your cloud-based business systems invoke your NiceLabel Web Client, which runs your custom label-printing Web application. You invoke your Web application from cloud applications via **URI requests** with label data provided in your query strings. Your Web application uses your locally installed printer drivers to print your business system data on your labels.

You create custom Web application user interfaces in NiceLabel Designer and run your custom Web applications as PowerForms solutions. The sample Web application user interface we include in your integration pack is minimal and only contains printer selection options.

3.2.9. Technical details

NiceLabel Web Client registers custom URI schemes on your computer. URI technology allows cloud-based applications to start desktop applications and send them data.

In the following example, your Web application allows you to select default label printers. You can customize application user interfaces with features you need, including label preview displays.

Cloud-based systems call with URI methods like this:

```
nicelabelwebclient10:?server=https://<account>.onnicelabel.com/
print&openMode=2&application=<applicationname>&variable=Var1=123&variable=Var2=A
BC
```

You provide:

- `<account>` your NiceLabel Cloud account name.

- `<applicationname>` your Web application name (defined in Control Center).

- `openMode=2` when you receive new Web Client requests, your running Web application receives new variable values.

- `variable=Var1=123` sets variable Var1 to value **123**.

For more information, read Adjusting Web Printing.

For more information about your integration sample, read **Section 3.3, "Appendix A: Integration bundle"**.

3.2.10. SFTP file drops

SSH File Transfer Protocols (SFTP) provide **secure file transfer capabilities**. Your cloud applications use SFTP to drop files over the internet. Connections between clients and FTP servers are encrypted so passwords and other sensitive information can be securely transferred over the network.

NiceLabel Automation is configured to monitor folders, while SFTP stores delivered files. Your file system must be mounted locally and visible to NiceLabel Automation. When files arrive, NiceLabel Automation picks them up for processing.

BENEFITS

- **Supports all printer brands/models**.

- **Provides data enrichment** (merging data from various external systems on the same label).

- **Returns label previews**.

REQUIREMENTS

- On-premise installation of NiceLabel Automation (integration system).

- Printer drivers for all printers installed on your server with NiceLabel Automation.

- Network-attached printers. Printing to locally connected printers (like USB printers) is only possible when they are shared in a network.

- NiceLabel Automation must be able to access the file system where files from SFTP are stored.

3.2.11. Alternative on-premise integrations

When your business systems share local networks with NiceLabel integration systems, you have additional integration options for delivering label printing data to NiceLabel.

NiceLabel Automation accepts data the following ways:

- **File drops.** Your applications save data into selected local or network folders. You can use data formats supported by your applications to create matching configurations and parse data in NiceLabel Automation (typically XML, JSON or CSV structures, but you can also use others).

- **Serial ports.** Data comes from devices connected with RS-232 serial ports like barcode scanners and weight scales.

- **Databases.** NiceLabel Automation monitors specified databases for changes. When Automation detects new records, your labels print automatically. NiceLabel can also monitor your business system databases directly with your authorization. Master data is often exported from business systems into intermediate databases which NiceLabel can access.

- **TCP/IP sockets.** Allows raw data connections from network devices. NiceLabel can operate in server or client modes.

- **HTTP servers.** Receives data from clients sending HTTP messages (native web application communication method). Messages are usually JSON-formatted, but NiceLabel supports all other data structures.

- **Web Services.** Receives SOAP messages from clients. The underlying protocol is HTTP, and messages are XML-formatted.

3.3. Appendix A: Integration bundle

Your integration bundle includes a mockup application to demonstrate available types of integrated printing with Cloud Applications. We simulate your cloud-based applications with a JavaScript mockup application. Your mockup application can send printing data to:

- **Cloud Print APIs** to demonstrate direct printing with cloud-enabled printers.

- **Cloud Trigger APIs** to demonstrate server-based printing with Cloud Triggers.

- **NiceLabel Web Client** to demonstrate thin client printing.

We provide you with sample files in **LabelCloudDataIntegrationPack.zip**. You can download it here:

For NiceLabel 10: https://ftp.nicelabel.com/software/demo/v10/LabelCloudDataIntegrationPack.zip

For NiceLabel 2019: https://ftp.nicelabel.com/software/demo/v8/LabelCloudDataIntegrationPack.zip

Your sample is configured for NiceLabel Cloud's "demosystem" account. Follow the steps in the following sections to update your sample application to work with your NiceLabel Cloud account.

3.3.1. Configuring Control Center

3.3.2. Uploading files

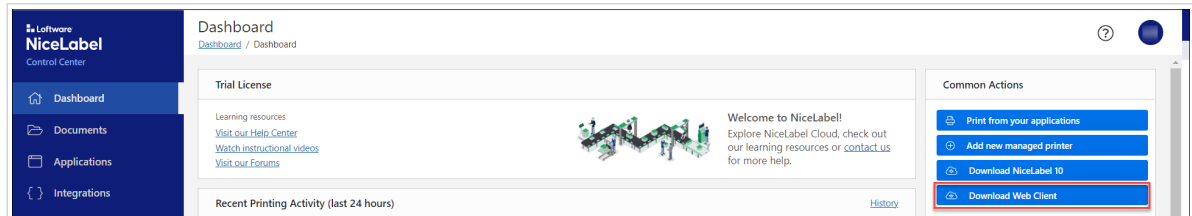1. Log in to your **NiceLabel Cloud Control Center**. In your browser, type:

    ```
    https://<account>.onnicelabel.com/dashboard
    ```

    > **NOTE**
    >
    > Replace `<account>` with your actual NiceLabel Cloud account name.

2. Download and install **NiceLabel Web Client**.

3. In **Documents**, create a new folder: **/Demo/LabelCloudDataIntegration**.

4. Open **LabelCloudDataIntegrationPack.zip** (provided).

5. Upload your **Document Storage** folder contents into your **CloudIntegration** folder (in your DMS).
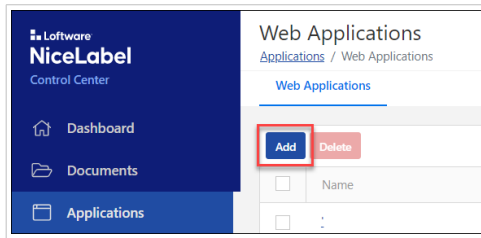
Your files are uploaded in Control Center and ready to use.

3.3.3. Registering Web Applications

In **Web Applications,** you define which applications NiceLabel Web Client can run. We also use Web Applications for "thin client" printing demos.

Register your Web Application:

1. Go to **Applications** > **Web Applications** (on your left) and click **Add**.

2. For **Name**, type **CloudIntegration-ThinClient**.

3. For **Path** (Solution or Label files), click **Browse** and select: **/Demo/LabelCloudDataIntegration/ ThinClient.nsln**.

4. For **Authorized Users and Groups**, add at least one user (yourself). Your users receive email invitations and all users you authorize can run the Web Application in NiceLabel Web Client.
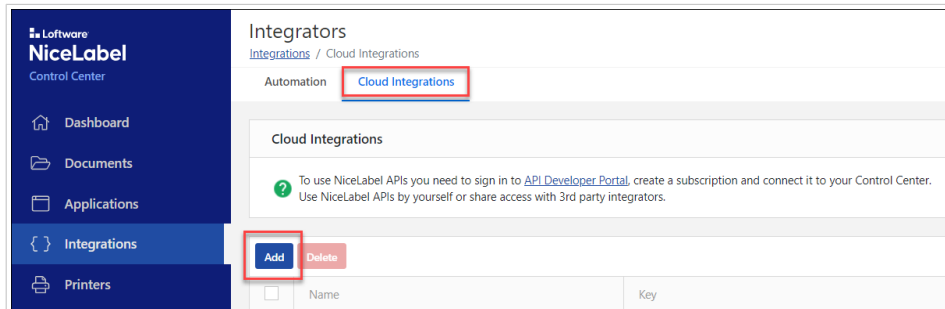
5. Click **Save**.

Your Web Application is registered and ready to use.

3.3.4. Creating Cloud Integrations

Cloud Integrations you create in Control Center authenticate your **Cloud Print APIs** and **Cloud Trigger APIs**. You use the same Cloud Integrations for **IoT print using Cloud API** and **Cloud trigger** printing demos.

Create your Cloud Integration:

1. Go to **Integrations** > **Cloud Integrations** (on your left) and click **Add**.

2. For **Name**, type a custom name for your integration (you can type any name).

3. Click **Save**.

Your Cloud Integration is created in Control Center and ready for Developer Portal registration.

3.3.5. Registering Developer Portal Cloud Integrations

Before you can use Cloud Print APIs or Cloud Trigger APIs, you must register your **Cloud Integration** in the **Developer Portal**. Registering links your NiceLabel Cloud account with APIs and requires subscription keys to authorize your API calls.

Register your Cloud Integration:

1.   Open your Control Center User Guide.

2.   Read the chapter **Cloud Print API**.

3.   Follow the provided instructions to complete the following:

     a.   Set up your new account on our Developer Portal.

     b.   Create your new subscription.

     c.   Register your Developer Portal subscription with your Cloud Integration (in Control Center).
          You must use your activated subscription key in custom HTTP headers (**Ocp-Apim-Subscription-Key**) with each of your API calls.

Your Cloud Integration is registered on the Developer Portal and ready for printer registration.

3.3.6. Registering cloud printers

Before you can execute Cloud Print APIs, you must register your cloud-enabled printers with your NiceLabel Cloud account.

For instructions and more information about the registration process, read our user guide:

•   Chapter **Understanding Cloud Print** in Control Center User Guide.
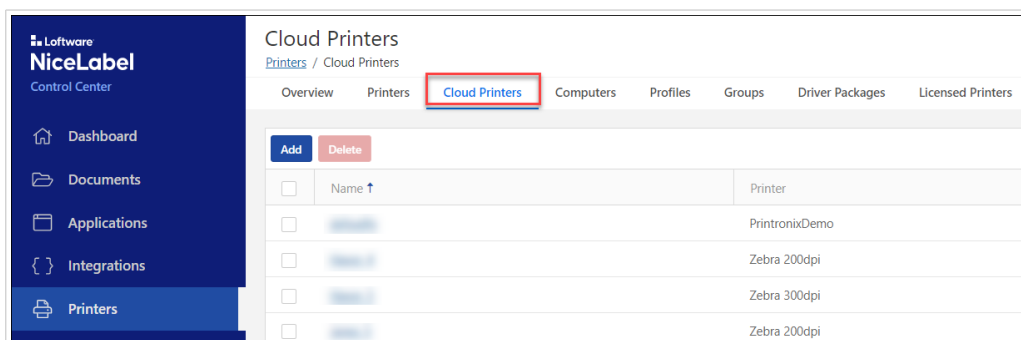
3.3.7. Configuring your mockup website

1.   Open **LabelCloudDataIntegrationPack.zip** (provided).

2.   Extract the **Website** subfolder to a temporary disk location.

3.   Open your extracted Website folder.

4.   Open **index.html** in your text editor.

5.   Find the string "nicelabelwebclient". Update the URI reference:

```
nicelabelwebclient10:?server=https://demosystem.onnicelabel.com/print
```

> **NOTE**
>
> Replace "demosystem" with your NiceLabel Cloud account name.

6. Save your file.

7. Open **scripts/config.js** in your text editor.

8. Type your subscription key as your **subscriptionKey** value.

> **NOTE**
>
> You can get your **subscriptionKey** from **Developer Portal** > **Products**.

