SIMPLIFY ■ STORAGE ■ INTEGRATIONS

SIMPLE STORAGE

LEARN HOW TO SUPER CHARGE YOUR APPLICATION WITH CLOUD STORAGE

TABLE OF CONTENTS

kloudless

ABSTRACT LAYER. REAL RESULTS.

SO, YOUR TEAM IS HARD AT WORK ON THE WORLD'S NEXT BIG APPLICATION, RIGHT?

Maybe it's something that will change the way people communicate; maybe it's to revolutionize the workflow of a leading industry; maybe it's an application to pioneer a new way of exchanging old family recipes. It doesn't matter. You're primed for success as long as your team can put together your vision and deliver it on time.

As ideas are laid out, Kanban boards begin to fill up with Post-It notes, issue tracking software slowly amasses tickets, and now you're left strategizing ways to make development easier and more swift. (Although really any programming language will do.)

One thing you know for sure at this point; your application needs to connect to cloud storage services.

Not just one, though. You need as many as possible.

That's the beauty of your vision for your application. You want your users to sign up for your app and jump right in. Ideally, they'll feel right at home using your application. You don't want to pidgeon-hole your users into using a service they're not already comfortable with. If your developers only build out a DropBox integration, your users that prefer Box or Google Drive are probably less likely to even try out your application.

Never underestimate a user's reluctance to being forced into yet another set of logins and passwords (Including all the follow-up emails. No one likes those.).

One thing is now very clear -- a single integration of one cloud storage service provider in your app won't do. Not only do you need to provide your users with the ability to use any of their preferred services in your application, but you also need these services to all work with each other.

Easier said than done.

Dealing with cloud storage integrations for your application can be a headache. Simply learning the ins-and-outs of a single storage API service can take you weeks to months of work before

your developers are finished building out an integration, much less having them incorporate multiple services. Your development team might be in for over a year of work after reading through documentation, prototyping, testing and bug fixes are finished.

In the fast-paced world of app development, speed to market is as important as anything. By the time you're through building out your functionality, other applications competing for your market space may have launched already, leaving you with the burdensome task of differentiating your product from competitors by introducing even newer functionality or pivoting your business plan; both of which take up more of your company's valuable time and money.

Kloudless' Unified Storage API was built to alleviate all of these issues and allow your developers to focus on more important things like providing your application with the functionality to set it apart from its peers. Kloudless functions an abstraction layer that sits on top of your application and handles integrations to many services by simply coding out once. You can integrate storage functionality for your end users from every major cloud storage service provider at once by simply using our Unified API. It doesn't matter what service your users need, integrating with Kloudless will allow them to access, sync and upload files and documents between all of them.

Quickly build integrations to many cloud storage services like DropBox, Box, Google Drive, Egnyte, and more with our Unified Storage API.

Office 365

salesforce

ShareFile

box

EVERNOTE

jive

Adobe Experience Manager

S3

kloudless

CHOOSING THE RIGHT STRATEGY FOR YOUR INTEGRATIONS

We created this guide to help you navigate through the rough seas of API integration for your application.

Over the course of reading, you will be presented with use cases, helpful tips and tricks, as well as in-depth articles to help your developers understand the underlying functionality of solutions they can use to save time.

Often in the rush-to-market or aim to impress early investors, we make decisions before having a chance to step back and get a birds-eye view of our application's gameplan from inception to market.

1 Determine your use case

The most important question is always: What use case do YOU need to accomplish for your users with your integration?

Maybe you need to allow your users to create an automation rule for a workflow that they are going to be performing over and over again. For example, maybe every time a user uploads a folder to Box, they also need to automatically sync with their Marketo resources.

It's quite possible that your application needs integration for on-demand user activity, such as allowing the user to pick and choose specific files from DropBox to upload directly into your app.

It may be that your application needs access to your user's data for an entirely different purpose, such as security.

2 Consider your user

Let's say you're building a security software app. To secure your user's data in their storage service applications, your application will need access to that data to secure it.

The point of this question is not to second-guess your application's need for storage integration, but to get you to think long and hard about the user who will take advantage of this functionality.

Considering who your user is is of the utmost importance.

Before you go digging in the weeds to incorporate outside services, libraries, and functionality, it's important to think about what kind of users would access this functionality. Is it an individual user, or is it an administrator who is setting a rule for everyone in an organization? Questions like these are key to understanding your application's purpose, as opposed to its functionality.

The two are not one and the same and should never be thought of as such.

kloudless

3 To embed or not to embed?

It is always ideal to embed an integration into your application to control the experience. It is never a good practice to force your users to leave your app and go to a third party, only to have them return to your product environment after.

