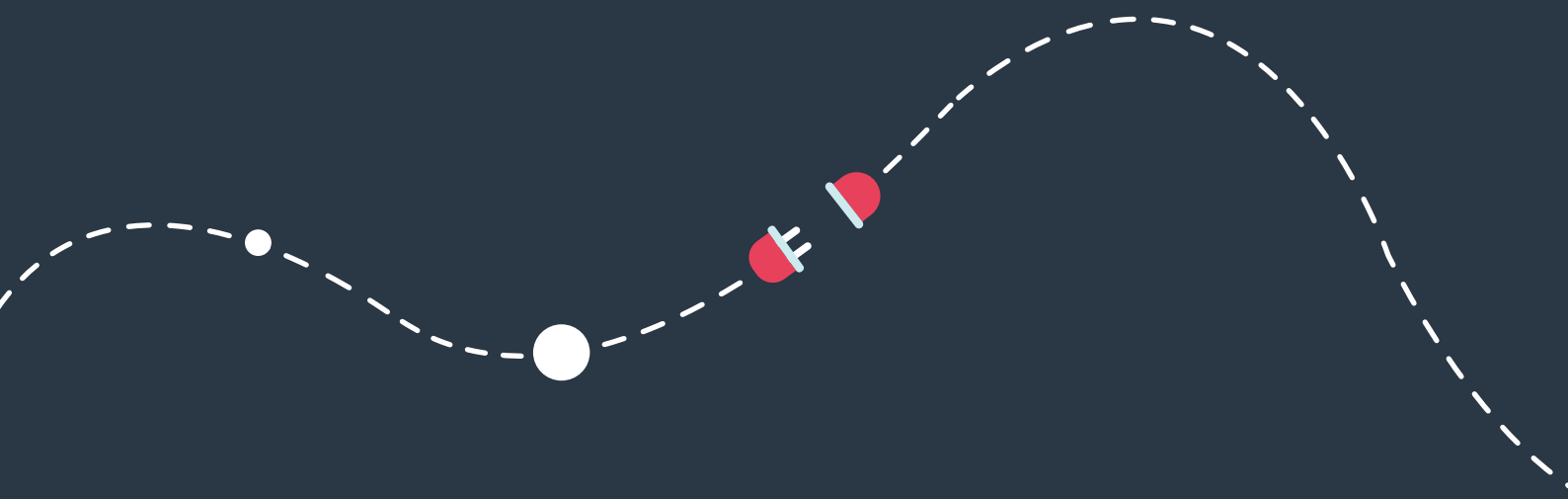
Getting started with scoping integrations for your users



Contents

Integration Overview

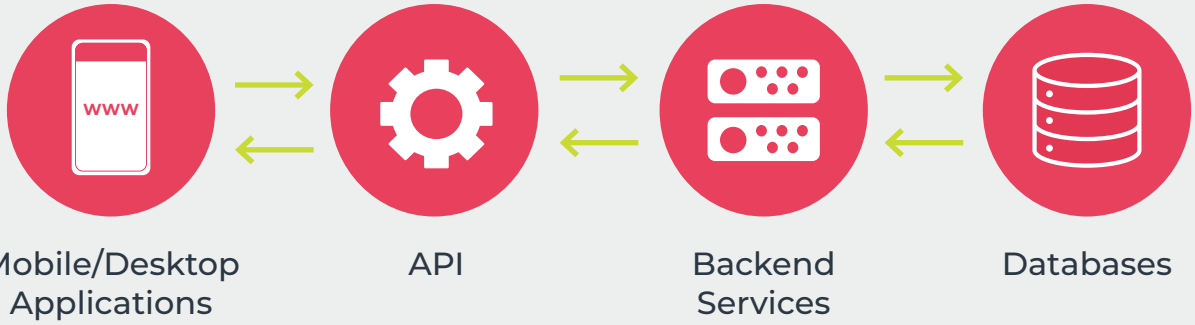
The importance of scoping your integrations

Integration involves connecting and unifying data spread across numerous platforms into one single view. It's important therefore that prior to building workflows you scope the integration project, to understand the use case and where the data is and will go.

For instance, if a customer requests an integration, or you require a new integration to expand your products functionality, you need to understand the API and if it is capable to perform what you need it to.

Frequently high-level requirements miss the low-level detail required to create a fully optimised integrations.

This lack of clarity in a definition can stem from a number of factors.



What should you look for before starting your integration builds?

There are several key things you should look for before starting your integration builds.

1.	Check whether the API is available in the providers API library and if unavailable you will need to build a custom API, or request it to be built.
2.	See if there are test credentials available for you to access authentication details of the API
3.	Understand the specific **API** authentication type, methods, end-points and setup required.
4.	Locate and utilise the appropriate APIs documentation to fully exploit the APIs capabilities.
5.	Understand the full use case and know what you are building.

Scoping an Integration

The key to creating a successful integration hinges on its original definition. While on the face of it this should seem like a simple enough task, often high-level requirements miss the low-level detail required to create a fully optimised integration.

Break down language requests

SaaS companies regularly receive 'Integration Request' tickets but the ways in which they are conveyed can vary vastly.

For example a common message may simply be:

> "I want an integration with Zendesk."

This request identifies a need for an integration between the host application and Zendesk. It is however the lowest level of request. Offering little insight into what they want to integrate.

It is an almost near-impossible brief as popular applications like this offer a vast array of services. With no indication as to the basics of what the user is trying to achieve, creating an integration without some back-and-forth makes responding quickly to this request a lottery.

Now at least there is some detail with this integration request with areas of the application that need to be worked with such as:

> "I want to sync my contacts with Zendesk."