> **NOTE**
>
> Do not change values for **label** and **uniqueIdentifier**:
>
> - **Label** specifies which label file to use.
>
> - **UniqueIdentifier** specifies unique cloud trigger identifiers (defined in your Automation configuration).

9. Save your file.

Your mockup website is set up and ready to use.

3.3.8. Configuring your integration server

1. Run **NiceLabel Automation Manager**.

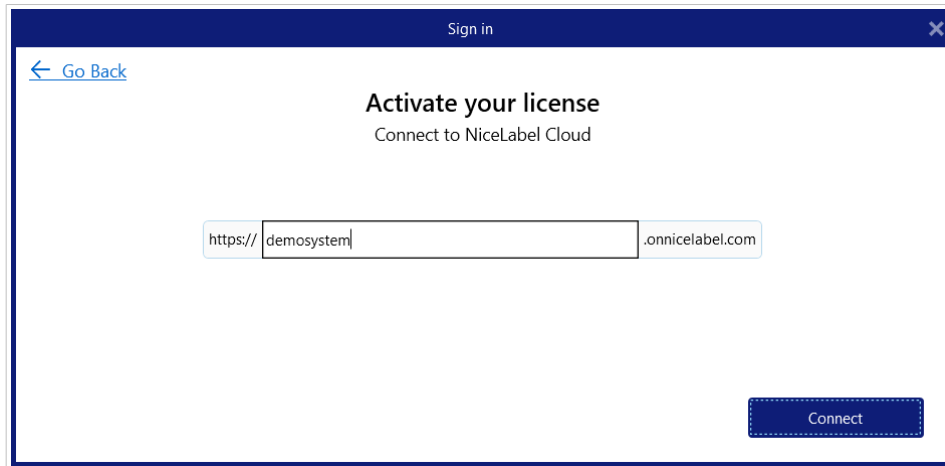2. Click **Connect** under "Connect to NiceLabel Cloud" to begin activation.
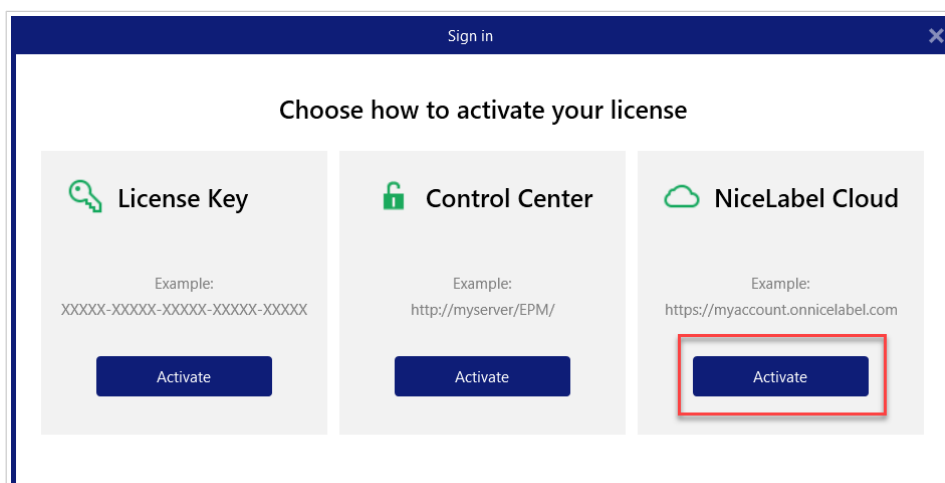
3. Type your NiceLabel Cloud account name.

> **NOTE**
>
> Replace "demosystem" in the screenshot below with your actual NiceLabel Cloud account name.

4. Click **Connect**.

5. Select Microsoft or Google authentication (depending on how you logged in to Control Center from your invitation email). NiceLabel activates on your computer.

6. In Automation Manager, click **+Add.** Browse to your Automation configuration file (.MISX) in Document Storage (Load the file you copied to **/Demo/LabelCloudDataIntegration**).
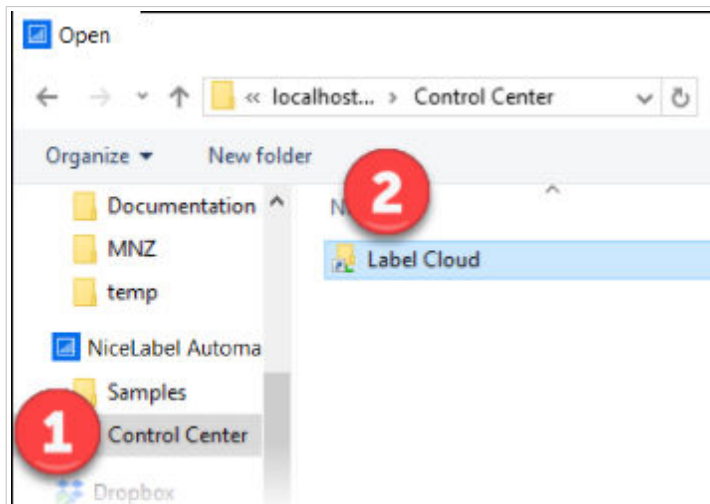
7. Start all the triggers in your configuration.
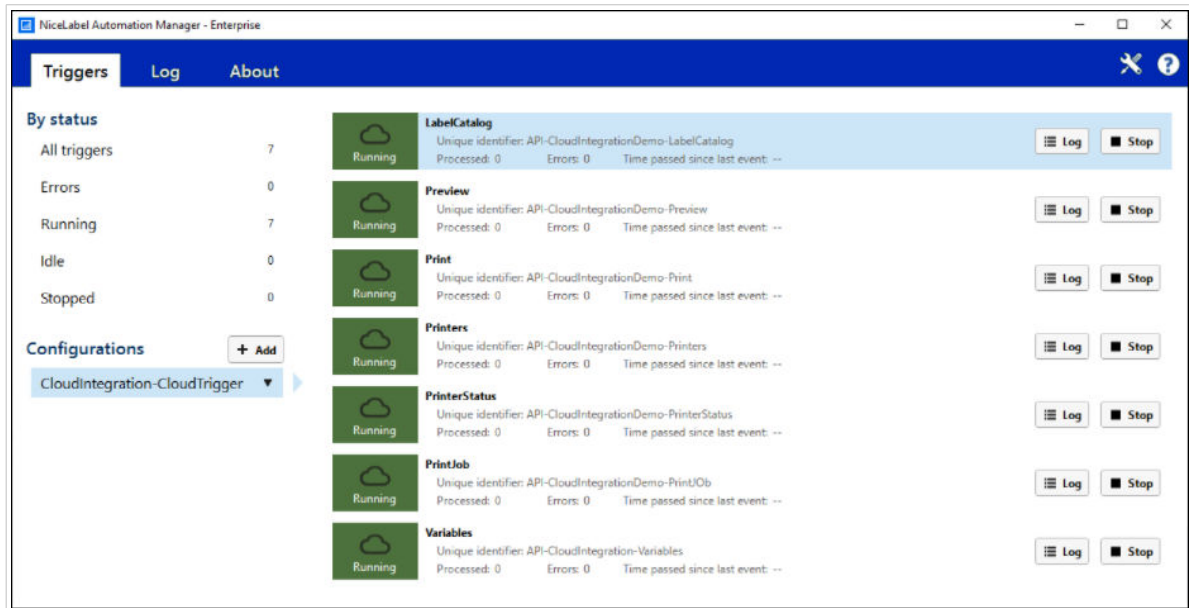
Your integration server is set up and ready to use.

3.3.9. Configuring your Web Client

Configure your NiceLabel Web Client for "thin client" printing:

1. Open **ThinClient.nsln** in NiceLabel Designer.

2. Select the "question mark" Picture object.

3. Double-click the "question mark" Picture object to open object properties.

4. In **Events**, click **Actions** for **On Click** event.

5. In the **Open Document/Program** action, direct your file name to your mockup website URL.

> **NOTE**
>
> If your mockup website runs from your local disk and not through a web server, you can delete the **Open Document/Program** action altogether.

6. Save your changes.

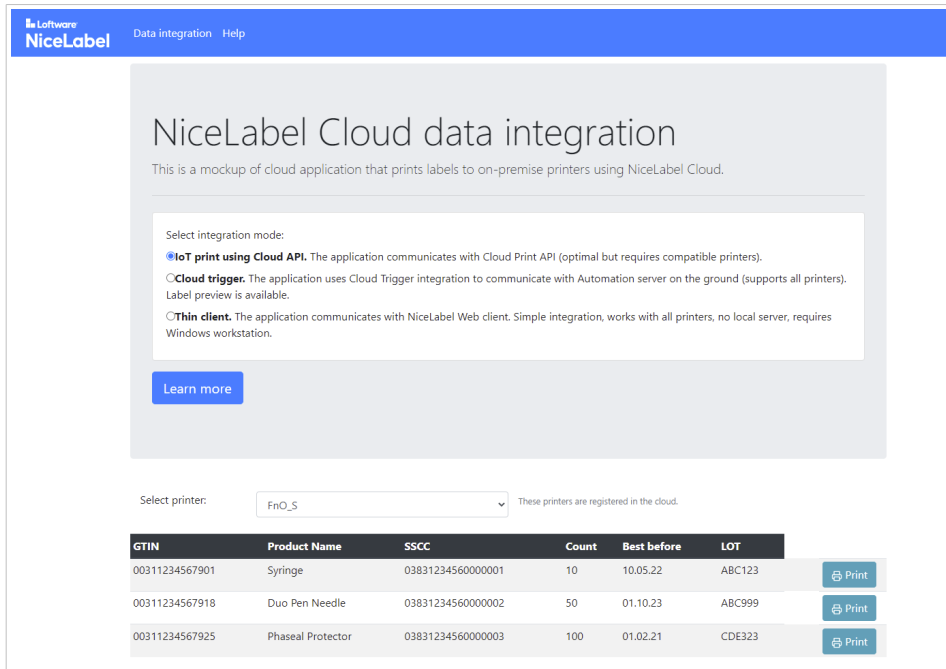NiceLabel Web Client is ready for "thin client" printing.

3.3.10. Executing sample integrations

Execute your sample integration:

1. Go to the temporary folder on your disk with your extracted files from **LabelCloudDataIntegrationPack.zip**.

2. Go to your **Website** folder.

3. Double-click **index.html**. Your cloud system mockup opens in your browser:

Your browser console logs outbound XML requests and inbound responses. To view your logs:

1. Open your browser.

2. Press **F12**.

3. Click **Console**.

3.3.11. IoT printing with Cloud API

For each Print request, your mockup website writes and sends JSON messages as HTTP requests to our **Cloud Print API**. Cloud Print API processes the XML data, creates print jobs, sends print jobs to cloud-connected printers, and provides feedback.

Execute cloud printing:

1. For your integration method, select **IoT print using Cloud API**.

2. Select your printer from the list.

> 📓 **NOTE**
>
> Your list dynamically displays printers registered to your NiceLabel Cloud account. If your list is empty, register your printers. See **Registering cloud printers**.

3. Click **Print** to print labels.

Your cloud printing is set up and ready to use.

3.3.12. Cloud triggers

For each Print or Preview request, your mockup website writes and sends XML messages as HTTP requests to our **Cloud Trigger API**. Cloud Trigger API redirects the messages to your on-premise NiceLabel Automation. Automation processes the XML data, executes the required actions (such as print or preview), and provides feedback.

Execute your server-based integration:

3. Click **Print** to print labels.

4. Click **Preview** to see label previews.

> **NOTE**
> Your list dynamic displays printers available on the computer where your NiceLabel integration system (NiceLabel Automation) is installed.