The upside of not embedding an integration is that your team doesn't have to support the integration. But this benefit is outweighed by the overhead of asking your users to sign up for another service that they may not be comfortable with. By asking them to adhere to your preference of external third party integration means that you lose control of the user experience. Never assume that a user won't side with apathy toward your app when forced into a service they may not already be accustomed to. By doing this, you miss out on valuable information about your users and how you can improve to meet their evolving needs.

The upside of embedding an integration, however, is that your user has less work to do and stays in the environment you have provided. You learn more about them and their needs this way.

You control the experience.

4 Automation or interactivity?

A good example of this would be Slack's add-ons and integrations that are built directly into the app. When you upload a file in Slack, being able to click on the DropBox button is an on-demand capability. If Slack didn't support that integration, users would be forced to use a service like Zapier to define the action that they want to take. This route would make the action automatic, meaning that it is limited to the rules that govern it and stripping it of any a la carte interactivity like having a file automatically download to a user's DropBox when any file is uploaded to Slack.

A huge benefit to embedded integrations is the ability to build more use cases beyond these rules.

When your application's integration user flow is not written in stone, it allows you to have more flexibility and a variety of interactivity--Both of which are pivotal to setting your app apart from competitors.

Users are just people. They will gravitate toward what makes them feel comfortable and in-control of their experience. Some will resist the unknown in favor of the familiar. Give them the ability to pursue either.

5 Identify needed services

Next, prioritize what cloud services your application needs to connect to.

Look to the market leaders in each category. These are table stakes. They are the clear cut minimum that your application needs to support in order to thrive.

Any integrations beyond that will be driven by your end-users' demands. Gather info from customers requesting different connectors and prioritize your product roadmap accordingly.

Choose these integrations based on what will yield the most business opportunity.

Lastly, you need to strategize on competitive differentiation. What do you need to connect to in order to stay ahead of the curve?

Maybe these connections are longtail or simply have some sort of cult following. They may not be the most heavily utilized in their category, but they can help your application stand out among the pack. If you simply add integrations for the services that lead their respective categories, your demand will dry up after a certain pool of customers. Make your product as competitive as possible by integrating with these smaller and more niche alternatives. Their rabid fans will thank you, and for all you know, these services may become the industry leaders down the line.

6 Build smart

None of these questions can be answered without forcing you to step back and evaluate your application's real purpose.

That's really the entire point of this guide. We can't provide you with one clear-cut, industry-standard roadmap toward your integration strategy because it doesn't exist. Each and every application with a need for integrations has a different use-case and different end-users.

What we can provide you with are examples of success, ways to go about incorporating functionality, and specific use cases for our products in relation to your application. This is the time to search out the tools and make the right development decisions that will help your vision come to life and flourish.

Cutting down on the time and resources consumed by your dev team should be a top priority. Tools such as Unified APIs that allow for a connection to many services with the coding equivalent of connecting to a single service can not only save you time and money on development, but also protect you from unexpected roadblocks in the future should your integration services change the nature of how you connect to them. On top of that, the ability to give your users a veritable smorgasbord of services at once is something that can push your app over the edge when it comes to differentiating it from its competitors.

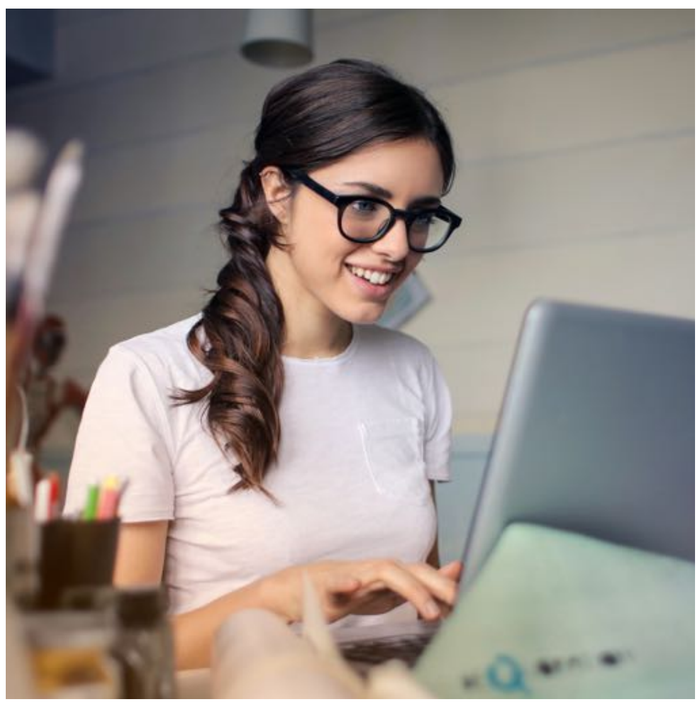
The real work isn't what libraries or languages you choose to build with. It's not even about the specific connectors you choose to go with initially. It's about the strategy you take to make your application stand out. What makes it unique. What makes it comforting. What makes it desirable.