- the application to connect to
- the methods to be used to create the integration
- that it's a two-way integration

Unpacking this you'll get the barebones view of what the integration will look like. However, you'll still need to further define what data fields within the contacts are required to update between the applications.

Finally, this request provides more detail in order to perform a more complex bespoke task.

> "I want to create a new ticket in Zendesk when a deal reaches a certain stage."

On top of the initial requirements you have discovered:

- a trigger (a deal reaching a certain stage)
- an action (create a ticket in Zendesk)

This request also tells you that you will have to include some form of logic in your integration so it will only run when a record meets a certain condition. When it comes to integration requests the more detail the better.

Understanding the language behind requests

One of the first stumbling blocks in defining an integration is the language used. Depending on the technical competence of the person requesting the integration you may find verbs such as “sync”, “automation”, “connection”, “workflow” or even branded terms such as Zap or Cycle.

While they mostly all refer to the same concept, there are hidden details to be found within them:

Term	Definition
Automation	Term often used to describe a background task to improve productivity, typically with an automated trigger and a defined action
Connection	Indicates that a data link between systems is required, with non-specific requirements
Sync/Integration	Two-way data integration, ensuring records in both SaaS applications match
Workflow	Indicates a full process, often with multiple systems and the need for logic

Therefore don’t take the terminology at face value. We regularly have conversations where the individual says ‘sync’ but means ‘automation’, ‘automation’ but means ‘workflow’.

Understanding the language behind methods

An API can have multiple methods all of which provide sequences of instructions for an API call to make. The API only has access to the data known to that API. This ensures data integrity among the set of objects in an application.

An APIs methods require specific information to piece together the API calls. For instance, the account ID, email address, order number and so on.

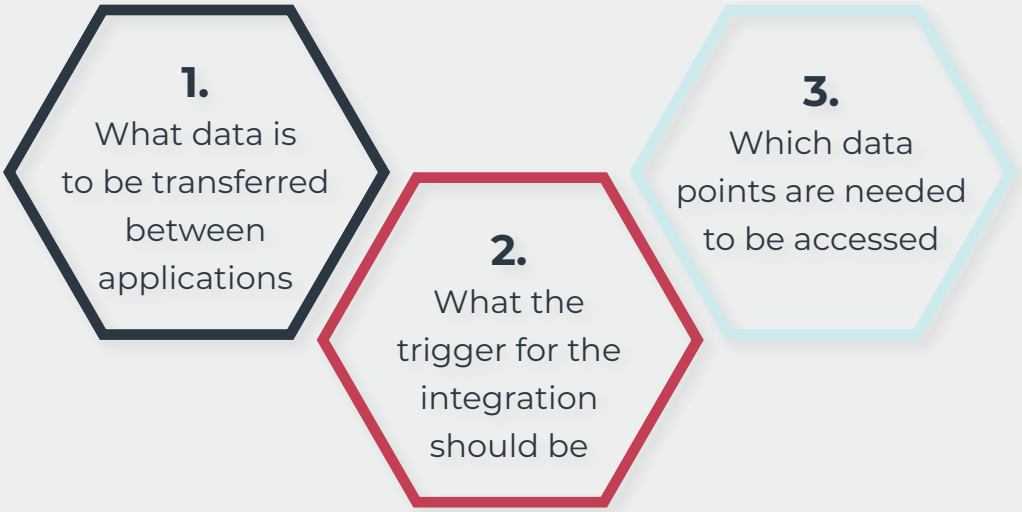
In regards to methods it is important to understand their pluralisation. As one method perhaps calls one record, and another calls multiple. For example, ‘Get Contact’ or ‘Get Contacts’, one calls one record whereas the other calls a batch. This changes what you would do with the data after it is received.

Required Fields

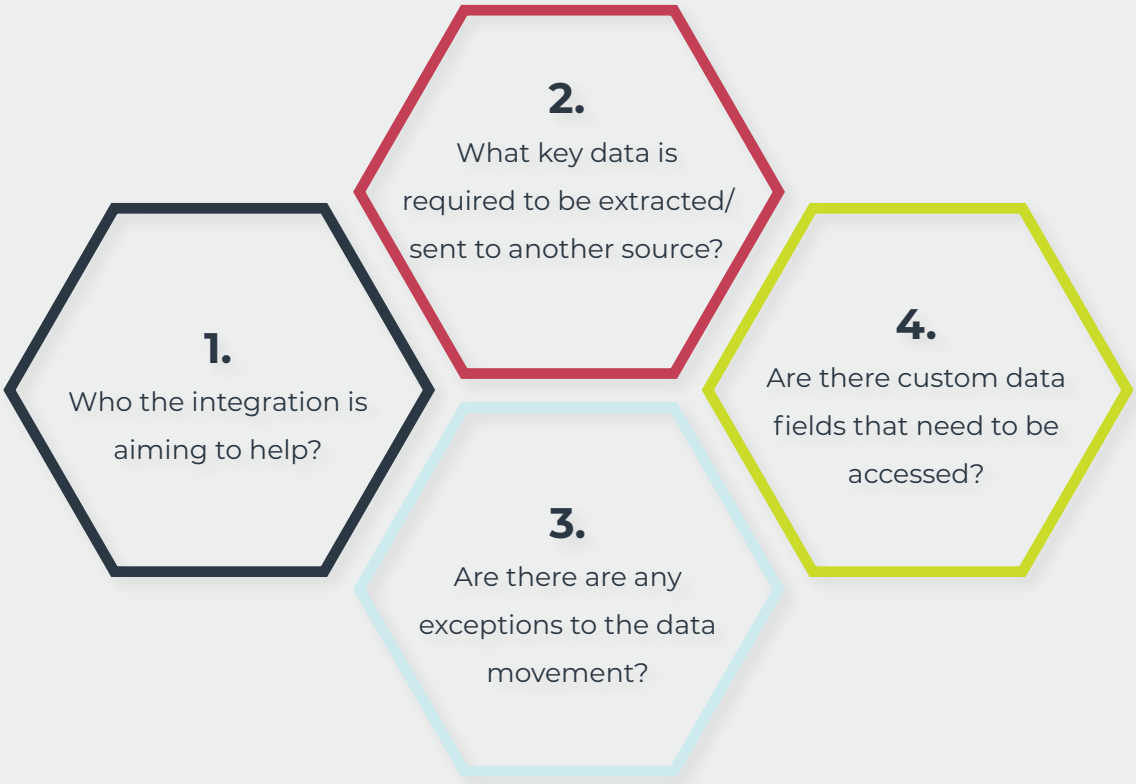
Required fields are what the API needs to find the right records ID’s, names, or dates especially for big data sets.