5. For your integration method, select **Cloud trigger**.

6. Select your printer from the list.

Your server-based integration is set up and ready to use.

3.3.13. Example thin clients

Execute your client-based integration:
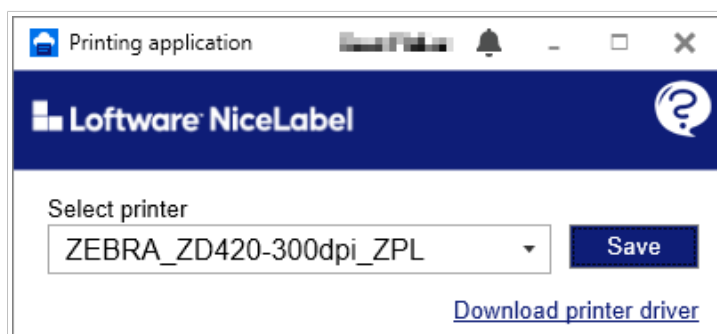
1. For your integration method, select **Thin client**.

2. Click **Print** to open **NiceLabel Web Client**.

> **NOTE**
> You must have Web Client installed on the computer where your mockup web application runs.

3. Log in to the Web Client with your Microsoft or Google account (Select **Remember me** to save your credentials). Web Client opens your Web printing application:

4. Choose a printer from the drop-down list to print your labels.

5. Click **Save** to remember your printer.

6. Keep your Web application running.

7. Click **Print** in your mockup application to transmit product data to your Web Client in a query string. Your labels print to your local printer.

8. If you have no driver installed, click **Download printer driver** and install your driver.

Your client-based integration is set up and ready to use.

3.4. Appendix B: Cloud trigger API

To enable Cloud trigger printing, you use an integration system (NiceLabel Automation) installed on-premise to receive and process data from your business systems. NiceLabel Automation exposes APIs you can connect to using Cloud trigger APIs from your cloud business applications.

In addition to label printing, with APIs you can:

- Get label previews (PDF, PNG or JPEG)

- Report live statuses of label printers

- Generate lists of all labels in your Document Management System (DMS)

- Get lists of available printers

- Get lists of variables defined in your label

- Provide print jobs (binary)

> ❗ **IMPORTANT**
> **You are not required to use or follow these samples. These instructions are based on the sample NiceLabel Automation configuration provided in your integration package. You are free to modify your configuration or create your own from scratch.**

3.4.1. Executing Cloud trigger APIs

Each Cloud trigger is identified by a unique name. You must reference the unique name in each HTTP request to our Cloud trigger API.

In the example below, replace the "unique_name_of_the_Cloud_trigger" with the actual unique Cloud trigger name defined in your NiceLabel Automation configuration (.MISX file).

Each Cloud trigger defined in your configuration exposes one API method. You use different triggers for each method. One trigger for the PRINT method, another for the PREVIEW method, and yet another for the PRINTERS method, etc. Unique names tell Automation which trigger it must execute.

When you load and deploy/activate your Automation configuration, Automation registers the Cloud triggers (unique names) in your NiceLabel Cloud account. You can access the triggers with Cloud trigger APIs. When you execute Cloud trigger APIs, your NiceLabel Cloud account knows where your registered Automation trigger is running and forwards data to your trigger.

You can have multiple on-premise NiceLabel Automation servers running the same configuration at the same time with automatic round-robin load balancing enabled.

URL	https://labelcloudapi.onnicelabel.com/Trigger/v1/CloudTrigger/<unique_Cloud_Trigger_name>

You must include the following custom HTTP header with each request:

Ocp-Apim-Subscription-Key	<Subscription key is generated on the Developer Portal when you register "Clound Integration" developer from the Control Center>

You must use the POST method and include the JSON or XML data in your message body.

3.4.2. Registering as integrators

To execute Cloud trigger APIs, you must register as a developer and get a subscription key. You can complete this self-service operation once you can sign in to your NiceLabel Cloud account and when you have set up your account in our Developer Portal.

For more information about the registration process, read our user guides:

- Chapter **Cloud Integrations** in NiceLabel Control Center User Guide.

- Chapter **Cloud Trigger** in NiceLabel Automation User Guide.

3.4.3. API methods

Our API methods help you test and understand the structures of messages transferred between your business applications and NiceLabel Automation. We include examples and descriptions of inbound requests and NiceLabel Automation feedback structures.

You can use JSON and XML data structures with our API methods.

For more information about incoming data payloads, see the chapter **Data structures**.

3.4.4. PRINTERS method

This method retrieves lists of available printers NiceLabel Automation can use. Your lists contain names of printer drivers installed on the computer where NiceLabel Automation runs.

You must send an empty data payload for the API to determine which type of response you need.

Request URL

https://labelcloudapi.onnicelabel.com/Trigger/v1/CloudTrigger/CloudTrigger/Api-CloudIntegrationDemo-Printers

JSON data payload

```
{}
```

XML data payload

```
<?xml version="1.0" encoding="utf-8"?>
```

3.4.5. PRINT method

This method prints your labels. In data payloads, you provide:

- Label names

- Label versions

- Printer names

- All label data to print

Request URL

https://labelcloudapi.onnicelabel.com/Trigger/v1/CloudTrigger/Api-CloudIntegrationDemo-Print

JSON data payload

```
{
    "FilePath": "/folder/label.nlbl",
    "FileVersion": "",
    "Quantity": "1",
    "Printer": " ZEBRA R-402",
    "PrinterSettings": "",
    "PrintJobName": "",
    "Variables": [
        {
            "Name": "FIELD1",
            "Value": "Value1"
        },
```

```json
        {
            "Name": "FIELD2",
            "Value": "Value2"
        }
    ]
}
```

XML data payload

```xml
<?xml version="1.0" encoding="utf-8"?>
<PrintData>
    <FilePath>/folder/label.nlbl</FilePath>
    <FileVersion></FileVersion>
    <Quantity>1</Quantity>
    <Printer>ZEBRA R-402</Printer>
    <PrinterSettings></PrinterSettings>
    <PrintJobName></PrintJobName>
    <Variables>
        <Variable Name="FIELD1">Value1</Variable>
        <Variable Name="FIELD2">Value2</Variable>
    </Variables>
</PrintData>
```

3.4.6. LABELCATALOG method

This method returns lists of available label templates in specific folders (and subfolders) from your DMS.

Request URL

https://labelcloudapi.onnicelabel.com/Trigger/v1/CloudTrigger/Api-CloudIntegrationDemo-LabelCatalog

JSON data payload

```json
{
    "CatalogRoot": "/folder/subfolder",
    "SubscriptionKey": "your_subscription_key"
}
```

XML data payload

```xml
<?xml version="1.0" encoding="utf-8"?>
<PrintData>
    <CatalogRoot>/folder/subfolder</CatalogRoot>
    <SubscriptionKey>your_subscription_key</SubscriptionKey>
</PrintData>
```

3.4.7. PREVIEW method

This method returns label previews in your required format. Like the PRINT method, you provide label names, versions, and data values. Supported preview formats include PDF, JPG, and PNG.

Request URL

https://labelcloudapi.onnicelabel.com/Trigger/v1/CloudTrigger/Api-CloudIntegrationDemo-Preview

JSON data payload

```
{
    "FilePath": "/folder/label.nlbl",
    "FileVersion": "",
    "Quantity": "1",
    "Printer": "",
    "PrinterSettings": "",
    "PreviewFormat": "",
    "Variables": [
        {
            "FIELD1": "Value1",
            "FIELD2": "Value2"
        }
    ]
}
```

XML data payload

```
<?xml version="1.0" encoding="utf-8"?>
<PrintData>
    <FilePath>/folder/label.nlbl</FilePath>
    <FileVersion></FileVersion>
    <Quantity>1</Quantity>
    <Printer></Printer>
    <PrinterSettings></PrinterSettings>
    <PreviewFormat></PreviewFormat>
    <Variables>
        <Variable Name="FIELD1">Value1</Variable>
        <Variable Name="FIELD2">Value2</Variable>
    </Variables>
</PrintData>
```

3.4.8. PRINTJOB method

This method returns your labels converted to print jobs for target printers. For example, if your target printer is Zebra, your content returns in ZPL (Zebra Programming Language). You provide label names, versions, and data values like the PRINT method.

Request URL

https://labelcloudapi.onnicelabel.com/Trigger/v1/CloudTrigger/Api-CloudIntegrationDemo-PrintJob

JSON data payload

```json
{
    "FilePath": "/folder/label.nlbl",
    "FileVersion": "",
    "Quantity": "1",
    "Printer": "",
    "PrinterSettings": "",
} "Variables": [
    {
        "FIELD1": "Value1",
        "FIELD2": "Value2"
    }
]
```

XML data payload

```xml
<?xml version="1.0" encoding="utf-8"?>
<PrintData>
    <FilePath>/folder/label.nlbl</FilePath>
    <FileVersion></FileVersion>
    <Quantity>1</Quantity>
    <Printer></Printer>
    <PrinterSettings></PrinterSettings>
    <Variables>
        <Variable Name="FIELD1">Value1</Variable>
        <Variable Name="FIELD2">Value2</Variable>
    </Variables>
</PrintData>
```

3.4.9. VARIABLES method

This method returns lists of the variables defined in your specified labels. You may need to know the variables defined in your label so you can assign them values when you execute the PRINT method.

Request URL

https://labelcloudapi.onnicelabel.com/Trigger/v1/CloudTrigger/Api-CloudIntegrationDemo-Variables

JSON data payload

```
{
    "FilePath": "/folder/label.nlbl",
    "FileVersion": "",
}
```

XML data payload

```xml
<?xml version="1.0" encoding="utf-8"?>
<PrintData>
    <FilePath>/folder/label.nlbl</FilePath>
    <FileVersion></FileVersion>
</PrintData>
```

3.4.10. PRINTERSTATUS method

This method returns live printer statuses reported by your printers. Not all printers are capable of reporting their statuses.

Request URL

https://labelcloudapi.onnicelabel.com/Trigger/v1/CloudTrigger/Api-CloudIntegrationDemo-PrinterStatus

JSON data payload

```
{
    "Printer": "ZEBRA ZD420-300dpi ZPL"
}
```

XML data payload

```xml
<?xml version="1.0" encoding="utf-8"?>
<PrintData>
    <Printer>ZEBRA ZD420-300dpi ZPL</Printer>
</PrintData>
```

3.4.11. Data structures

Your API configuration accepts JSON and XML data payloads. Your configuration automatically determines your payload type and responds using the same type.

3.4.12. Incoming data payloads

The data structures below include all possible fields. Fields you use depend on your API method. See the chapter **API Methods** for a list of fields applicable to your particular API method.

JSON payload structure

```json
{
    "FilePath": "/folder/label.nlbl",
    "FileVersion": "",
    "Quantity": "1",
    "Printer": "",
    "PrinterSettings": "",
    "PrintJobName": "",
    "PreviewFormat": "",
    "CatalogRoot": "",
    "SubscriptionKey": "",
    "Variables": [
        {
            "Variable1": "Value1",
            "Variable2": "Value2"
        }
    ],
    "Report": [
        {
            "Pos": "01",
            "Code": "100",
            "Name": "Product 1"
        },
        {
            "Pos": "02",
            "Code": "200",
            "Name": "Product 2"
        }
    ]
}
```

You can use the optional JSON Report object when you print shipping documents like packing slips and delivery notes.