We look forward to what you build.

kloudless

LESSONS LEARNED WHILE MOVING BEYOND BESPOKE INTEGRATIONS

Learn how one company used the Kloudless Unified Storage API to relieve their cloud storage integration woes and empower education.



Recently an educational technology company, Blackboard, signed up for Kloudless in order to move beyond its inherited, bespoke integration strategy.

First, here's some background on the company. Blackboard supports both educators and students in schools, universities, and higher education institutions. Founded in 1979, Blackboard is among the first to pioneer the space, exploring and innovating new methods to enable technological methods of instruction. Blackboard's suite of educational products equips teachers and administrators with the tools to ensure that their students have access to a safe, secure, and engaging place to learn.

With over 90 million users across thousands of institutions worldwide, Blackboard needed an integrations solution that was effective, secure, and scalable. Students and educators share files in and out of the classroom. Blackboard's flagship product – let's call it EdTech Learn – is an educational platform built for both teachers and students to communicate and collaborate outside of the classroom.

Yet even with the full suite of platform features, Blackboard Learn was still just one of the many software services found

kloudless

in the average educator's technology stack. Educators and students alike would store their files, assignments, and other assets in any number of popular cloud storage services, such as Google Drive, Dropbox, and Box. Blackboard's users wanted to be able to access, upload, and edit those files within Blackboard's own platform without their workflow being interrupted.

The Problem

Version upkeep for multiple cloud storage services proved challenging. The product team at Blackboard had inherited a bespoke integration—Dropbox—and initially sought to build out the rest of the popular file storage services the same way: internally. They started with Microsoft OneDrive, a popular request, but soon ran into a number of issues.

To their surprise, the initial integration was not what required the bulk of their time and effort: it was the maintenance of the integration that proved to be a headache.

Over three years of maintaining their bespoke integration, Blackboard received more than 600 JIRA tickets and racked up approximately 8,800 hours towards OneDrive support alone. Unpredictable changes that Microsoft made to the OneDrive API would interrupt Blackboard's automated scripts, breaking their meticulously designed integration infrastructure. Each issue would remain open for anywhere from two to four weeks before it was fully resolved. It was starting to distract engineering attention away from the core product, and even began to hurt the confidence that their customers and users—especially the newer ones—had in the Blackboard platform. It was becoming painfully clear that Blackboard needed to find a scalable solution that did not involve building and managing each integration in-house.

The Solution

To solve the problem, Blackboard used Kloudless to code once and integrate many services. The Blackboard team needed a solution to manage their integrations efficiently and effectively without diverting resources from their core product. After evaluating several options, including building directly from the API providers, they selected Kloudless as their new integrations solution. The clean and intuitive developer experience coupled with the depth of features enabled

kloudless

> *"With Kloudless, Blackboard is able to save up to $150,000 and 5,000 hours of developer time a year, while simultaneously ensuring reliable access to the multiple software services they need."*

Blackboard developers to quickly and efficiently develop a proof of concept that brought clarity to their decision.

Blackboard uses the Kloudless File Storage API to integrate Blackboard Learn with cloud storage integrations like Box, Dropbox, Google Drive, OneDrive, and SkyDrive. The "one-and-done" approach allows Blackboard to access multiple applications and in a fraction of the time while drastically reducing the need to consistently monitor their integrations for breaking changes.

The open source nature of Kloudless allows Blackboard to take advantage of customizable UIs to scale internationally, prompting users in various languages to select their files and folders in the Kloudless FileExplorer user interface.

Managing Integrations On-Premise

Blackboard manages integrations on-premise. Doing so ensures the security of data across integrated services.The flexibility of the Kloudless Enterprise on-premise deployment allows the team to easily scale up the instances they needed while simultaneously ensuring that their sensitive data—students' personal information—stays safe and secure. The freedom of scaling even allowed them to successfully reach their product international expansion goals.

