Nylas

The Ultimate Guide to Email Integrations

The Pros and Cons of Using APIs vs. Building It Yourself

2019

Nylas

Contents

Why now: the urgency to offer business tools like email, calendar, and contacts within your SaaS app

"Data from smart, connected products is generating insights that help businesses, customers, and partners optimize product performance."

— Harvard Business Review[1]

We live in an era of connectivity: connected cars, connected homes, and social platforms for connecting. It's no wonder, then, that businesses and consumers are keen to adopt software that connects with other tools and applications they use every day — like email.

Even with the explosion of messaging tools, email continues to be the bedrock of business communication. In fact, more people use email today than ever before in history. By 2024, the number of worldwide email users will increase to more than 4.2B, and the email marketing industry alone will be worth $22.16bn.[3]

In response to this data, more SaaS applications offer an email communication layer within their apps: from customer relationship management software (CRM), to applicant tracking software (ATS), to HR, legal, and real estate software.

Email integration is a powerful asset to any SaaS app. Users become more engaged with your platform when they can send, receive, and track analytics on emails all within one dashboard. The need to context-switch between what were formerly two disparate systems — email service providers (ESPs) and software applications — is reduced or eradicated. Time spent on manual data entry goes down, your customer's efficiency goes up, and user adoption grows.

In fact, one study from Accenture shows that sales reps alone spend as much as 20% of their day on manual data entry.[4] By eliminating this need, you make the end users of your software 235% more efficient — a huge value-add that can easily be monetized.

While it's clear that creating mailbox integration for your application is beneficial for both your customers and your business, implementing the functionality is more difficult. When approaching the decision to build the integration in-house from scratch, or employ an API, there are a few things to consider. The purpose of this paper is to provide you with the tools you need to navigate the build vs. buy decision and get the biggest ROI from your email integration (both time and financial).

Option 1: Building an email, calendar, and contacts integration from scratch

Engineers love problem-solving, which is why they're often eager to tackle new and complex challenges. Building an email integration is exactly that — a highly complex build with a large number of edge cases per ESP. For some engineers, this could be a fun challenge, but there's a lot to consider when deciding if it's the right step for your team, for example:

How complex is it to build the integration?

Building an email, calendar, and contacts integration occurs in four stages: research, development, implementation, and (often forgotten) updates and maintenance.

Building an integration for one ESP (such as Microsoft Exchange) from scratch can take as long as 12 months for a team of four engineers. As you build integrations for additional ESPs, you'll find that the number of edge-case scenarios, bugs, and server-side issues you encounter increases exponentially.

Below is a quick overview of the integration process for the five major ESPs.

Gmail

Connecting your app to Gmail's email, calendar, and contacts data might seem simple at first. Ask any developer who has built the integration end-to-end, and they'll tell you that simple features like recurring events, time zones, and email attachments are just the tip of the iceberg. Google's OAuth updates and security concerns add another few weeks or months of development time to your timeline.

Anticipated time to integrate: 3-6 months for a team of 4 engineers.

Exchange/ActiveSync (EAS)

Microsoft Exchange is one of the most widely used ESPs, especially in the enterprise — but it's also one of the most challenging integrations to build. Originally, the Exchange protocol was created for pocket PCs developed in the early 2000's. Today, Exchange uses two main protocols - EAS and EAS to send data in binary XML. "Among public companies using cloud-based email, Microsoft is more popular with larger organizations and has more than an 80 percent share of companies using cloud email with revenue above $10 billion," said Jeffrey Mann, research vice president at Gartner.[5]

Microsoft's documentation is thorough but dense — as long as 1,000 pages — and the systems EAS relies on are outdated and unnecessarily complex.[6] For example, all ActiveSync emails must be encoded in a binary version of XML to save bandwidth. In addition, you'll often encounter server-specific issues that can be challenging to debug or replicate.

Anticipated time to integrate: 6-12 months for a team of 4 engineers.

IMAP

Every IMAP server implements the RFC spec slightly differently. To start, you'll want to decide how strict or how permissive you want your implementation to be; the stricter the implementation, the less email functionality and support you'll have. If your implementation is more comprehensive, the integration period could stretch for months.

Building the IMAP implementation will add only email functionality to your app; to add support for calendars and contacts, you need to parse CalDAV and CardDAV objects and then return them in a consistent way to match the way you handle other ESPs.

Anticipated time to integrate: It takes two engineers about a year to support the long-tail of IMAP providers like iCloud, Fastmail, Yahoo, AOL, Mail.ru, and Dovecot.

Office 365

The Office 365 APIs are fairly rudimentary and lack some of the features needed to build a fully supported email integration with your app. For example, the Office 365 APIs don't include email analytics like open and click tracking. You can add attachments to emails, but you can't add inline images.