Items within JSON Report objects are converted into structures suitable for Report objects on your label templates. For example, the variables "Pos", "Code" and "Name" from the example are used in the Report object with the same names. You can also use your own variable names. If you don't intent to use Report objects in your label template, you can skip the JSON Report object.

XML payload structure

```xml
<?xml version="1.0" encoding="utf-8"?>
<PrintData>
    <FilePath>/folder/label.nlbl</FilePath>
    <FileVersion>1</FileVersion>
    <Quantity>1</Quantity>
    <Printer></Printer>
```

```xml
<PrinterSettings></PrinterSettings>
<PrintJobName></PrintJobName>
<PreviewFormat></PreviewFormat>
<CatalogRoot></CatalogRoot>
<SubScriptionKey></SubscriptionKey>
<Variables>
    <Variable Name="Variable1">Value1</Variable>
    <Variable Name="Variable2">Value2</Variable>
</Variables>
<Report>
    <Item>
        <Pos>01</Pos>
        <Code>100</Code>
        <Name>Product 1</Name>
    </Item>
    <Item>
        <Pos>02</Pos>
        <Code>200</Code>
        <Name>Product 2</Name>
    </Item>
</Report>
</PrintData>
```

Field descriptions

NAME	DESCRIPTION
FilePath	Full path name and label name from Document Storage.
FileVersion	Label file version. If you use major/minor versioning, type your version as "0.1" or "1.1". If you use major versioning, type your version as "1" or "2". When you do not type values, the latest available revision is used. If you use approval workflows, the latest approved label version is used.
Quantity	Number of labels to print.
Printer	Printer name where your labels print. This must be the exact printer name installed on your Windows computer with NiceLabel Automation.
PrinterSettings	Base64-encoded DEVMODE structure. Use to override printer settings saved in your label template or recalled from your printer driver. **NOTE** You can create PowerForms applications to configure printer settings and export settings to DEVMODE. You can also export settings to DEVMODE from your printer driver with an application you can download from our website (Downloads > Utilities and Support Software).

NAME	DESCRIPTION
PrintJobName	Job file names that display in Windows Spooler. If omitted, your job name is the name of your label file by default.
PreviewFormat	Label preview format. Available formats include PDF, PNG, and JPEG. If not specified, PDF is your default format.
CatalogRoot	Starting folder where you create your label catalog. Your catalog also includes labels from all subfolders. **NOTE** If you request label catalogs from large Document Storage repositories, it may take several moments to complete your request.
SubscriptionKey	A subscription key to generate the label catalog. Automation configuration consumes the Document API to generate the label catalog and needs a subscription key for authentication. You can get your Subscription Key from the Developer Portal.
Variables	Name-value pairs for your label variables. For each "name", you must provide a matching "value". You send all name-value pairs to your label to use with variables defined with the same "name". Value mapping takes place automatically based on matching names.
Report	Multiple items providing name-value pairs for the variables used in the Report object in the label template. Each "item" is used as a new row in the Report object. This is an optional field.

3.4.13. Feedback messages

Automation provides data processing results to business applications in synchronous responses. Responses always match incoming data payload structures. When you send JSON-formatted requests, you receive JSON responses. When you send XML-formatted requests, you receive XML responses.

- **Successful execution.** If there are no data processing errors, business systems receive JSON or XML-formatted messages. Your response structure depends on on the API method you execute.

- **Erroneous execution.** If there are processing errors like "label not found", "printer not available", or "wrong data for label objects", your NiceLabel Automation trigger raises an error and sends back details in the trigger response.

For XML data payloads, the API provides feedback in the HTTP header and in the message body. For JSON data payloads, responses contain HTML messages without error details (An upcoming NiceLabel release will feature JSON responses with error details).

You can see detailed error logs in **NiceLabel Automation Manager** for each response type.

PrinterList Structures

Each printer has **Printer** elements.

NAME	DESCRIPTION
PrinterName	Windows printer driver name visible to NiceLabel Automation. Windows users can change this name.
DriverName	Windows printer model name (defined by printer driver). Windows users cannot change this name.

Variable Structures

Each variable on your label contains **Variable** elements.

For more information, read our user guide:

- Chapter **Get Label Information** in NiceLabel Automation User Guide.

NAME	DESCRIPTION
Name	Variable name.
Description	Variable description.
DefaultValue	Default value defined for your variable during label design.
Format	Acceptable types of variable content (characters).
IsPrompted	Shows if the variable is prompted at print time.
PromptText	Text that prompts your users for value input.
CurrentValue	Actual value used for printing.
IncrementType	Shows if the variable is defined as a counter. If identified as a counter, **IncrementType** shows the counter type.
IncrementStep	Counter step information. Your counter value increases/decreases by this value on your next label.
IncrementCount	Counter value increment/decrement information. Counters usually change values for each label, but you can choose custom increments.
Length	Maximum number of stored characters in the variable.
IsPickListEnabled	Shows if your users select variable values from a pick list.
PickListValues	Actual (selectable) pick list values.

LabelCatalog Structures

Below is just a sample structure returned for the Label Catalog.

- The Automation configuration for NiceLabel 10 generates the label catalog using Document API from the Developer Portal (you have to sign in to see the content). The sample uses a small subset of what the Document API provided.

- The Automation configuration for NiceLabel 2019 uses a CLI (command-line interface) helper application, which is limited in its functionality.

> **NOTE**
>
> You can change the returned label catalog structure to fit your requirements. Just edit the trigger that generates the label catalog in the provided Automation configuration.

The sample structure that NiceLabel 10 configuration provides is the following:

NAME	DESCRIPTION
Id	Internal id of the label templates as generated in the Document Management system.
ItemPath	The full path and filename to the label template.
Created	Creation timestamp.
Modified	Modification timestamp.
ModifiedBy	Id of the user that modified the label template last.
VersionNumber	The version of the most recent label template.
WorkflowName	The name of a workflow defined for the folder in which the label template is saved.
WorkflowStepName	The name of the workflow step that the label template occupies.

The structure that NiceLabel 2019 configuration provides is the following:

NAME	DESCRIPTION
Name	Label name, including path from Document Storage root folder. Server names and port numbers are not included; NiceLabel Automation knows DMS locations.
Width	Label width in 1/1000 millimeters.
Height	Label height in 1/1000 millimeters.
PrinterName	The printer your label is designed for. > **NOTE** > You can specify different printers for printing. NiceLabel Automation adapts label templates for your new printer types.
Description	Label description you can configure.
Revisions	Comma-delimited list of revisions accessible to NiceLabel Automation. > **NOTE** > Role-Based Access Control (RBAC) determines label visibility. When you run NiceLabel Automation as an Operator (best practice), you can only access approved label revisions.

4. Microsoft Dynamics 365 Supply Chain Management Integration

> **NOTE**
>
> This integration works with:
>
> - **Microsoft Dynamics 365 Supply Chain Management** (**D365SCM**)
> - **Microsoft Dynamics 365 Finance**
> - **Microsoft Dynamics 365 for Finance and Operations** (previous name)
> - **Microsoft Dynamics 365 for Finance and Operations Enterprise Edition** (previous name)
>
> We refer to all these editions as **D365SCM**.
>
> This integration **does not work** with:
>
> - **Microsoft Dynamics 365 for Finance and Operations Business Edition**

Learn to integrate NiceLabel printing in **Microsoft Dynamics 365 Supply Chain Management** (**D365SCM**) with **NiceLabel Cloud**. Customize code from our sample integration on GitHub to enable NiceLabel printing directly from D365SCM in your specific printing environment. You download and deploy our sample integration to D365SCM Lifecycle Services like other integrations. You can customize your integration code to include your printer setup information and other D365SCM data to use on labels you print.

NiceLabel Cloud is a cloud-based label management system that lets you print on cloud and classic printers.

- **Cloud printers** are smart devices that connect to NiceLabel Cloud with no software installation required. NiceLabel Cloud prints on cloud printers using **Cloud Print REST APIs** with predefined request formats.

- **Classic printers** require installation on Windows computers. NiceLabel Cloud prints with NiceLabel Automation. Install NiceLabel Automation on your computer to handle data processing and printing for classic printers. You can trigger printing with **Cloud Trigger REST APIs** in any request format. Your Automation configuration determines how to process requests.

Cloud printers	Classic printers
Only NiceLabel Cloud enabled printers.	All Windows supported printers.
No on-premises footprint.	On-premises Windows computer with NiceLabel 10 and printer drivers installed.

Cloud printers	Classic printers
Predefined data processing and printing.	Automation configuration for customized data processing and printing.

Your integration connects to NiceLabel Cloud and prints from D365SCM with REST APIs. NiceLabel Cloud combines data from REST API calls with label templates stored in NiceLabel Cloud to create complete labels you print on cloud or classic printers you choose:
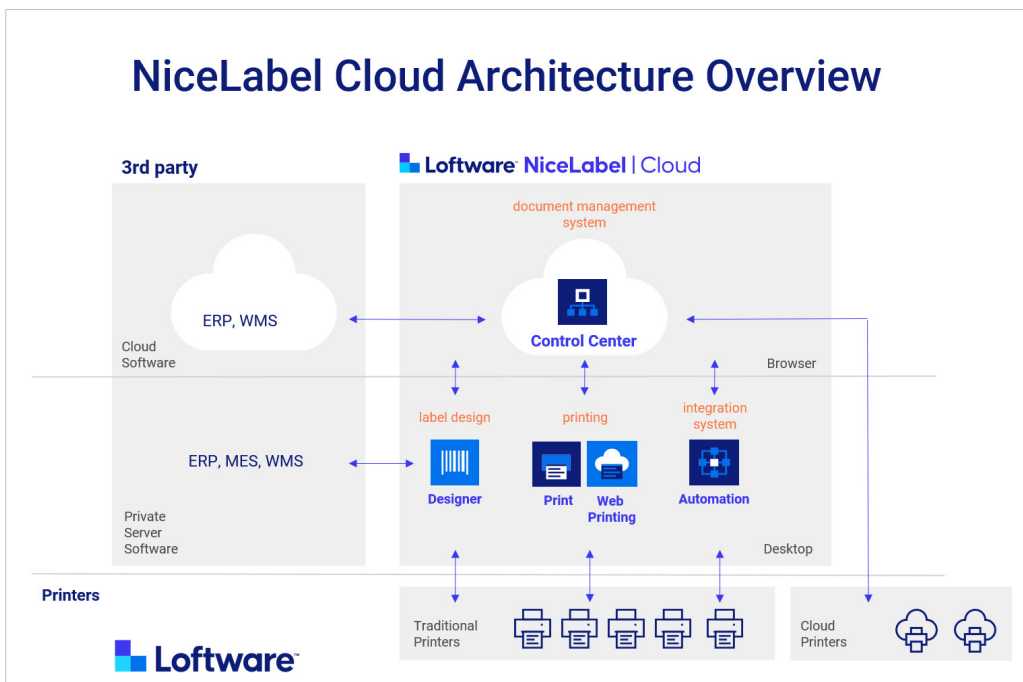
How D365SCM integrations work.

4.1. NiceLabel setup

4.1.1. 1. Get your NiceLabel Cloud account

To integrate NiceLabel cloud printing from D365SCM, you need a NiceLabel Cloud account. Contact sales to get a license or trial account.

When you get your account, install **NiceLabel 10** on your computer.

Connecting cloud printers requires no software, but you need NiceLabel 10 to design and upload new label templates.

Go to NiceLabel **Control Center** > **Dashboard** > **Common Actions** and click **Download Nicelabel 10**.

