# Optimizing Your Internal Developer Platform with MongoDB Atlas

MongoDB

# Table of Contents

# Summary

It's no secret that there has been a lot of change happening across the developer landscape. While the proliferation of tools and cloud technologies has given developer teams greater choice in how they want to build, it has also created complex developer toolchains that have become difficult to manage. This has led to growing levels of friction and cognitive load for developers.

As a result, implementing Platform Engineering teams and Internal Developer Platforms (IDPs) within developer organizations is becoming a popular solution for reducing cognitive load and increasing developer productivity. While IDPs reduce complexity for developers, choosing which tools should go into an IDP can be a daunting process for platform teams–especially when it comes to an IDP's data layer.

This white paper provides an overview of how MongoDB Atlas, MongoDB's developer data platform, can make that choice easier and showcases how you can easily incorporate MongoDB into an IDP using the tools your developers are already familiar with.

# Building Out the Optimal Internal Developer Platform

DevOps establishes a culture of incorporating processes and tools designed to deliver applications and services to users faster and with more control than the traditional software development process. While DevOps practices have significantly improved how developers build applications, the proliferation of new technologies and architectures has added complexity to modern cloud-native setups. These shifts have contributed to developer cognitive load and created wider organizational risk. Developers now face the need for an in-depth, end-to-end understanding of their toolchains, and the increased levels of access controls to tools available across developer organizations can lead to compliance issues, including inconsistent security controls and incorrect reporting.

IDPs and platform engineering are quickly becoming popular solutions to address these issues. IDPs are bespoke toolchains built and managed by platform teams, used to create greater developer self-service. IDPs lower toolchain complexity through standardization and reduce the operational expertise required to build or maintain them, as that responsibility is passed on to the platform team. This handover enables developers to focus on building applications, which not only improves developer productivity but also enhances the developer experience. As a result, developer organizations experience greater collaboration, satisfaction, engagement, and retention.

While each IDP looks different from team to team, there are a few common components–such as a CI/CD pipeline or an infrastructure management tool. Whatever tools go into an IDP, it's important that they contribute to enabling greater developer self-service.