Required Use Case Information

At a base level, you are going to need to know the following information to fully design an integration:



These may be unpacked from an initial request (as the examples above), but you may need a deeper level of understanding, including:



These pieces of information can help you hone the areas of a system you'll be working with. For example, if you are told that the integration is to help sales team members manage their deals, you know that within a CRM system you'll be working with objects such as deals, organisations and contacts as a minimum.

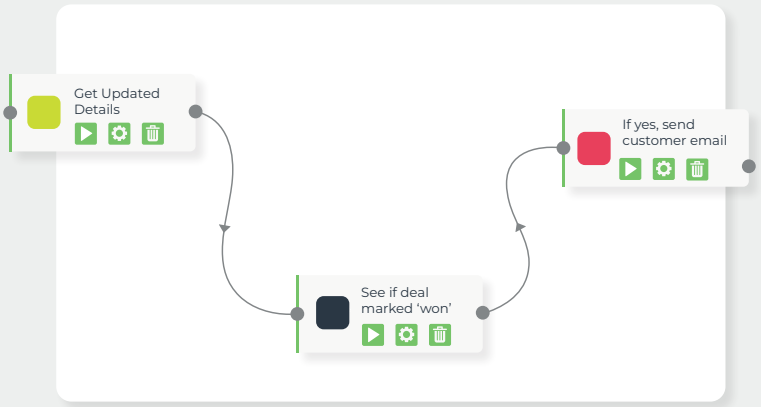
Unpacking use cases to create workflow integrations

Once the complete integration use case has been defined by your user it will need to be translated into how the applications you want to integrate work.

Different applications language and terms can vary. So while you may know the areas and objects you want to interact with not all APIs will be ready baked to give you the necessary data from a single endpoint.

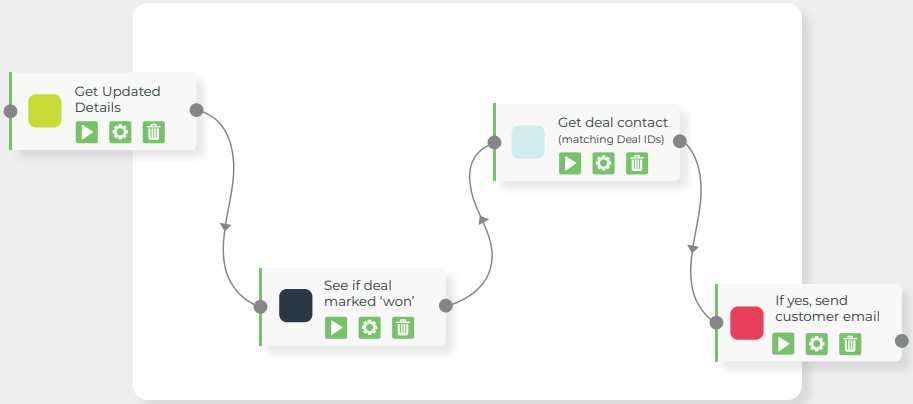
For example, you want to be able to automatically send a user a message when the deal they are attached to in your CRM has been marked “won”.

This use case could look something like this:



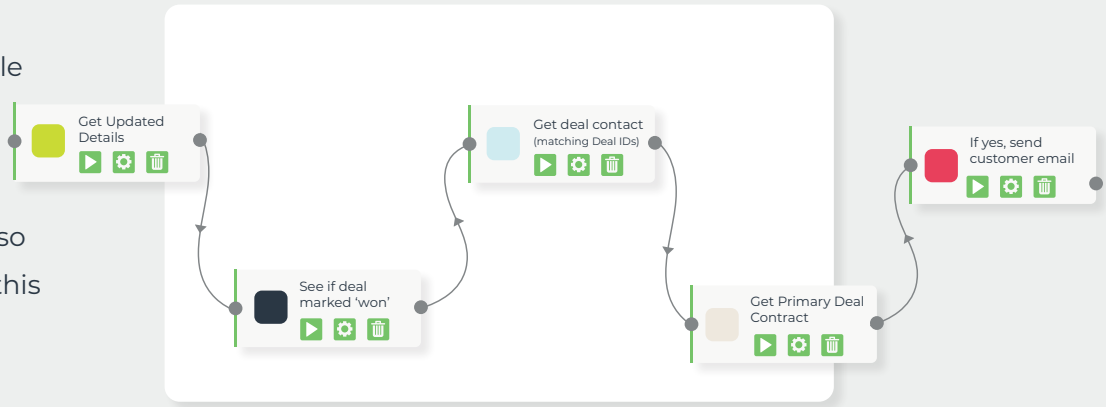
Not all CRMs directly provide the contact details of those attached to a deal.