Using a unified API saved Blackboard substantial resources. Months later, Blackboard continues to see meaningful benefits from their investment across several teams and functions. Blackboard estimated that maintaining a single in-house integration took up to one third of a full-time engineer's time, dedicated solely to troubleshooting integration issues for that one service alone. Since Blackboard's core competency as a product is not centered around integrations, it didn't make sense to block the development cycle of the entire product by tasking developers with the challenge of building an integrations solution from scratch.

Kloudless allows Blackboard developers to stay on track and focus on what they're good at: improving and expanding their core educational technology product. With Kloudless, Blackboard is able to save up to $150,000 and 5,000 hours of developer time a year, while simultaneously ensuring reliable access to the multiple software services they need.

Integrating with multiple services also helped Blackboard catch up in the market. As integrated workflows became more widely adopted, Blackboard saw their competitors expanding their integrations offerings to meet user demand. After integrating with Kloudless, they quickly caught up to—and even surpassed—the industry standards by adding five fully functional and stable integrations in mere days after the initial implementation.

With the breadth and depth of integrations supported by Kloudless APIs, Blackboard is considering countless opportunities to enable creative new use cases for their user base to set themselves apart from their competitors. These results are in line with those experienced by other companies across industries. On average, Kloudless customers deploy applications eight times faster and decrease their maintenance overhead by a factor of four. What's more, Kloudless customers code once and connect additional services twelve times faster than if they had built the integrations from scratch.

kloudless

YOUR USERS
ARE WAITING.

MAKE YOUR DEVELOPER'S LIVES EASIER.

> *It takes time and money to build out integrations, and you're not in the business of wasting either. Take away the pain of studying documentation, testing, and fixing bugs from the process of implementing your application's much needed scheduling functionality.*

> *The worst challenge was keeping bespoke integrations up-to-date. Having access to Kloudless APIs to integrate with a single set of APIs greatly simplifies our work*

Brent Mundy

Sr. Director or Product Management
@ Blackboard

Blackboard

kloudless

WHAT TYPE OF STORAGE IS RIGHT FOR YOUR APPLICATION?

Depending on your application's needs, you may want to incorporate object storage such as Amazon S3 as opposed to file storage, which is the more traditionally thought of service when it comes to cloud storage. Read on to get a high-level breakdown of object storage and how it may end up being exactly what your application needs to manage its user's data.

Depending on your application's needs, you may want to incorporate object storage such as Amazon S3 as opposed to file storage, which is the more traditionally thought of service when it comes to cloud storage. Read on to get a high-level breakdown of object storage and how it may end up being exactly what your application needs to manage its user's data.

When many of us think of cloud storage, we envision the entirely UI-based services that we use to share files at work or keep our documents safe in the cloud. Dropbox, Box and Google Drive are a few examples of these services that most of us are probably familiar with, and we refer to them as file storage. Generally, we use them to sync files across multiple computers, collaborate and

share documents, or possibly we just house our personal files on these platforms when we run out of room on our computer's hard drive.

File storage is usually enough for a single user or non-data-driven company to store and share documents, as it conventionally uses a folder structure hierarchy that requires a very specific set of paths for locating files. This folder structure makes it very easy to compartmentalize files or documents in a way that can be easily remembered, but when the data that we house grows substantially, this means of pathing to retrieve files can be cumbersome and unmanageable. Imagine having to click through hundreds of folders every time you needed to retrieve a file, and you start to understand why when the amount of files grow substantially, another method may be a preferred course of action.

Object storage, on the other hand, allows for an entirely different means of storing files that isn't burdened by a prior knowledge of the physical location of the specific file. Object storage uses something called flat address space to allow for the infinite scalability of storage, and principally uses an HTTP-based REST API to access the data contained therein. Object storage also allows for extensive levels of metadata to be associated with each object, permitting us to store whatever relevant data we need coupled with our files. While file storage generally only allows us to store a finite amount of associated data for each file, object storage will store massive amounts of associated metadata for each file, allowing us to keep track of almost any associated attribute a file may have.

Retrieval of these files is done by assigning each object stored in our flat address space with a unique identifier, or more commonly referred to as an ID, and then requesting the file directly. Object storage allows us to store a large amount of files or data in an object storage service provider without worrying about an overly complicated folder structure that we have to navigate every time we need to access or retrieve a file. Essentially, we have something in place to locate our files for us, as opposed to having to continuously click on sub-folders to locate our files ourselves.

Since Amazon Web Services' S3 (simple storage service) became the de-facto provider of cloud-based object storage, other services such as Google Cloud Services and Microsoft's Azure have become S3-compliant. No matter the service you choose, if it's compatible with S3, Kloudless can help you and your users to access and save files in object storage. To read more in depth about the benefits of integrating file storage VS object storage, please check out this article from our Kloudless CTO, Vinod Chandru.