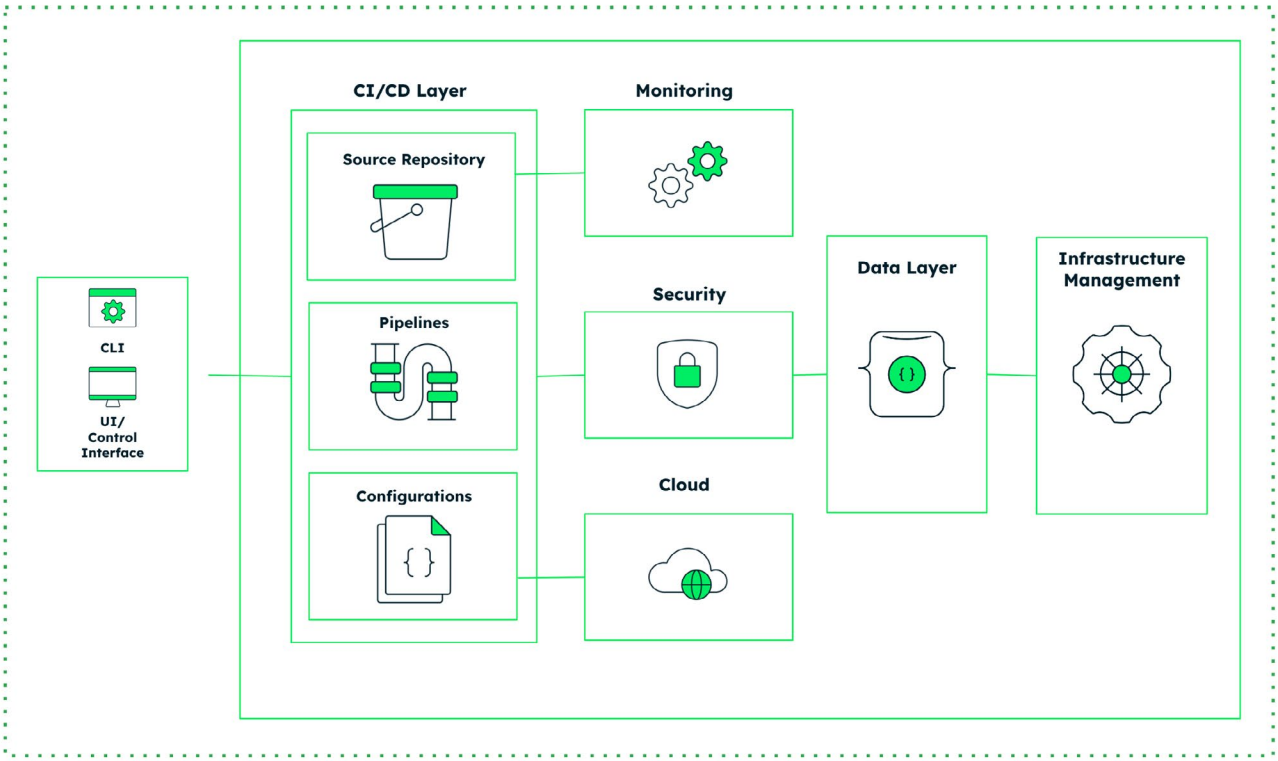


Image 1: Example of an internal developer platform a team might use

Though IDPs contribute to greater developer productivity, incorporating the right tools into them can be a challenge for platform teams –especially when it comes to an important component of any IDP: the data layer. Competing stack preferences across developer teams, as well as data solutions that are incompatible with or hard to incorporate into existing developer toolchains, make choosing the right tools a daunting task. As a result, most platform teams resort to incorporating a number of data point solutions into their IDP, creating a sprawl of infrastructure. This makes both the data layer and, consequently, the IDP as a whole harder to build out, maintain, and scale. It also causes disruptions in existing workflows, which could potentially lead to developer dissatisfaction and cause them to resist platform adoption altogether.

All of this defeats the purpose of introducing a standardized IDP in the first place. Therefore, it's critical that any data solution selected for an IDP is easy for platform teams to incorporate and enables the developers using it to address a breadth of data workloads with ease.

## MongoDB Atlas in your Internal Developer Platform

This is exactly what sets MongoDB Atlas, MongoDB's developer data platform, apart from other data solutions. With Atlas, platform teams can reduce the complexity that comes with building out an IDP's data layer and eliminate the need to add a multitude of data point solutions.

Atlas integrates all the data services you need to build modern applications into a unified developer experience. It accelerates time to value and reduces complexity through a consistent interface (Query API) that standardizes operations to drive innovation at scale. It handles transactional workloads, full-text search, AI-enhanced experiences, stream data processing, and more, all while reducing data infrastructure sprawl and complexity. Atlas's cutting-edge and comprehensive controls ensure data security and privacy, intelligent performance optimization, and global and multi-cloud deployments, protecting developer organizations from risk and compliance issues. It not only prioritizes an intuitive developer experience but also integrates easily into your development and deployment workflows to simplify provisioning and management.

Most importantly, Atlas aims to meet developer organizations where they are. That includes being easily incorporated into existing development stacks through the workflows and tools familiar to developers.

## How to Incorporate MongoDB Atlas into your Internal Developer Platform

Most tools in an IDP are connected through APIs and are exposed to developers through simplified configuration and management interfaces. MongoDB Atlas can be easily integrated into existing IDPs through the Atlas Administration API.

The Atlas Administration API is a RESTful interface that allows interaction with Atlas resources and the performance of various actions in MongoDB Atlas–either directly or using one of the many tools that leverage the API. Developers can interact with the Atlas Administration API and downstream tools to directly manage and automate various aspects of MongoDB Atlas, including resources such as clusters, database users, and backups, to name a few.

MongoDB Atlas can be integrated and controlled using tools that likely already exist within your developers' toolchain. Three commonly used tools built on the MongoDB Atlas Administration API enable developers to interact with Atlas: the Atlas Kubernetes Operator, the HashiCorp Terraform Atlas Provider, and AWS CloudFormation. Each of these tools allows teams to incorporate MongoDB Atlas into their IDPs through the method(s) of their choice. All of these options are available to Atlas customers free of charge.