You may need to amend this flow to something like this:



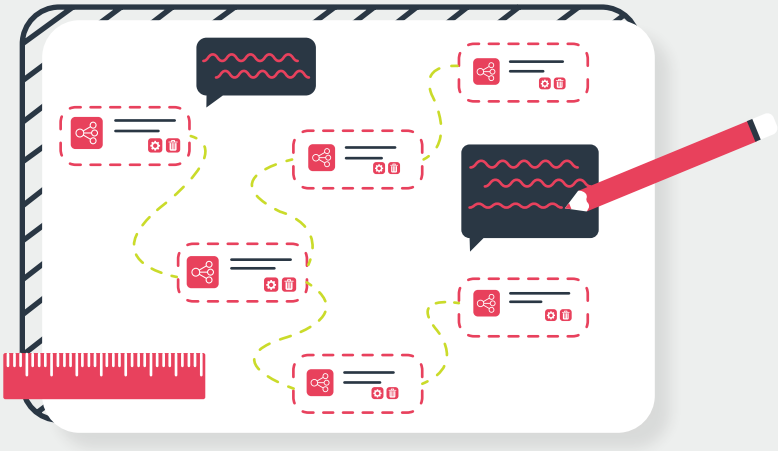
But what if there are multiple contacts on a deal?

With some CRMs you can define a “Primary Contact”, so the integration can extend this workflow like so:



By familiarising yourself with how an application interacts with itself you will begin to be able to formalise these integrations far quicker.

Tips and Tricks



1 Integration Planning

Sketching out initial ideas to create and manage the process shouldn't be skipped.

A quick sketch can help identify data sources, logic/ data orchestration and set objectives which can make the world of difference when it comes to building efficient and resilient integrations.

2 Know your Use Case

If you're building integrations between your SaaS and an array of third party applications then you already know how your own users interact with your application and the data contained therein.

This helps selecting what data you need to access for the integration, how to attribute records within your application, and work out if the integration will be triggered by polling or an event driven webhook.

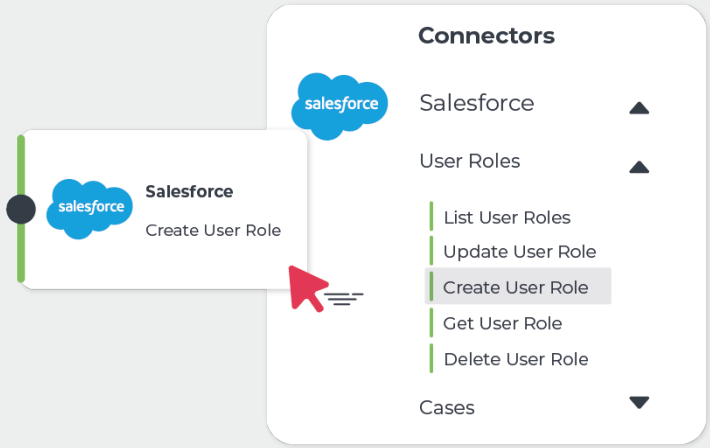
With those in mind, you can convert the conceptual idea behind each of your use cases to practical, pseudo-integrations that break down the individual steps needed to take to achieve your integration's goals.

Better yet, draw out a flow chart of the process you are aiming to automate. Being able to visually look over your automation is a huge advantage when laying up your integration.

3 Identifying Methods to Use

Every SaaS application has its own particularities, this can lead to inconsistencies when making assumptions on the API capabilities of applications in the same vertical markets.

One application may opt to complete a task using a polled method, such as Get New Contact, another platform may choose to offer the same service through a Webhook event. While the difference may be subtle, it can affect the choices you make when designing your integration workflow.



It is therefore important in your integration planning and preparation to check what methods are available with the APIs you want to work with. Methods are typically listed in an applications documentation or within their API library.

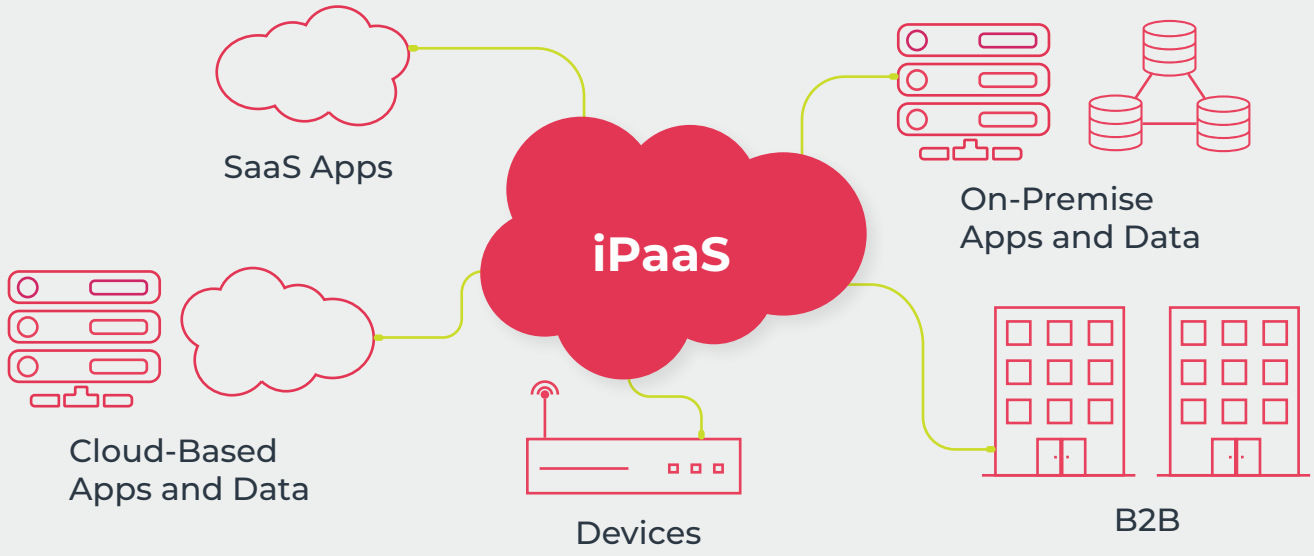
What tools are available to build integrations?

iPaaS

The iPaaS (integration Platform as a Service) cloud model allows customers to build integrations between applications or data sources within the cloud. In other words they provide users with ready-made tools and elements to construct connections between systems.