In addition, the Office 365 APIs have reliability issues that Exchange/ActiveSync doesn't experience. With 0ffice 365, you don't get the high-value hosted or on-premises Microsoft Exchange environments that the Fortune 500/Global 2000 companies overwhelmingly run.

Anticipated time to integrate: 4 months for a team of 4 engineers to build a feature-limited integration.

Outlook.com

Outlook.com is used by millions across the globe, which makes it an essential part of your MVP for your email integration. However, ongoing support and maintenance costs can rise over-time, and updates to the API can introduce breaking changes.

Anticipated time to integrate: 2-3 months for a team of 4 engineers.

What are the ongoing maintenance costs?

Maintenance costs rise as you update your app to support the changing requirements and capabilities of the email service providers. And if a new customer uses an ESP you haven't yet built an integration for, the research, development, implementation, and maintenance process starts all over again.

As your company grows and you sign more clients, the cost of securing and supporting hundreds of thousands of new accounts can cause severe drag in your team's productivity, slowing your time to market and reducing your revenue.

For small engineering teams, maintaining an email integration may require additional headcount. For large engineering teams, it may postpone building other core features that differentiate your product from competitors — be sure to plan accordingly with your product management team and other stakeholders in the organization.

Opportunity cost

One of the biggest risks in building your own email, calendar, and contacts integration is the opportunity cost. What prominent features get delayed while your engineering team is focused on building the email integration from the ground up? How does this affect your competitive advantage over other companies in the same space?

Will the email integration be for internal use only, or will your customers use it?

Some companies simply want to integrate their email with their application internally, while others offer it as a value-add to their SaaS app. If email, calendar, and contacts are a core part of your business and will scale as you bring on new customers, many companies opt to build on top of an API that specializes in these integrations, which we'll cover later.

Summary: Building an email integration from scratch

PROS	CONS
✓ Full control over infrastructure	✗ Difficult and costly to maintain over time
✓ Decreased vendor reliance	✗ Customer support costs
✓ Become experts in a niche field/topic	✗ Security — you have to build your own security infrastructure for the integration
	✗ Opportunity cost
	✗ Slower time to market

Calculating total cost of ownership: Can your bottom line take a custom build?

Costs to consider	Cost per month	Depending on the project scale
Cost of people to build and maintain the integration	± $150-160K per month (average integration build time is 24 months for all email service providers)	Minimum team required: 4 experienced back-end infrastructure engineers (±$800K-1.2M per year for 4 salaries with insurance and other costs) + 1 dev ops engineer + 1 customer success engineer to debug issues + project manager+ CTO/supervisor hours
Cost of operating servers	± $100-200K per month (assuming 100k accounts)	Database storage + I/O + instrumentation cost *Don't underestimate the cost of documentation and training when building your own tech stack! At the very least you'll need to carefully document how new engineers will get up to speed on how to interface and query your integration, and keep that up to date.
Support costs	Technical debt and scope creep that add to costs #1 and #2	10-20 engineers needed to update integrations when ESPs put out new releases, provider APIs are depreciated, or new customers with unique email/calendar infras are onboarded. Additional team of 5-10 technical customer support people will be required to debug customer authentication and syncing issues for around 100K accounts.
Cost of security	± $50K per month	SOC2, ISO 270001, etc. + staff to oversee implementation **Additional costs arise as regions – like GDPR in the EU – update email security standards regularly.
Opportunity cost	Your engineering and product teams will be distracted building and maintaining infrastructure instead of features that they love building and that help drive revenue further.	You may need to delay a handful of features as you build and maintain an email integration Moving slowly may cause you to lose the competitive edge over.

Option 2: Integrating with the Nylas API

"There's real value in APIs both from the technology point of view — e.g., to lower technical barriers to implementing collaborative multi-enterprise processes — and the business point of view — e.g., how APIs help companies to tap into "The Collective" of their external constituents to increase insight into their customers' needs — and even their own business."

— Gartner, 'A Cloud API Manifesto for Integration As A Service' [7]

If you want to go to market faster with an email integration, integrating with a fully supported API is the way to go. APIs are created by developers, for developers after years of research, exploration, and investigation of a multitude of edge-cases (and solving for them). The very best APIs have easy-to-use SDKs for the languages your application is built in as well as clear, updated documentation that make integrating a two- or four- week sprint as opposed to a months long project.

The Nylas API is built for SaaS applications that want to integrate with email faster, and with less risk. It's one simple integration that powers your application with 100% of ESPs, and saves engineering teams as much as 18 months of development time.[8] There are a few additional benefits to using the Nylas API:

Focus on core competencies

While using an API still requires some build time from your in-house time, it's substantially less time-intensive than building a solution from the ground up. The shortened development cycle frees your team up to focus on your company's core competencies — the "secret sauce" that differentiates you from the competitors.