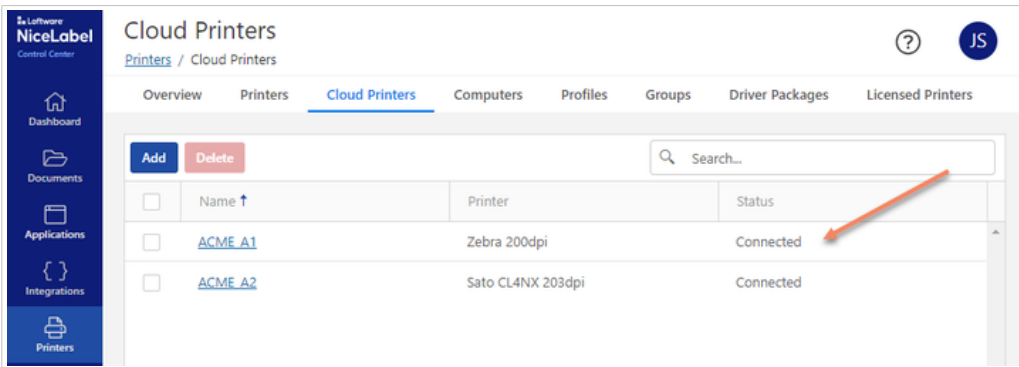## 4.1.2. 2. Connect printers to your NiceLabel Cloud account

From **Control Center**, you can connect:

- Cloud printers supported by NiceLabel Cloud **or**

- Classic printers on Windows computers

4.1.2.1. Connecting cloud printers

Connect your cloud printers to NiceLabel Cloud so you can print from D365SCM with Cloud Print REST APIs.

For more information, read https://help.nicelabel.com/hc/articles/4407466158737-Cloud-Printers.

*Adding a cloud printer named ACME_A1.*

4.1.2.2. Connecting classic printers

1. Install your printers with NiceLabel printer drivers on the same computer where you install NiceLabel 10. **This computer acts as a server for your NiceLabel printing environment and should remain on and connected.** For more information, read https://help.nicelabel.com/hc/en-001/categories/4408474153489-Printer-Driver-Installation.

2. To see your list of installed printers, on your computer, press **Windows+R** and run **printmanagement.msc**.
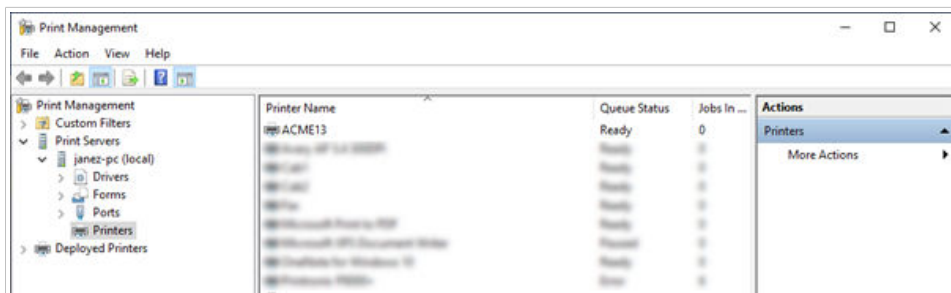
   *Your list of printers installed on your Windows computer in **printmanagement.msc**.*

3. Download `LabelCloudDataIntegrationPack.zip`. Extract the zip file and open the `Document Storage` folder. Run the `CloudIntegration-CloudTrigger.misx` file, deploy the configuration and start triggers to allow Cloud Trigger REST APIs to use printers installed on your computer.
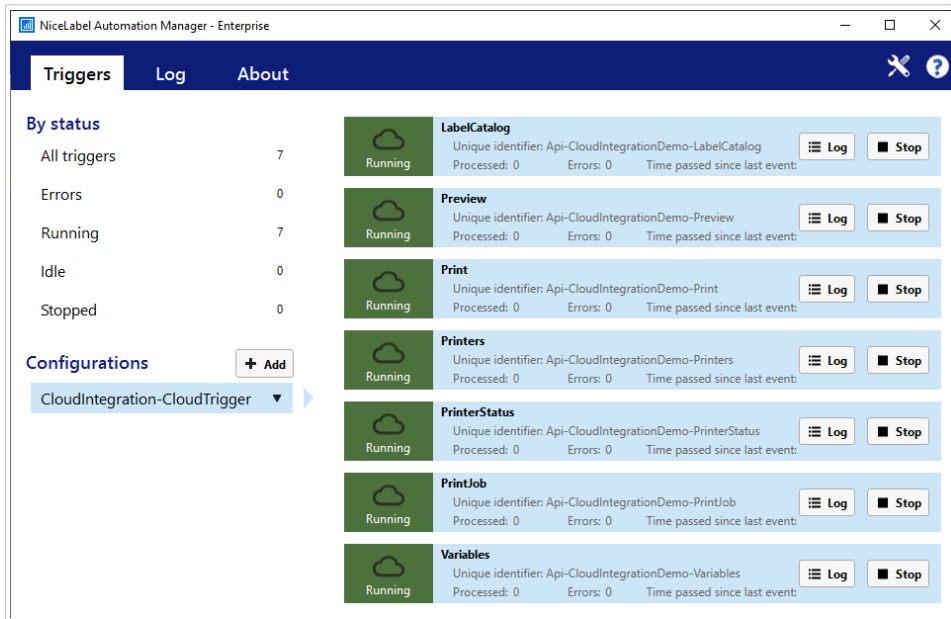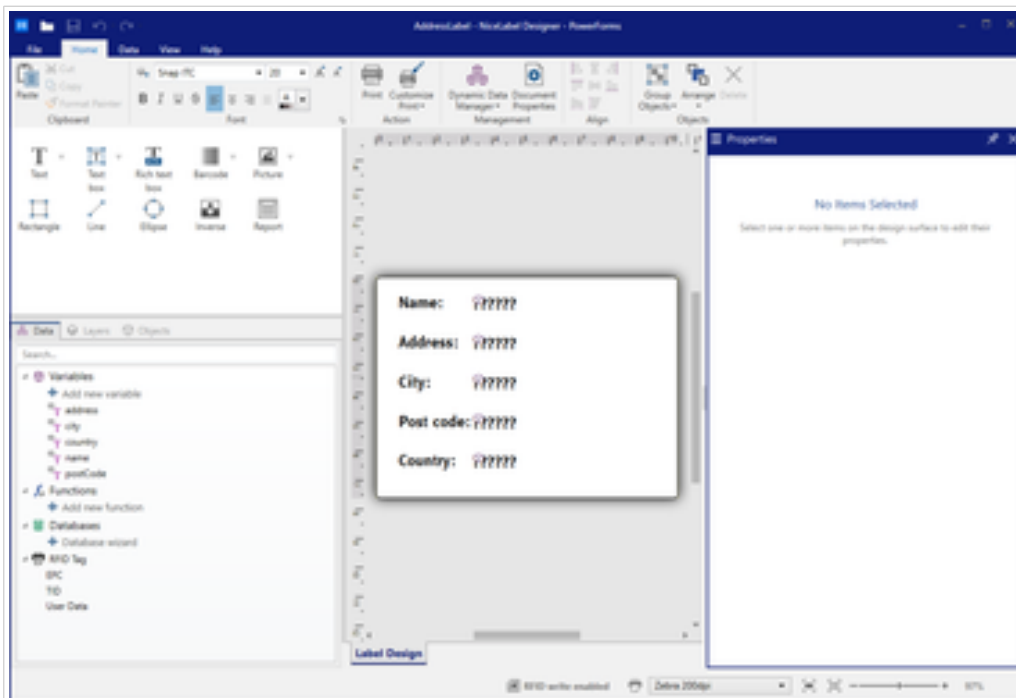
*Running **CloudIntegration-CloudTrigger.misx** in Automation Manager.*

To enable Cloud trigger printing, see the Cloud trigger API topic.

For more information, read:

- Cloud Trigger topic
- Integration bundle topic.
- Cloud trigger API topic

4.1.3. 3. Create your label template

*Creating a dynamic label template in NiceLabel Designer.*

Label templates you create in NiceLabel Designer can use variables connected to data from D365SCM. Our sample integration connects data from your D365SCM global address book to create dynamic address labels from templates stored in NiceLabel Control Center.

To print address labels from D365SCM with our sample integration:

1. Create a label template in NiceLabel Designer. For more information, read https://help.nicelabel.com/hc/articles/4406561608081-Label.

2. Add variables on your template to use with data from your D365SCM global address book. For more information, read https://help.nicelabel.com/hc/articles/4402152654097-Variables.

3. Name your label template **AddressLabel.nlbl**.

4. Save **AddressLabel.nlbl** in **Control Center** > **Documents > Labels**. For more information, read https://help.nicelabel.com/hc/articles/4402320800401-Documents.

Your saved label template is ready to open in your sample integration and print from D365SCM.

4.1.4. 4. Enable REST APIs
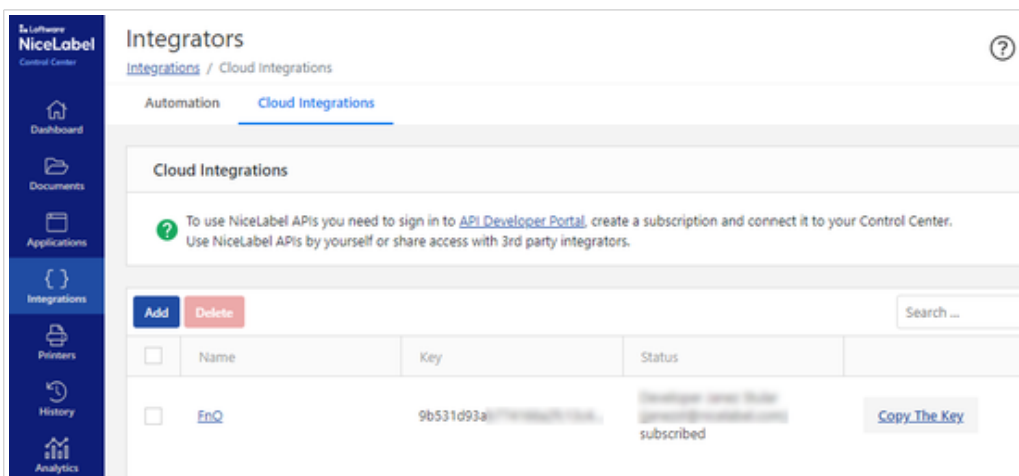
Adding a new integration in Control Center.

To perform REST API calls and print labels from D365SCM, you need to connect your NiceLabel developer portal subscription to **Control Center**:

1. Go to **Control Center** > **Integrations** > **Cloud Integrations**.

2. **Add** a new **Cloud Integration** for D365SCM.

3. Sign up on the developer portal.

4. On the developer portal, add a new product subscription and connect it to your NiceLabel Cloud account. Use this subscription with your primary D365SCM production environment.

5. Open **Control Center** > **Integrations** > **Cloud Integrations** again and make sure your integration is connected.

For full setup instructions and more information, read:

- [Cloud Printers](#)

- [Cloud Integrations](#).

4.1.5. 5. Test Print

On the developer portal, test print your labels on your connected printers with REST API calls. If your test prints don't succeed, check your configuration and try again before you set up your D365SCM sample integration.

- For **cloud printers**, call the print function with printer names at https://developerportal.onnicelabel.com/docs/services/cloud-print/operations/put-printfunction.

- For **classic printers**, call the trigger function with the trigger name **Api-CloudIntegrationDemo-Print** at https://developerportal.onnicelabel.com/docs/services/cloud-trigger/operations/put-cloudtriggerfunction.