> **"A suite of cloud services enabling development, execution and governance of integration flows."** Gartner

In other words an iPaaS connects combinations of on site and cloud based processes, applications and data within individual or across multiple organisations.

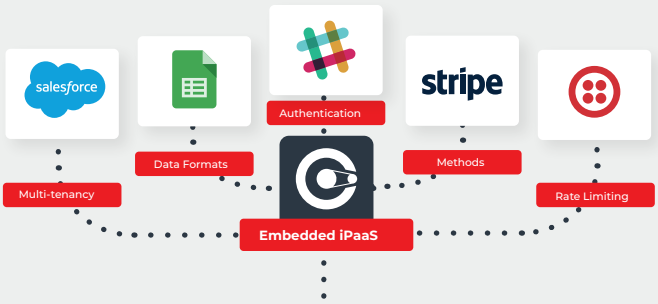


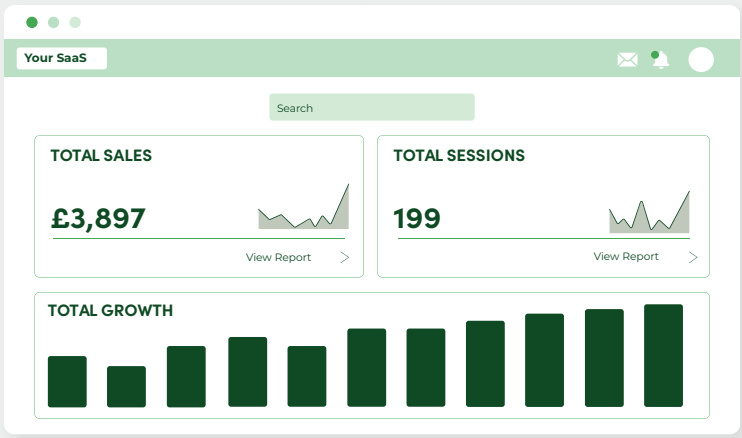
Embedded iPaaS

On the other hand, an Embedded iPaaS solution allows SaaS vendors to create, manage and deploy integrations to their clients directly from within their own platform.

Therefore enabling a SaaS vendor to resolve integrations on behalf of their users. All seamlessly from within their own SaaS product.

In addition the embedded iPaaS is able to run automations and logic, providing more than just simple updates from one platform to another.





Integration Tools

Developer: POSTMAN

Postman is an API platform for building and using APIs by simiplifying each step of the API lifecycle and streamlines collaboration to create better APIs faster.

The developer tool provides a comprehensive set of API tools that help accelerate the API lifecycle through design, testing and documentation. As well as an API repository where you can store, catalog and collaborate around your API artifacts on one platform.

Postman enables advanced intelligence and insights about all of your API operations. It does this by leveraging alerts and security warnings, search and reporting. It also allows you to integrate the most important tools in your software development pipeline to enable API-first practices.

Postman allows users to build their own integrations with the Postman API. Its endpoints help to integrate Postman within your development toolchain by allowing you to programmatically access data stored in your Postman account with ease. As an open source users can extend their API workflows in any way they want.

Developer: IDE/Programming Text Editor

A text editor is simply a computer program and a tool used for editing plain text. It allows you to create and edit a range of programming language files.

An IDE (integrated development environment), on the other hand, is a full-fledge software environment that consolidates basic developer tools required to build and test software. It combines common developer tools into a single graphical user interface (GUI).

```
const connectorName = 'MailChimp';

Cyclr
    .installConnector ({
        name: connectorName,
        description: 'US1 testing account',
        apiKey: 'secret'
    })
    .then (response => console.log(response) )
    .catch (error => console.error(error) );

let templates =
Cyclr.getTemplates(connectorName);
```

Integration Type Examples

System integration is the process of joining software into a single infrastructure. As a result data spread across numerous platforms can be fed into each other. Then taken from these disparate systems into one platform.

There it will provide a single unified view of all the data across the organisation. This helps with business intelligence and making informed future decisions.

Legacy System Integration

Legacy systems are common among older Enterprise organisations. Within these organisations the legacy software typically performs core business functions. Therefore it would be costly and complicated to remove or change systems due to the volume of data within.

As a result integrations can help update and modernise the system rather than completely replace it.

The modernised legacy system can then communicate data with other new systems. This helps the organisation to move forward in its digital transformation process.

Enterprise Application Integration

EAI is useful when a growing enterprise uses several different applications. These tend to accumulate huge amounts of data separately.