Security and reliability

According to Gartner, "a growing number of Gartner clients are using public cloud services because they expect it to represent a security improvement."[9] It's true that in many cases, API and SaaS companies that focus on a specific product know how to secure that product better than anyone in the business.

Since email security is of utmost importance, and a breach of security could risk all business operations, teams are beginning to outsource security needs to the experts. With Nylas, your data is held to the highest levels of enterprise-grade security and privacy controls. Nylas is EU Privacy Shield Compliant and PCI and HIPAA ready.

Plan for the future with scalable architecture

An email API is the best line of defense against the numerous issues that arise as your data, number of users, and types of email accounts scale.

It also protects you against ESP-specific API updates so your application doesn't break every time Google or Microsoft makes unannounced updates. Nylas syncs terabytes of data and hundreds of thousands of inboxes from ESPs, including Gmail, Office 365, Exchange/ActiveSync, IMAP, Yahoo, and hundreds of others. All of this data has strengthened the Nylas API, which has matured to solve for edge-cases and unique server-specific issues that you'll experience only at scale. By using the Nylas API, your customers have the most seamless experience possible.

The Nylas platform is designed for reliability and future-proofs your app against any unforeseen changes. Our developers are on the front lines of email API updates, maintaining great partnerships with the ESPs your users rely on to conduct everyday business.

How does it work?

The Nylas API powers your app with 100% of ESPs, driving powerful analytics and features for your endusers. We use a simple OAuth flow to authenticate your user's inboxes, after which every email they send syncs automatically with your SaaS app in the background; no need for bcc.

Here's what you get with Nylas

1. Best-in-class API and SDKs
Get up and running with Python, Ruby, JavaScript, and Node.

2. Services
SLAs support your team every step of the way.

3. Business Value
The Nylas email API allows your users to automate sales, marketing, and recruiting efforts, collect invaluable analytics on email data like open rates, click rates, and replies. Emails actually land in the recipient's inbox, not the spam folder, since the IP address is tied to the individual user's IP, not a massive transactional email sender.

Summary: Building with an API

PROS

- ✓ Go to market faster
- ✓ Lower management costs as you scale
- ✓ Dedicated support team and onboarding engineer
- ✓ Security
- ✓ Low risk of failure

CONS

- ✗ Not a turnkey solution — still requires some engineering time to implement
- ✗ If you/your customers use only Gmail, the Gmail API is fairly simple to integrate with
- ✗ Not applicable if you sell only on-premises (APIs are better for cloud solutions)

6 Questions for CTOs and Product Managers to future-proof your company

1

Is there an urgency to build email and calendar functionality into my application?

Am I seeing competitors building this functionality or are our customers and prospects looking for a more integrated solution?

2

What is the total cost of ownership for integration (including engineering headcount, maintenance, infrastructure, customer support)?

How does that look 12, 24, or 36 months from now?

3

Is a company goal to increase customer engagement and retention?

Which email integration solution will help me make a more immediate impact on this goal?

4

Long-term vs. short-term gain:

Which approach sets me up for success in the longterm without adding substantial additional resources?

5

What happens if the person who builds and maintains the integration leaves the company?

6

How will this affect our on-call rotation?

If our systems have errors or customers aren't happy, do we need to hire additional headcount dedicated to support and maintenance?

Conclusion

Email continues to be the center of all business processes and it’s growing exponentially year over year. The question, then, is not if an email integration is a useful feature for most SaaS apps, but rather how quickly you can add this functionality. A comprehensive email API is an easy way to build this functionality while freeing up your engineers’ time to focus on other key features.

As Gartner VP of Research Paolo Malinverno says, “We already live in an API economy where CIOs must look beyond APIs as technology and instead build their company’s business models, digital strategies and ecosystems on them.”[10] Learn more about how the Nylas API can add value to your SaaS app by requesting a demo from one of our platform specialists.

Sources

1. Harvard Business Review, “How Smart, Connected Products Are Transforming Companies,” October 2015

2. The Radicati Report, “Email Statistics Report 2015-2019,” March 2015

3. PR Newswire, “Worldwide Email Marketing Industry Worth US$22.16 bn by 2024, Increasing Number of Email Users to Boost Market’s Growth, Says TMR,” April 2017

4. Accenture Report via Accent Technologies, “Why you need to increase sales rep selling time (and how to do it)”

5. Gartner, “Gartner Says Cloud Email is Gaining Traction Among Enterprises Worldwide,” February 2016

6. Microsoft Technical Documentation

7. Gartner, “A Cloud API Manifesto for Integration As A Service,” March 2010

8. Nylas, “How LeadGenius Generated $50M+ In Revenue By Using Nylas APIs,” October 2016

9. Gartner, “Everything You Know About SaaS Security Is Wrong,” June 2016

10. Gartner, “Welcome to the API Economy,” June 2016

Nylas

Nylas.com

Github.com/Nylas

@Nylas