You can change your printer and file path values to actual values and test print a label with a JSON request:

```
{
    "Printer":"ACME_A1",
    "FilePath":"/Labels/AddressLabel.nlbl",
    "Quantity":1,
    "Variables":[
        {
            "name":"A. Datum Corporation",
            "address":"123 Main Street",
            "city":"Sacramento",
            "postCode":"94279",
            "country":"United States"
        }
    ]
```

With your cloud integration configured correctly in NiceLabel Cloud, you can begin your D365SCM integration setup.

4.2. D365SCM sample integration

Name:	A. Datum Corporation
Address:	123 Main Street
City:	Sacramento
Post code:	94279
Country:	United States

Printed address label.

Our sample integration adds a **NiceLabel printing integration setup page** and **NiceLabel print button** to your **global address book** page in D365SCM.

Our sample integration allows you to print NiceLabel labels directly from D365SCM using your D365SCM **global address book** data and NiceLabel label templates. The integration uses cloud printers or classic printers with Cloud Trigger configurations and predefined Cloud Print requests. To run our sample integration in D365SCM, upload the deployable package from GitHub to your **Lifecycle Services** interface.

Customize your D365SCM sample integration code for your specific printing environment and requirements. To edit and adapt our sample integration to your own needs, you need a virtual machine with Visual Studio running in your Lifecycle Services interface. To edit source code, you need the Dynamics 365 add-on in your Visual Studio. Click here to read how to import and export D365SCM projects in Virtual Studio. Include your custom printer setup information and other data from D365SCM to use on labels your users print. Add D365SCM data you want to use in variables on your label templates to your integration code.

Before your users can print with our sample integration, configure their D365SCM integrations setup with 3 setup tabs so your users can print selected records using selected labels on selected cloud and classic printers directly from D365SCM.

Our sample integration calls Cloud REST APIs and sends the following data:

- Printer name

- Label template

- Print quantity

- Data from selected records to use on the label template

4.3. NiceLabel Printing Integration setup

Set up our sample integration with the basic data your users need to perform print REST API calls by pressing the **NiceLabel Print** button in D365SCM.

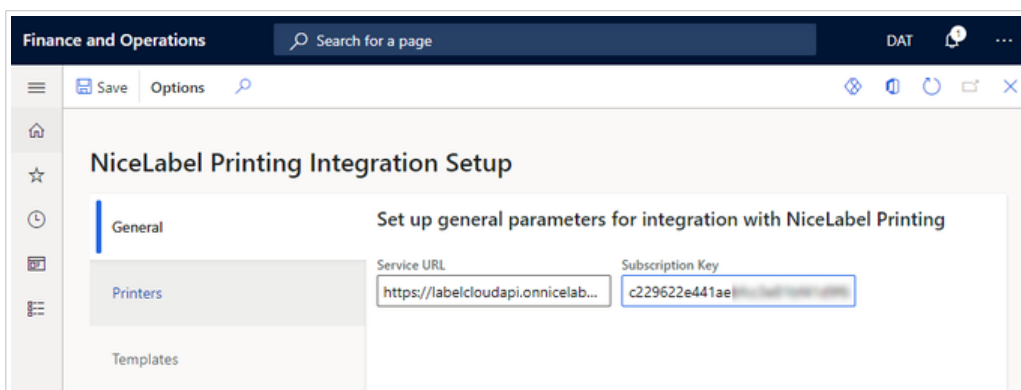In **D365SCM**, go to **System administration** > **Setup** > **NiceLabel Printing Integration**
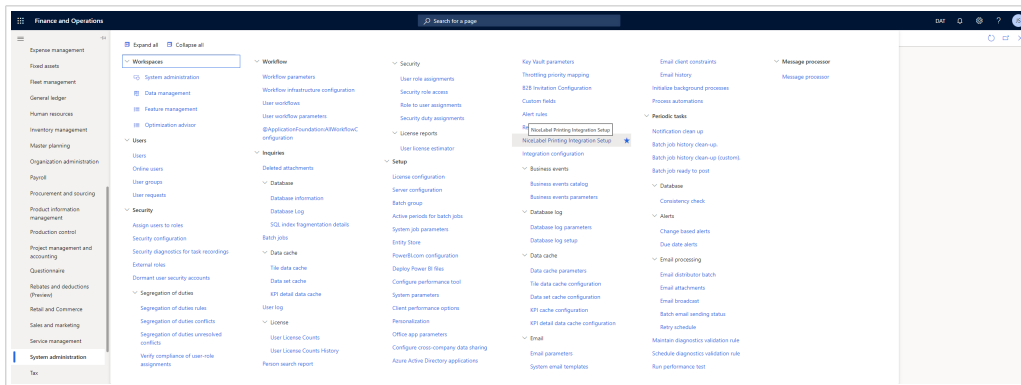
Opening NiceLabel Printing Integration setup in D365SCM.

Configure the following 3 setup tabs:

- General

- Printers

- Templates

Before you begin, carefully configure any data you use to perform print REST API calls.

4.3.1. General tab

The **General** setup tab includes your account information necessary for every REST API call.

- Set **Service URL** to the REST API location https://labelcloudapi.onnicelabel.com.

- Set **Subscription key** to your developer portal subscription key. Use the subscription key for your primary production environment.

4.3.1.1. Code: General tab

The **NLPIntegrationParameters** form saves your parameters into the **NLPGeneralParameters** table.

4.3.2. Printers tab
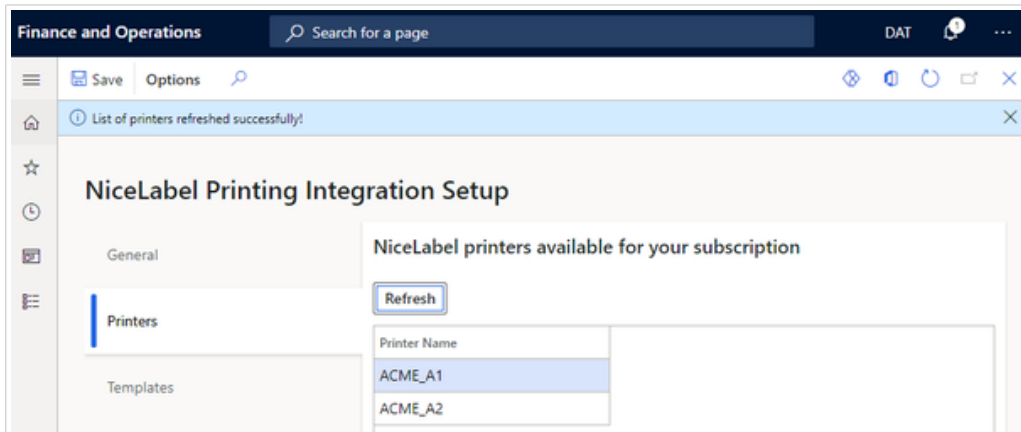
Configuring the Printers setup tab for D365SCM users.

The **Printers** setup tab allows your users to retrieve a list of printers connected to NiceLabel Cloud and makes them available to select on the printing form.
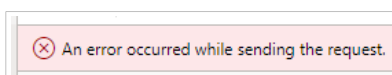
On the Printers tab, click **Refresh** so you can:

- Get an updated list of printers connected to NiceLabel Cloud.

- Save your list of printers to your D365SCM database.

If you configure your **General** tab correctly, you get a successful response:
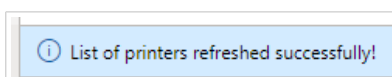
If you configure your **General** tab incorrectly, you get an error:

4.3.2.1. Code: NLPIntegrationParameters

Clicking **Printers** > **Refresh** calls the **NLPIntegrationParameters.RefreshButton_OnClicked** method. This method has 3 main functions:

- Deleting existing printer lists from the **NLPPrinterParameters** table.

- Getting new printer lists from REST APIs.

- Saving new printers list into the **NLPPrinterParameters** table.

Note the **NLPrinting** class in this method. To get lists of connected printers, call its associated **NLGetPrinters** method.

```
public static void RefreshButton_OnClicked(FormControl sender,
FormControlEventArgs e)
{
    // delete previous printers list

    ...

    NLPrinting NLPrint = new NLPrinting();
    System.String[] listOfPrinters = NLPrint.NLGetPrinters();
    int numOfPrinters = listOfPrinters.Length;

    ...

    // save new printers list
}
```

4.3.2.2. Code: NLPrinting class basics

NLPrinting is a C# wrapper for **NLApiClient.dll**. Customize your integration by changing the value of the **useCloudPrinting** parameter to print on cloud printers with Cloud Print API or classic printers connected to NiceLabel Automation with predefined Cloud Trigger API configurations (.misx).

- To use **Cloud printers**, set **useCloudPrinting** to **true**.

- To use **Classic printers**, set **useCloudPrinting** to **false**.

```
class NLPrinting
{
    private boolean useCloudPrinting = true;

    public boolean configValid;
    private str NLPrintServiceURL;
    private str NLPrintSubscriptionKey;
    private NLApiClient.BaseIntegrator obj;
    ...
}
```

The NLPrinting class constructor reads your **Service URL** and **Subscription Key** from the **NLPGeneralParameters** table and makes sure they are present. You save both of these required parameters in the **Integration Setup** > **General** tab. The check validates your instance of cloud integrator class based on the **useCloudPrinting** parameter. Your class instance is responsible for all REST API calls.

```
class NLPrinting
{
   ...
```

```
    void new()
    {
        this.NLPrintGetURLAndKey();

        if(configValid)
        {
            if (useCloudPrinting)
            {
                obj = new NLApiClient.CloudPrintIntegrator(NLPrintServiceURL);
            }
            else
            {
                obj = new
NLApiClient.CloudTriggerIntegrator(NLPrintServiceURL);
            }
        }
        else
        {
            obj = null;
        }
    }
    ...
}
```

4.3.2.3. Code: NLPrinting.NLGetPrinters

You call the REST API based on your **UseCloudPrinting** configuration:

- For **Cloud Printers**, use GET Printers REST API. You can test this call on the developer portal: https://developerportal.onnicelabel.com/docs/services/cloud-print/operations/get-printersfunction