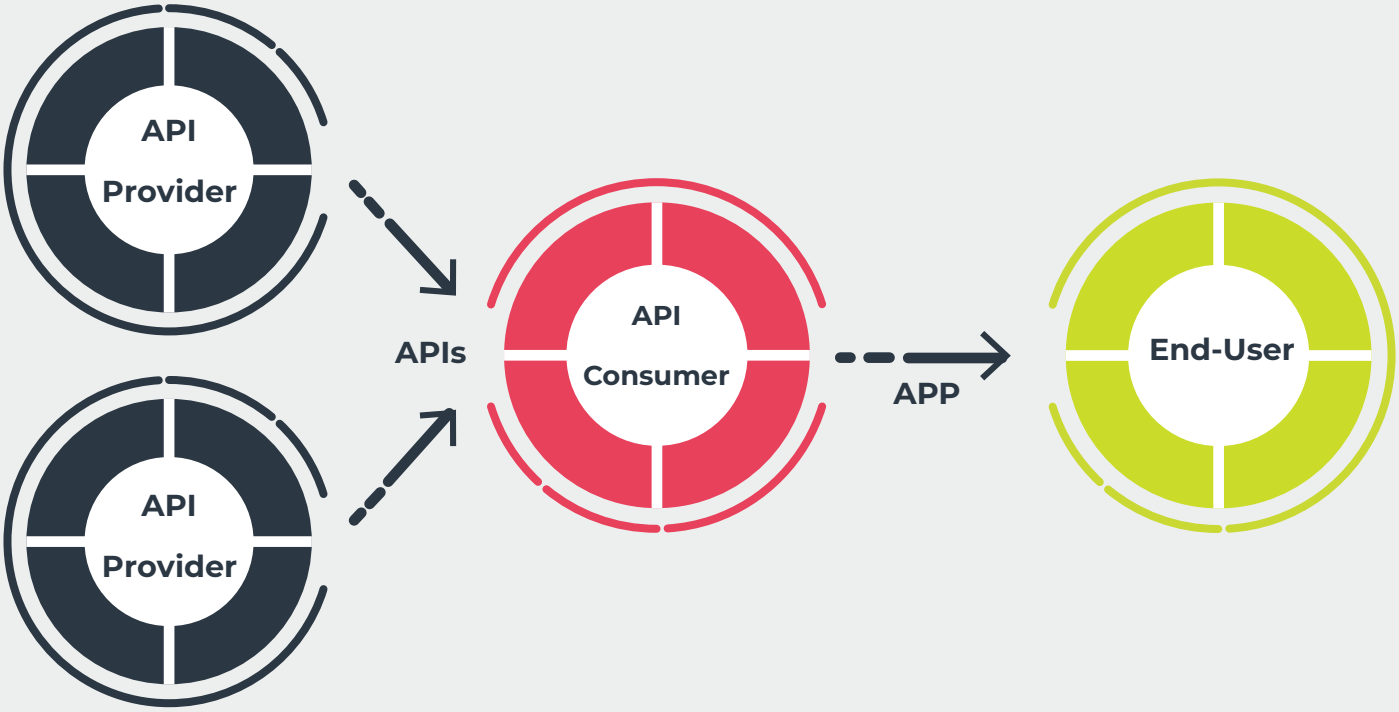
Using enterprise application integration would unify the different subsystems into one environment. As a result it automates real-time data exchange between these different applications.

Third-Party System Integration

Using a third-party integration enables an organisation to add new functionality to an existing application.

There could be several reasons as to why an organisation would do this, but mainly because of a lack of resources.

Resources such as time and money to develop new features from scratch.



Business to Business Integration

Finally, B2B integration helps to connect systems of multiple organisations to automate transactions and exchanges.

For instance, connecting a purchasing system to a supplier and distribution ERP.

Specific types of SaaS integrations

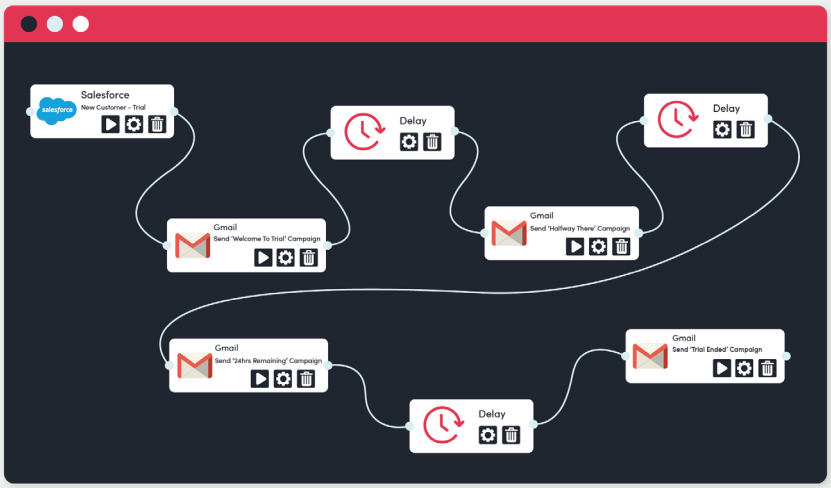
When it comes to integrations they can be as simple or as complex as you want them to be. You can build workflows in a multitude of different ways to suit you and your customers needs. These tailored workflows can help streamline processes, increase productivity and save time.

CRM

Some CRM software have pre-built native integrations, these however may not meet your specific requirements, and requesting new integrations takes time. A more complex integration could tackle a new customer signing up to trial your product and the onboarding process.

Their data is automatically input into your CRM and keeps track of their trial progress.

When linked to your email provider you can automatically send out personalised onboarding/ reminder emails throughout their trial, helpful tips, how long is remaining, and the process to take once it has come to an end.



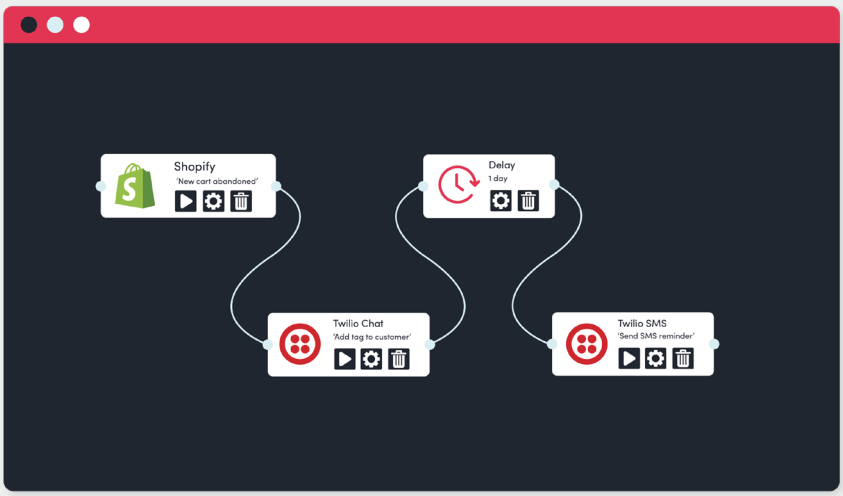
Support

Support, especially chat based applications, are a typical third-party integration typically with eCommerce sites and are designed to help users with a new product or an existing product they own.