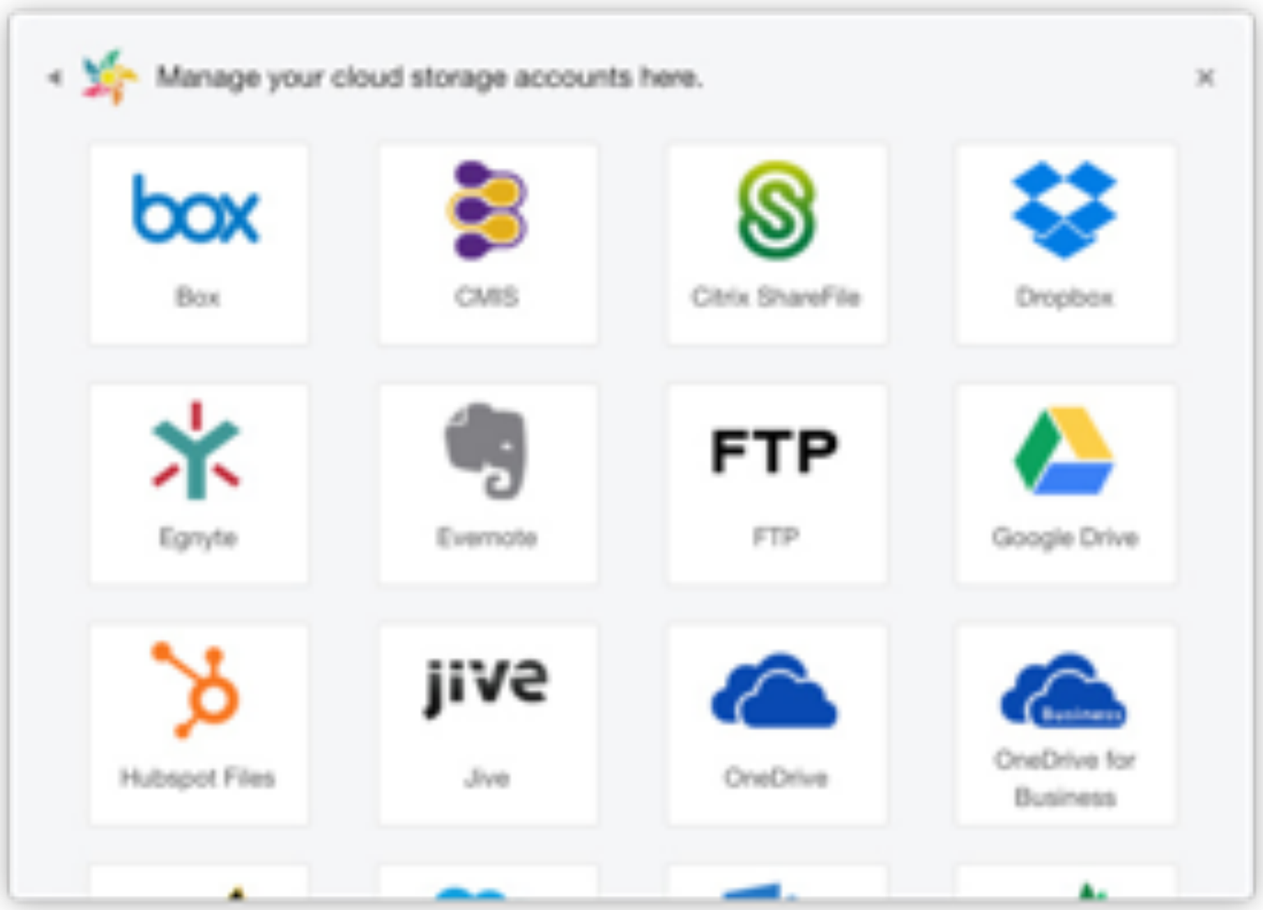
INTEGRATING OUR OPEN-SOURCE FILE EXPLORER INTO YOUR APP

Kloudless provides a free open-source, easy-to-use JavaScript UI tool to embed into your application so users can seamlessly schedule meetings.

Now that we've covered a high-level real-life use case, let's dive into the nuts and bolts of some of the features of the Kloudless Unified Storage API. The Kloudless File Explorer is an open-source JavaScript library that allows your app's users to to have complete control over their files and documents. To take some of the pain away from your developers having to build their own UI tools, we have built this easy-to-implement, embeddable interface that will enable full CRUD functionality in your application right out of the box. With this easy-to-implement library, you can give your users a native file browsing experience with just 2 lines of code.