- For **Classic printers**, use PUT Cloud Trigger REST API with the trigger name **Api-CloudIntegrationDemo-Printers**. You can test this call on the developer portal: https://developerportal.onnicelabel.com/docs/services/cloud-trigger/operations/put-cloudtriggerfunction

```
public System.String[] NLGetPrinters()
{
    if (configValid)
    {
        if (useCloudPrinting)
        {
            return (obj as
NLApiClient.CloudPrintIntegrator).GetPrintersArray(NLPrintSubscriptionKey);
        }
        else
        {
            return (obj as
```

```
NLApiClient.CloudTriggerIntegrator).GetPrintersArray(NLPrintSubscriptionKey);
        }
    }
    else
    {
        return null;
    }
}
```
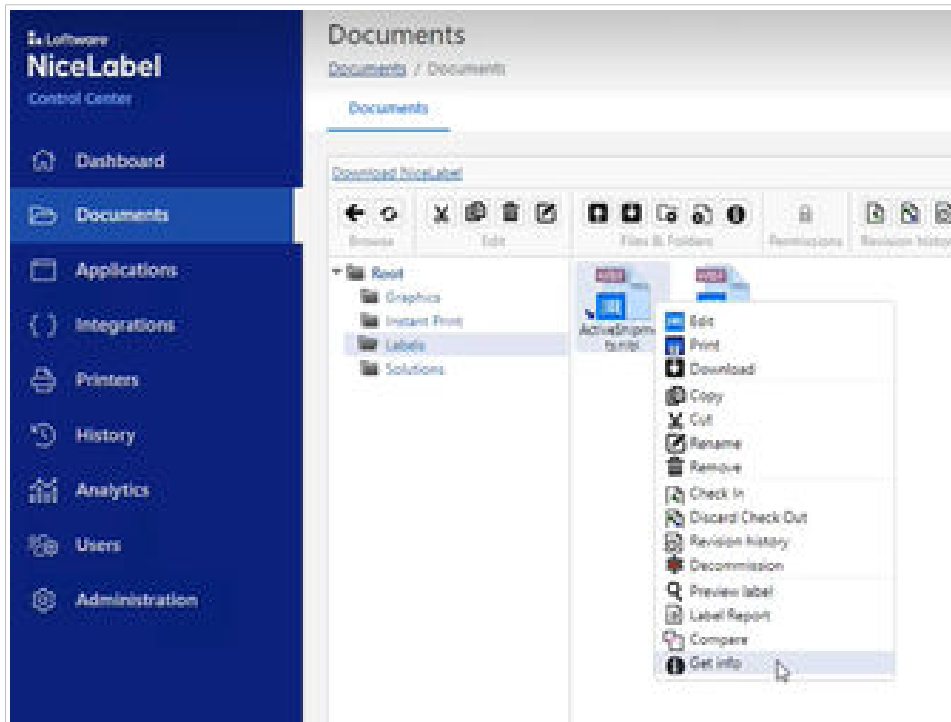
4.3.3. Templates tab

Printing from D365SCM with REST APIs uses label templates stored in **NiceLabel Cloud** > **Documents**. To make selecting templates easier for your users, you create a list of label templates and assign each template a user-friendly name in the **Templates** setup tab.
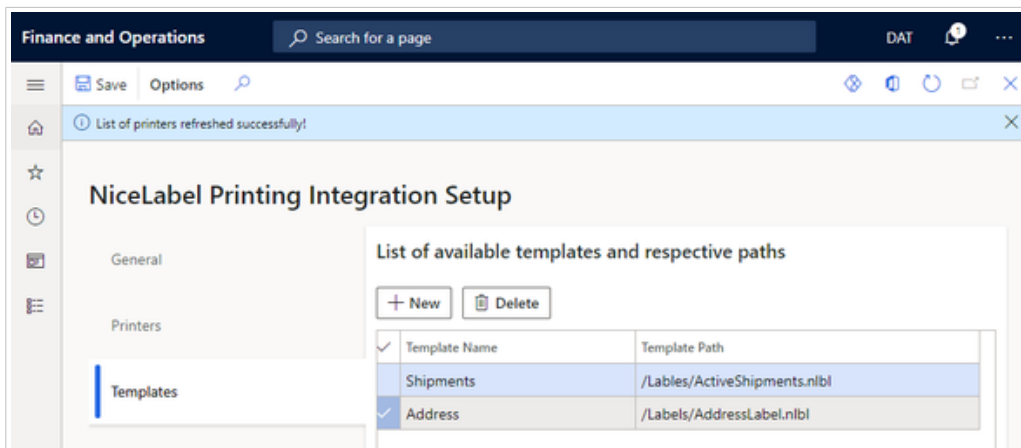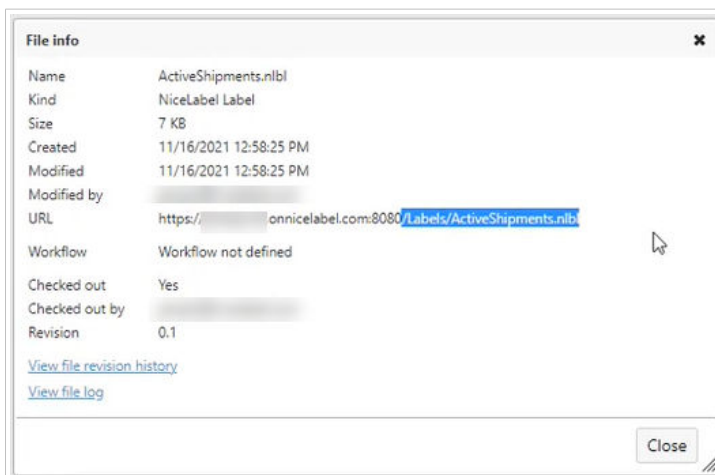

Adding and naming label templates from Control Center to use in D365SCM.

Add label template paths from your files stored in **Control Center** > **Documents**.
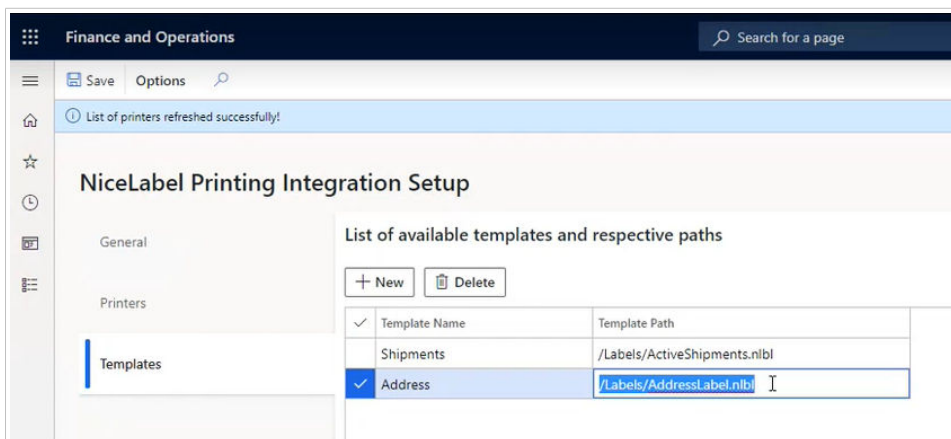
1. Right-click a label template file you want to include and choose **Get info**. The **File info** window opens:

2. Copy the file path after your NiceLabel Cloud URL:

3. Paste the path in the **Template Path** field for a new template in the **D365SCM** > **NiceLabel Printing Integration Setup** > **Templates** tab.

4. Type a unique user-friendly **Template Name** for each template you add so your users know which labels to select and print from D365SCM.

4.3.3.1. Code: templates tab

The **NLPIntegrationParameters** form saves your template paths to the **NLPTemplateParameters** table.

4.4. Printing from D365SCM

Our sample integration adds a **NiceLabel Print** button to D365SCM that opens a **NiceLabel Print window** to allow your users to print labels after they choose their template, printer, print quantity, and click **OK**.
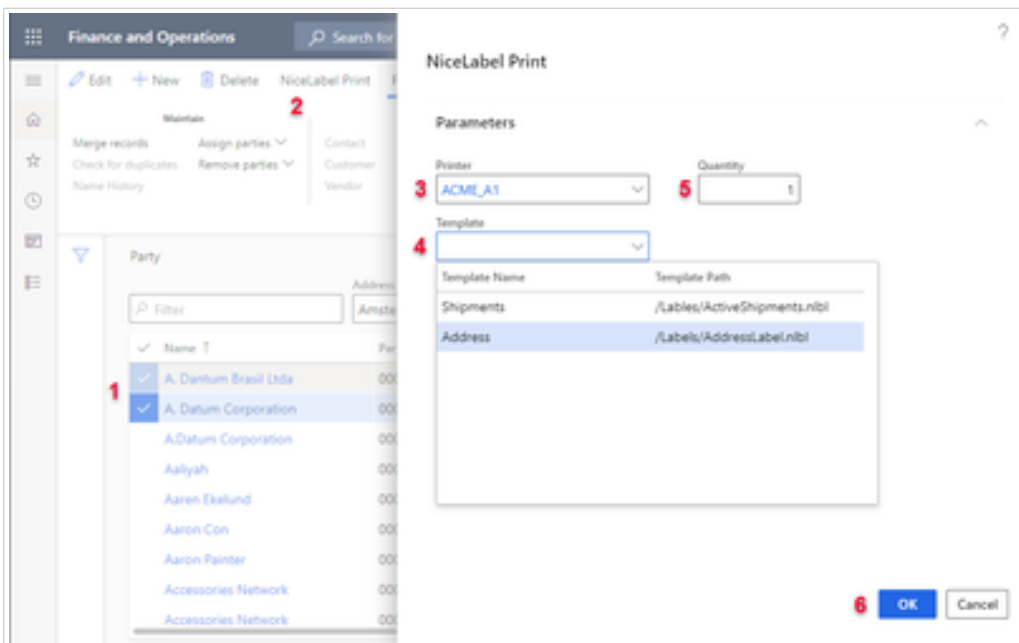
Custom printing form inside D365SCM.

To print labels from D365SCM in our sample integration, go to **D365SCM** > **Modules** > **Common** > **Global address book**.

1. Select records to print.

2. Click **NiceLabel Print** in the top menu. The **NiceLabel Print** window opens.

3. Choose your **Printer**. The dropdown shows a list of your previously retrieved printers.

4. Choose your **Label template**.

5. Choose how many copies to print for each selected record.

6. Click **OK**.

Your selected labels print on your selected printer from D365SCM. You see a success notification message.

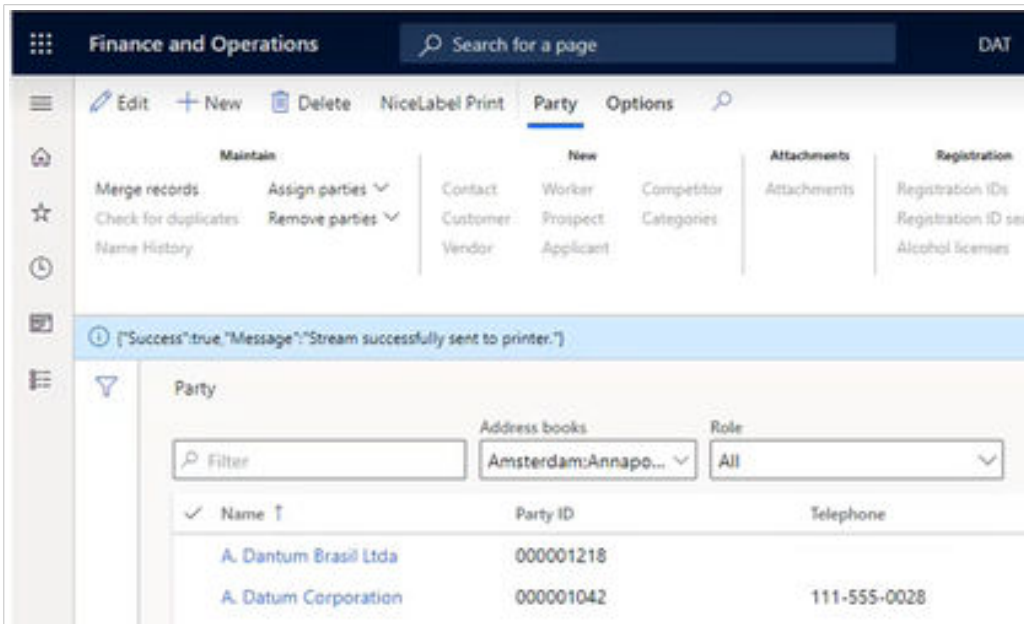*Printed address labels with data from D365SCM.*

Successfully printing labels from D365SCM.

4.4.1. Code: DirPartyTable.NiceLabelPrinting

This sample integration extends your global address book with the **DirPartyTable.NiceLabelPrinting** form extension, and adds a **NiceLabel Print** button. When your users click the NiceLabel Print button, they call the **DirPartyTable_NiceLabelPrinting.NLPrint_OnClicked** method.