They allow for quick interactions and improve a customer's experience with a website.

For instance, when an existing customer abandons their cart, this could trigger an integration for a ticket to be created, and a tag added to their account.

Then the integration could send out a SMS to the customer reminding them of the items in their cart.



Specific types of SaaS integrations

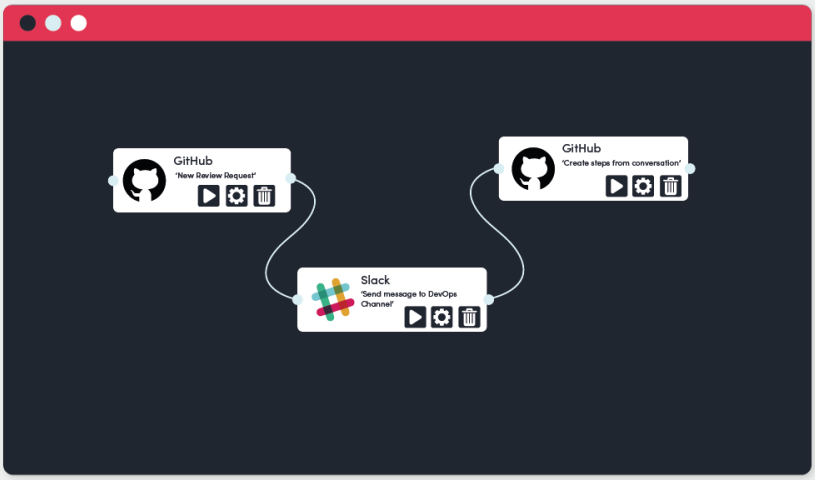
Project Management

Project management integrations are typically used to help organise tasks, assign them, track file storage, encourage team collaboration, and see project timelines.

For example a communication integration can help teams to organise, and assign tasks.

One integration example could be used to send messages to Slack when a new comment, issue or review request has occurred in GitHub.

The messages are immediately sent to the appropriate channel and picked up or assigned to a team member to chase.



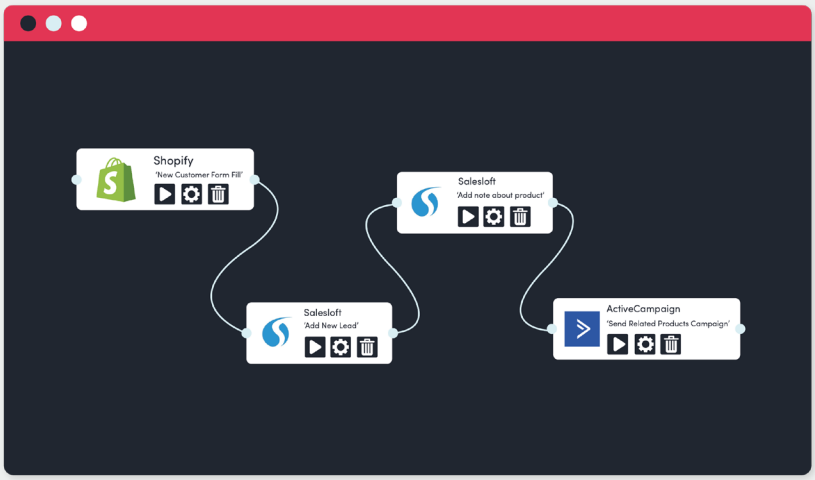
This creates a solution for recurring tasks, and automates a process that can take up time.

Sales & Marketing

Using integrations between sales and marketing applications can help boost lead generation, nurturing and reduce human error.

For example, a sales funnel integration could be initiated after a customer expresses interest in a particular product.

This results in an automated email response suggesting other products that are similar and provide more information to the customer. These are however at a higher price point.