Let's take a minute to go over how it works and see how simple it is to get your users browsing, uploading, and manipulating their files and documents quickly and easily!

kloudless

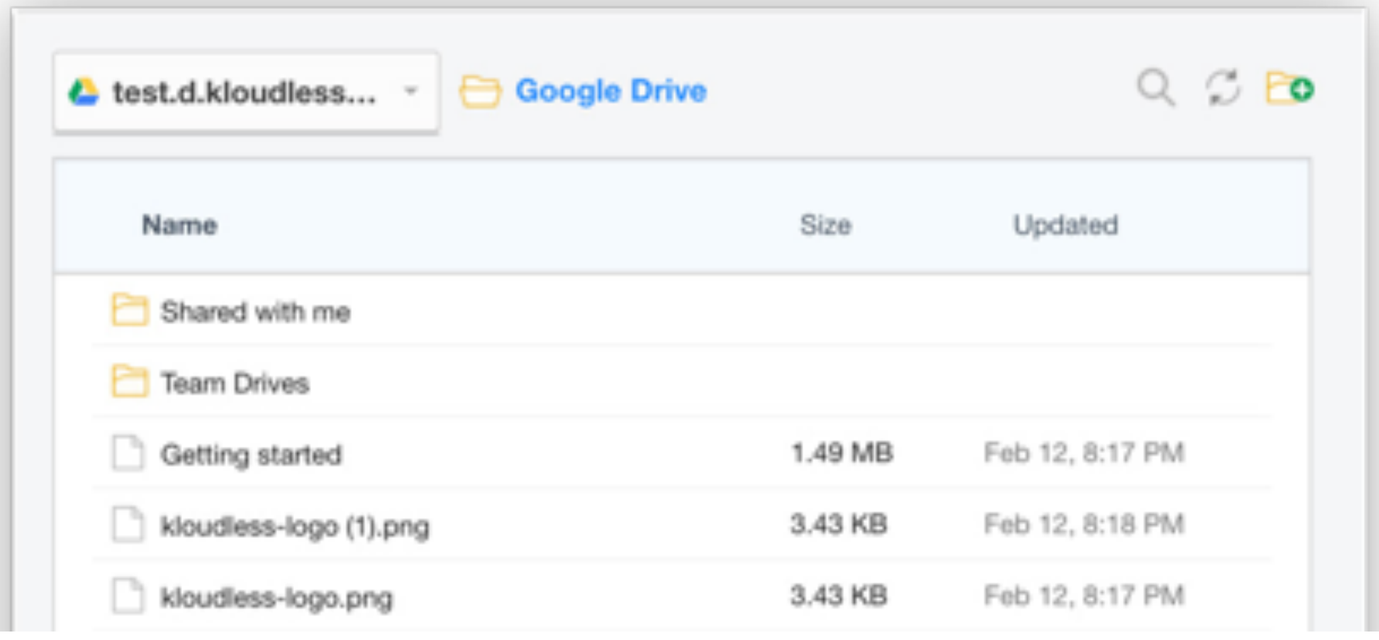


Manage Your Cloud Storage

First, a user is presented with a list of available cloud storage services. For example, in the case of wanting access to their Google Drive files, a user would click on the tile marked with the Google Drive logo. They will then be taken to a screen prompting them to log in to their account. If you wish to, you can whitelabel this screen so your own application's logo is presented as asking for permission to access their account, leaving them unaware of Kloudless' role in the process.

Connected Accounts

Once authenticated, their account for that service will be linked at the top of the File Explorer, allowing them access to enter into a UI dropdown of their folders, files and documents. The user is now free to go about uploading, updating or accessing any of their files stored in their linked service. Users may authenticate with as many services as they wish, giving them full CRUD functionality over all of their files across any authenticated cloud storage services.



The File Explorer currently integrates with all of the available cloud storage services provided by the Kloudless Calendar API.
Getting started is as simple as having your developers embed a script tag into your code.
Read more about the possibilities that your application can take advantage of in our in depth article about the Kloudless File Explorer here.

kloudless

CUSTOMIZE YOUR APPLICATION'S NEEDS FOR THE KLOUDLESS UNIFIED STORAGE API

Learn useful ways to go beyond the built-in UI tools and give your application whatever functionality your users need. In this section, we will dive deeper into some of the functionality you can harness from the Kloudless Unified Storage API.