4.4.2. Code: DirPartyTable_NiceLabelPrinting.NLPrint_OnClicked (Part 1)

Clicking the **NiceLabel Print** button creates a **NLPrinting** object and verifies the configuration parameters from the **Integration Setup** > **General** tab. After verification, the process continues by opening the **NLPPrintDialog** dialog.

```
public static void NLPrint_OnClicked(FormControl sender, FormControlEventArgs
e)
{
    // Verify REST API configuration
    NLPrinting nlp = new NLPrinting();
    if (!nlp.configValid)
    {
        error("Printing configuration is not valid!");
        return;
    }

    // Open print dialog
    NLPPrintDialog dialog = new NLPPrintDialog();
    if(!dialog.prompt())
        return;

    // Collect basic print parameters
    ...
}
```

4.4.3. Code: NLPPrintDialog

Your users use the **NLPPrintDialog** to type the information required for each print request:

- Which printers to print on.

- Which label templates to use.

- How many copies to print.

> **NOTE**
> If any of this information is missing, you get REST API call errors.

The **NLPPrintDialog** uses data saved in your **Integration Setup** tabs so your users can select printers and template names from lists, and only need to type in a quantity to print.

```
class NLPPrintDialog extends RunBase
{
    public DialogField fieldPrinterName;
    public DialogField fieldTemplateName;
    public DialogField fieldQuantity;

    public Object Dialog()
    {
        Dialog dialog;
```

```
        dialog = super();

        dialog.caption("@NiceLabelPrintingDemo:NLPPrint");

        fieldPrinterName = dialog.addField(extendedTypeStr(NLPEDTPrinterId),
"@NiceLabelPrintingDemo:NLPPrinter");
        fieldPrinterName.mandatory_RU(true);
        fieldTemplateName = dialog.addField(extendedTypeStr(NLPEDTTemplateId),
"@NiceLabelPrintingDemo:NLPTemplate");
        fieldTemplateName.mandatory_RU(true);
        fieldQuantity = dialog.addField(extendedTypeStr(NLPEDTQuantity),
"@NiceLabelPrintingDemo:NLPQuantity");
        fieldQuantity.mandatory_RU(true);
        fieldQuantity.value(1);

        return dialog;
    }

}
```

4.4.4. Code: DirPartyTable_NiceLabelPrinting.NLPrint_OnClicked (Part 2)

Your user input determines the next step of the print process:

- Users click **Cancel** to exit the printing process.

- Users click **OK** to continue with dialog parameter values collection.

You can use selected printer names and the provided print quantities directly, but not for label template paths you get with the **NLPrinting.GetTemplatePath** method. All your basic print parameters are now collected.

```
public static void NLPrint_OnClicked(FormControl sender, FormControlEventArgs
e)
{
    ...
    // Open print dialog
    NLPPrintDialog dialog = new NLPPrintDialog();
    if(!dialog.prompt())
        return;

    // Collect basic print parameters
    str selectedPrinter = dialog.fieldPrinterName.value();
    str selectedTemplate = dialog.fieldTemplateName.value();
    int selectedQuantity = dialog.fieldQuantity.value();
```

```
    str templatePath = nlp.GetTemplatePath(selectedTemplate);


    // Start building JSON print request
    ...
}
```

4.4.5. Code: NLPrinting.GetTemplatePath

This method searches for label template paths saved under the template names you provide. Your data gets saved in the **NLPTemplateParameters** table from **Integration Setup** > **Templates**.

```
public str GetTemplatePath(str templateName)
{
    NLPTemplateParameters templateParameters;
    select TemplatePath from templateParameters where
templateParameters.TemplateName == templateName;


    return templateParameters.TemplatePath;
}
```

4.4.6. Code: DirPartyTable_NiceLabelPrinting.NLPrint_OnClicked (Part 3)

Your collected and prepared basic parameters construct your JSON print requests. Your basic print parameters go in JSON object roots and must always be present.

```
public static void NLPrint_OnClicked(FormControl sender, FormControlEventArgs
e)
{
    ...
    // Start building JSON print request
    StringWriter stringWriter = new System.IO.StringWriter();
    JsonTextWriter jsonWriter = new
Newtonsoft.Json.JsonTextWriter(stringWriter);


    jsonWriter.WriteStartObject();


    // Add basic print parameters to JSON request
    jsonWriter.WritePropertyName("Printer");
    jsonWriter.WriteValue(selectedPrinter);


    jsonWriter.WritePropertyName("FilePath");
    jsonWriter.WriteValue(templatePath);


    jsonWriter.WritePropertyName("Quantity");
    jsonWriter.WriteValue(selectedQuantity);
```

```
    // Add your variable parameters to JSON request
    ...
}
```

The variables object follows your basic parameters. Your sample integration fills variable values with the actual values from selected address book records.

Your variables go in root JSON objects as arrays of JSON objects. Each JSON object in a variables array represents 1 set of variables corresponding to 1 D365SCM record.

For example, to print 2 D365SCM records, you must provide 2 sets of variable values.

```
public static void NLPrint_OnClicked(FormControl sender, FormControlEventArgs
e)
{
    ...
    // Add your variable parameters to JSON request
    jsonWriter.WritePropertyName("Variables");
    jsonWriter.WriteStartArray();

    DirPartyTable partyRecord;
    DirPartyPostalAddressView addressRecord;
    LogisticsAddressCountryRegionNameView countryRecord;

    MultiSelectionHelper selection = MultiSelectionHelper::construct();

selection.parmDatasource(sender.formRun().dataSource(tableStr(DirPartyTable)));
    partyRecord = selection.getFirst();

    while (partyRecord.RecId != 0)
    {
        select Street, StreetNumber, City, ZipCode, CountryRegionId from
addressRecord
            where addressRecord.IsPrimary == NoYes::Yes && addressRecord.Party
== partyRecord.RecId;

        select ShortName from countryRecord
        where countryRecord.CountryRegionId == addressRecord.CountryRegionId
&& countryRecord.LanguageId == CompanyInfo::languageId();
        // add hard-coded en-us if there is no record for languageid()

        jsonWriter.WriteStartObject();

        jsonWriter.WritePropertyName("name");
        jsonWriter.WriteValue(partyRecord.Name);

        jsonWriter.WritePropertyName("address");
```

```
        jsonWriter.WriteValue(addressRecord.Street + " " +
addressRecord.StreetNumber);

        jsonWriter.WritePropertyName("city");
        jsonWriter.WriteValue(addressRecord.City);

        jsonWriter.WritePropertyName("postCode");
        jsonWriter.WriteValue(addressRecord.ZipCode);

        jsonWriter.WritePropertyName("country");
        jsonWriter.WriteValue(countryRecord.ShortName);

        jsonWriter.WriteEndObject();

        partyRecord = selection.getNext();
    }

    jsonWriter.WriteEndArray();

    // complete building JSON request
    ...
}
```

Adding variable values completes your JSON request objects. To print, you call the
NLPrinting.NLPrintJSON method.

```
public static void NLPrint_OnClicked(FormControl sender, FormControlEventArgs
e)
{
    ...

    // complete building JSON request
    jsonWriter.WriteEndObject();

    // call REST API and perform print
    info(nlp.NLPrintJSON(selectedPrinter, stringWriter.ToString()));
}
```

4.4.7. Code: NLPrinting.NLPrintJSON

To actually perform printing, you send JSON requests to the REST API. Your **UseCloudPrinting** configuration
determines the next REST API to call:

- **Cloud Printers** use the PUT Print REST API. You can test this call on the developer portal: https://
  developerportal.onnicelabel.com/docs/services/cloud-print/operations/put-printfunction

- **Classic printers** use the PUT Cloud Trigger REST API with the trigger name **Api-CloudIntegrationDemo-Print**. You can test this call on the developer portal: https://developerportal.onnicelabel.com/docs/services/cloud-trigger/operations/put-cloudtriggerfunction.

```
public str NLPrintJSON(str printerName, str JSONContent)
{
    if (useCloudPrinting)
    {
        return (obj as
NLApiClient.CloudPrintIntegrator).Print(NLPrintSubscriptionKey, printerName,
JSONContent);
    }
    else
    {
        return (obj as
NLApiClient.CloudTriggerIntegrator).Print(NLPrintSubscriptionKey, JSONContent);
    }
}
```

Example of a sample integration JSON print request for 2 selected records:

```
{
    "Printer":"ACME_A1",
    "FilePath":"/Labels/AddressLabel.nlbl",
    "Quantity":1,
    "Variables":[
        {
            "name":"A. Dantum Brasil Ltda",
            "address":"Rua Azevedo Soares 1245",
            "city":"São Paulo",
            "postCode":"03322001",
            "country":"Brazil"
        },
        {
            "name":"A. Datum Corporation",
            "address":"123 Main Street ",
            "city":"Sacramento",
            "postCode":"94279",
            "country":"United States"
        }
    ]
}
```

4.4.8. Checking variable names

When you customize your integration code, specify which data from D365SCM to use as label templates variables. Check the variable names on your label templates and in D365SCM to make sure your data matches and is used correctly on printed labels.

1. Go to **Control Center** > **Documents** and select your label template file.

2. Right-click your file and choose **Preview label**.

The **Preview label** window shows your variable names and generates previews using variable values.
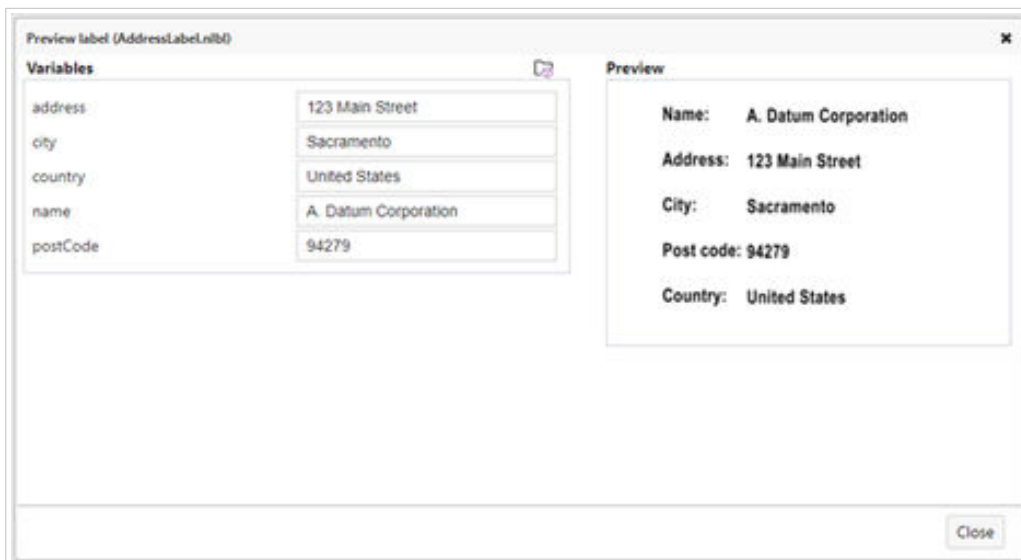
*Previewing label template files with variables in **Control Center** > **Documents**.*

> **NOTE**
>
> REST APIs use default values for any unspecified variable values on label templates, and ignore any variables not used on label templates.

Customize our sample integration code for your printing environment so your users can print records they select using selected label templates on selected printers directly from Microsoft Dynamics 365 for Supply Chain Management.