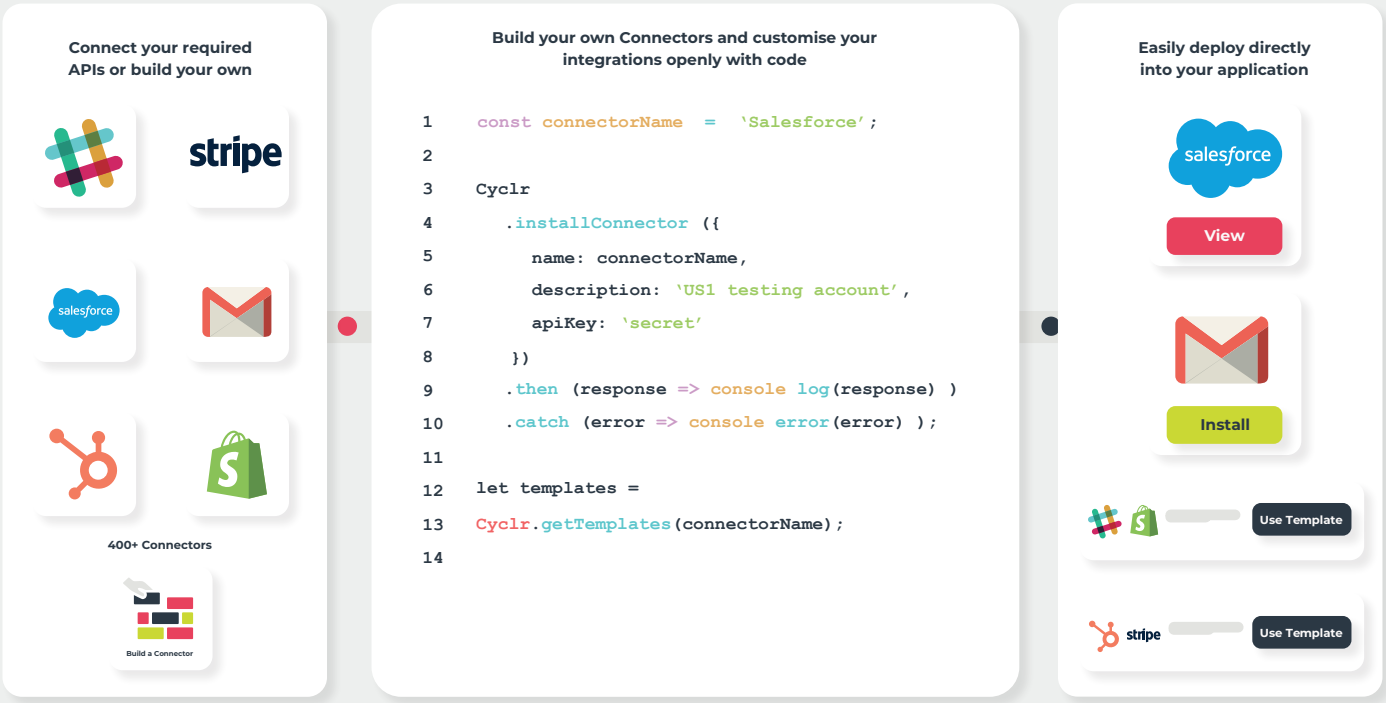
This integration example demonstrates one way of upselling/cross-selling to your customers.

Integration Proxy API

What is an API proxy?

An API proxy is a thin application program interface (API) server that exposes an interface for an existing service or services. It acts as an intermediary for something else.

For instance, Cyclr has a flexible proxy API to enable in-app integration and external data access.



How does Cyclr's API work?

Our API is commonly used to deliver advanced, user friendly custom integration experiences behind your SaaS styling and UI. It can also act as a proxy allowing you to perform more advanced data processing before the data enters your application.

An example of this is when a partner requests an API call, it could be something like 'Get Account ID' from Salesforce. Cyclr performs the request and once the call has been made the unmapped data is returned to Cyclr and then sent to the partner. (differences in API - this is a standard process)

This method provides a single API to connect and bring data into your application and interact with them in a uniform way. Cyclr's SaaS Connectors already handle authentication, rate limiting, endpoint mapping and more.

As a result developers can concentrate on creating the calls to access the data you need without worrying about the development overhead.

Benefits of Integration

8 Benefits of Integrations	
1	The introduction of integration and automation means resources can be diverted elsewhere, enhance workflows and improve accuracy and consistency in operations.
2	Adopting a low code integration platform can help these processes and significantly transform your business.
3	Streamlined operations can result in increased reliability, optimised performance and reduced employee turnover.
4	Automating and integrating is important for service providers as it can give the company a competitive edge.
5	It improves the user experience for customers, and has long term benefits of increased value and brand loyalty.
6	They provide better consistency in your service, faster response times and potentially a cost decrease.
7	Provides data accuracy that is more trustworthy as the risk of human error in data entry has been eliminated.
8	Automated data input helps business leaders to make quicker decisions. These decisions are based on the whole data picture from one platform, providing the organisation with a cost-effective solution.

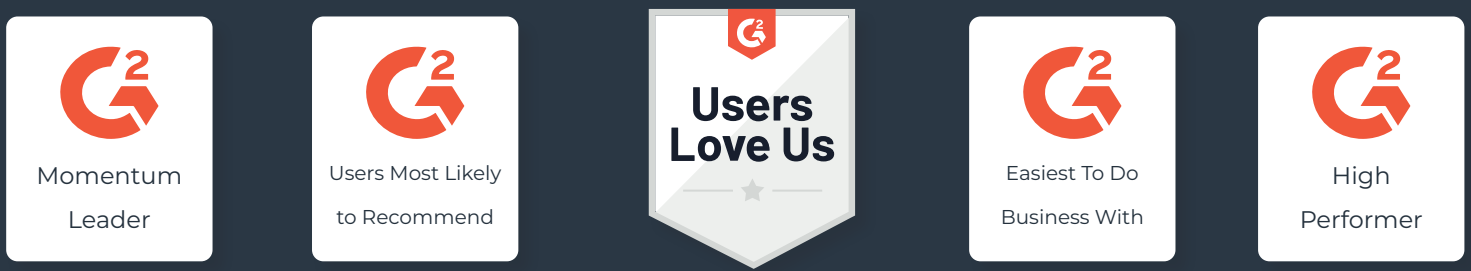
Integrated automation solutions allow people to do more with a lot less.

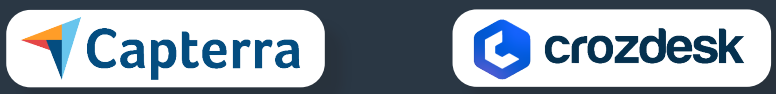
Cyclr is an iPaas built from the ground up for embedding, simple to use, with flexible deployment.

An easy to use integration platform with low-code tools empowering anyone in your SaaS to respond to user integration requests without adding to your developer backlog.

A developer toolkit that can be used by commercial teams.

So, starting your integration journey is closer and easier than you think.





cyclr.com