The Kloudless Unified Storage API offers an incredible depth of functionality for your developers to take advantage of beyond our File Explorer library. For the sake of your application's individual needs and requirements, we provide the tools necessary to build out any cloud storage support that your application requires. Do you need to import your users files in the cloud into your app? Does your application need to store user-uploaded content on Amazon S3 or s3-compatible services? What if your app needs to track changes across Google Drive?

The following articles can help you to learn more about these specific tasks, as well as provide an in-depth look at the code and functionality under the hood of our powerful Storage API. Empower your application's storage services by learning more about the specific functionality provided from each of our endpoints in the Unified Storage API.

In the next few pages, we will briefly go over pertinent articles from the Kloudless blog, and link you to them for further reading!

kloudless

IMPORT YOUR USERS FILES IN THE CLOUD INTO YOUR APP WITH THE KLOUDLESS FILE EXPLORER

Transferring your users files from their cloud storage provider to your app's backend is usually a tedious process riddled with excessive server-side logic. Make your developers lives easier with our simple tips to automate the process!

The Kloudless File Explorer comes with an assortment of additional options to automatically copy data to an app's configured backend storage, effectively taking the hassle out of your developer's hands. Kloudless can handle all the steps required behind the scenes to ensure that your users file transfer is successful. This includes shifting from regular uploads to multi-part uploads for large files, varying connection options and protocols depending on the upstream storage service, refreshing OAuth tokens, and handling errors and rate limits.

Read more here.

kloudless

STORING USER-UPLOADED CONTENT ON S3-COMPATIBLE SERVICES WITH KLOUDLESS' FILE EXPLORER

All cloud storage is not created equal, and many factors can influence which type of storage your application decides to go with. Thankfully, here at Kloudless, we have added the ability to use S3 and S3-compatible object storage alongside our already expansive file storage capabilities.

When choosing the right cloud storage service to integrate into your application, you must first decide which type of storage best fit your needs. Does your app simply require the ability to share internal documents among coworkers, or are you looking to store thousands of user-uploaded pictures and high-quality video files? In this article, we will cover the reasons why you might choose to use Amazon S3 or S3 compatible storage for your application, and then walk through the simple steps your developers can take to get it up and running.

Read more here.

kloudless

HOW TO TRACK CHANGES IN GOOGLE DRIVE, INCLUDING IN TEAM DRIVES

Developers commonly integrate with cloud storage services such as Google Drive to sync files that change between their app and users' cloud storage. Learn how the Kloudless Events API can automatically track those changes for your application.

The Kloudless Unified Events API enables your developers to track changes regardless of which cloud storage account a user connects. Google Drive remains one of the most popular services users connect accounts for, and as such, is one of our more requested storage options to outline this process in. In the following article, we'll discuss the various ways to track changes in Google Drive and when your developers should implement each one.

Read more here.

kloudless

WHERE DO WE GO FROM HERE ?

Kloudless Unified Storage API can handle your needs at the moment, as well as ensure that you are covered for the future.

Hopefully you've started thinking about the benefits of incorporating the Kloudless Unified Storage API into your application from reading this eBook. At its base level, Kloudless can help your developers integrate as many of your user's storage services that your app needs in one fell swoop. Instead of dealing with the headache of learning the ins-and-outs of multiple services and the problems that may arise from their inclusion into your application, your developers can code to access a single API: Kloudless.

A common problem that sidelines applications is dealing with unforeseen circumstances. Kloudless Unified APIs aim to alleviate the pain of these issues by making sure that your application doesn't have to suffer from downtime or broken functionality as a result of factors that are out of your control. When APIs change the way they return data, we handle that. When documentation requires that your developers learn a new function of the required storage API, we handle that. When a new version of the API launches, we handle that. Essentially, all your future needs are handled after you integrate our Unified API into your codebase. We take the legwork out of API integration, so that your developers can focus on more important things; like the functionality your application needs to set itself apart from the competition.

We also add new storage services to our APIs all the time, so if a newer and desired service comes along that your users need, chances are, we're hard at work incorporating it into our Unified API so you can allow your users to access it with no work at all on your part.

Because, at the end of the day, file storage functionality isn't what makes your application special or different. It's a necessary piece of functionality in your greater vision, but it's not what you should be endlessly laboring over or getting behind schedule on.
Your application is special because of what it does differently. You're looking to change minds; change how industries work; maybe even change the world. Let your developers focus on those things, while we focus on your cloud storage integrations.

We're looking forward to what you're building, and we can't wait to help your goals materialize quickly with our Unified APIs.

kloudless

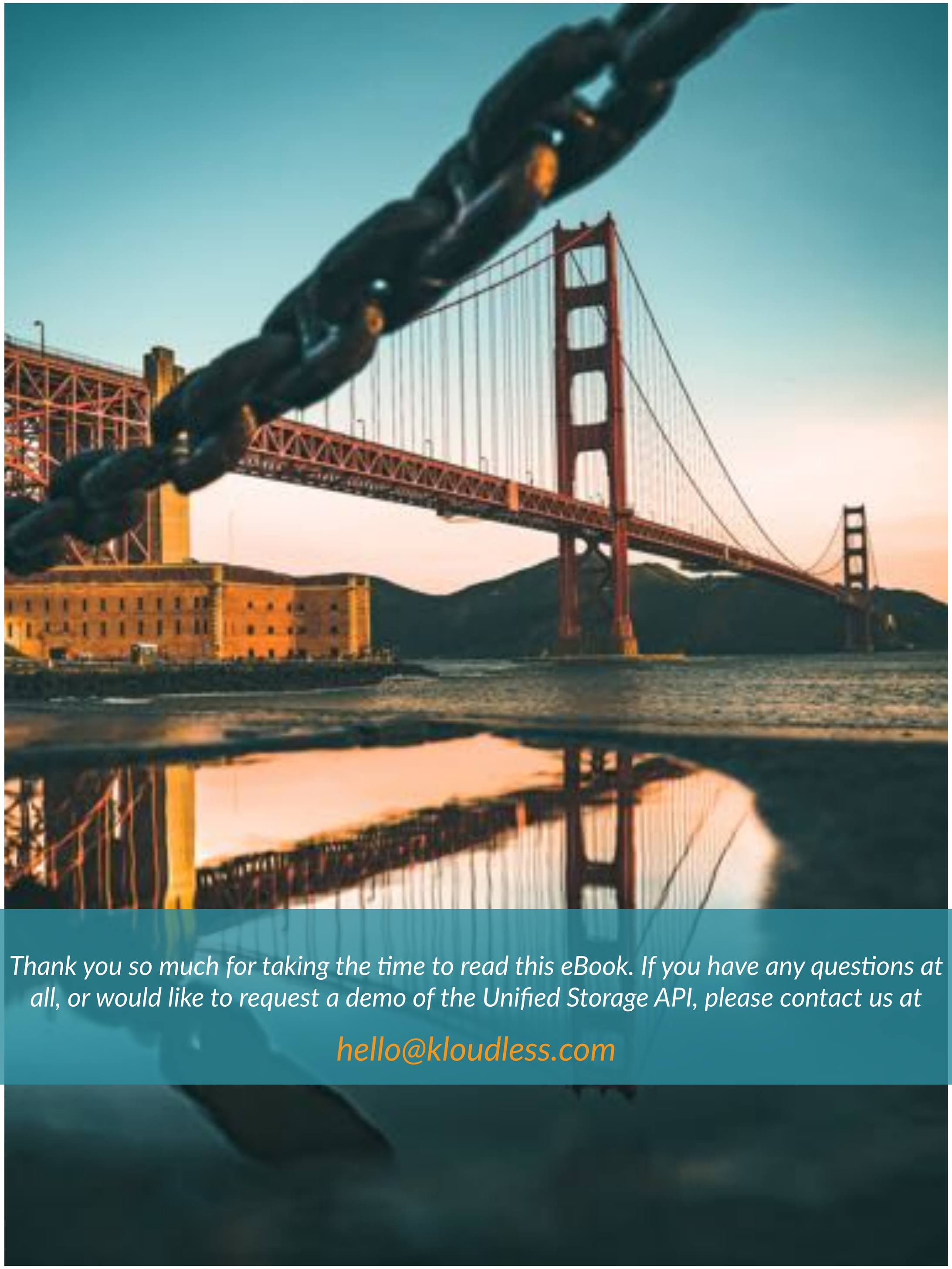
Built by developers, for developers. All the cloud storage features your devs need from CRUD to real-time webhooks. Upload and download from any storage service. Find files from all connected storage services. Get webhook notifications on any file or folder.

Simplify your integrations. Because you have more important things to worry about.

Write code once and reuse it for any service

Kloudless is an abstraction layer on top of the most popular calendar services, so you can work with one API, not five. We take care of all the Connector maintenance and updates, so your devs don't have to.

kloudless



Thank you so much for taking the time to read this eBook. If you have any questions at all, or would like to request a demo of the Unified Storage API, please contact us at

hello@kloudless.com

kloudless