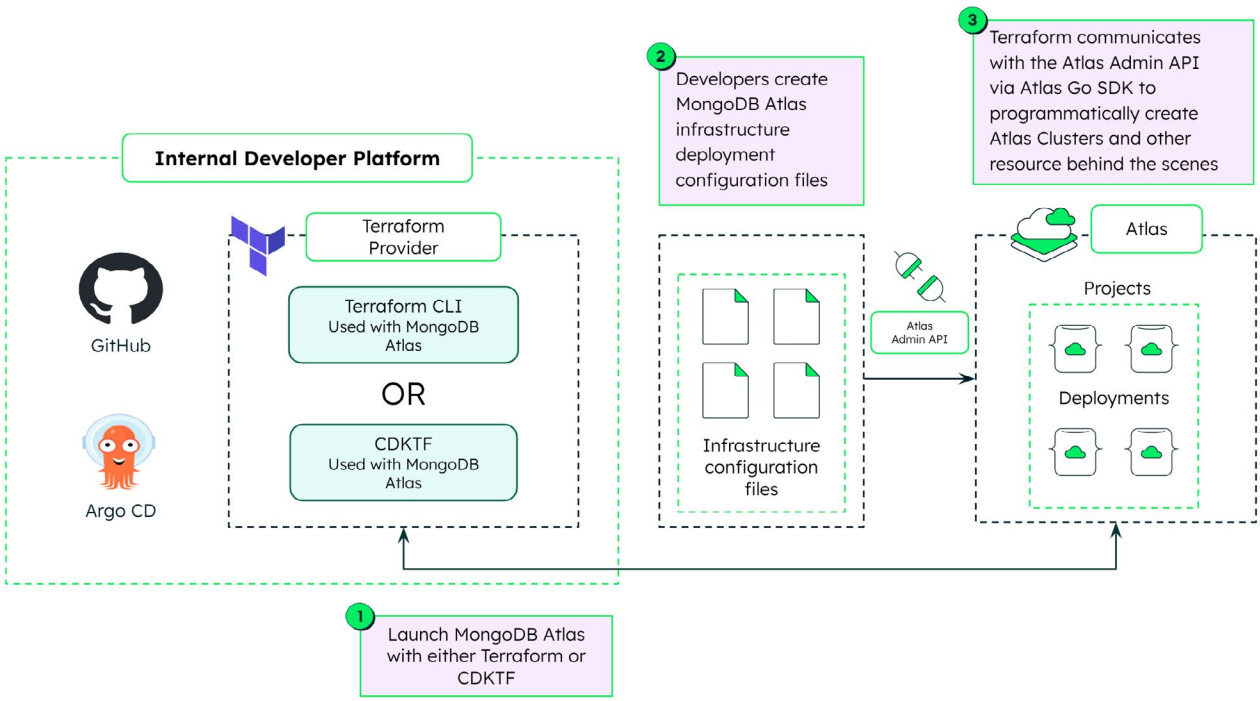


Image 2: Incorporing MongoDB Atlas into your developer toolchain using the HashiCorp Terraform MongoDB Atlas Provider

## Atlas Kubernetes Operator

Kubernetes has become a very common foundational layer for many IDPs, not only due to the rising popularity of containers but also because of the significant automation and flexibility it offers.

MongoDB provides a way to work natively with any certified flavor of Kubernetes–the Atlas Kubernetes Operator. The Atlas Operator enables users to manage Atlas with the same tooling and processes they use for their existing services in Kubernetes. It works by extending the Kubernetes API through custom resources, allowing developers to manage Atlas resources as native Kubernetes objects. This means your developers can deploy configuration files for Atlas via a CD tool, like ArgoCD, into Kubernetes, and the Atlas Kubernetes Operator automates the application of that configuration via the Atlas Admin API.

The Atlas Kubernetes Operator is an excellent tool for incorporating Atlas into a Kubernetes-based IDP for several reasons. The Operator helps reduce operational complexity, as developers only need to manage the custom resource configuration files in a Git repository and apply them to Kubernetes through an existing CI/CD pipeline. Developers can also standardize the management of Atlas in Kubernetes using developer templates provided by MongoDB for Atlas deployments, which can be easily adopted and used. The Operator can manage a large number of Atlas deployments consistently, in or across any of the three major cloud providers: AWS, GCP, or Azure.

## HashiCorp Terraform MongoDB Atlas Provider

Infrastructure as code (IaC) is a foundational practice in IDPs, enabling automated, consistent, and scalable management of infrastructure resources through code, which streamlines operations and enhances development agility.

The HashiCorp Terraform MongoDB Atlas Provider allows Terraform users to easily manage their Atlas lifecycle by managing Atlas infrastructure as code in HashiCorp Configuration Language (HCL). The Terraform Provider gives developers the ability to provision infrastructure in a way that is easy to manage and follows a standardized workflow they are comfortable working in. Customers using our official plugin also have access to customer

support. Users can leverage the Terraform Cloud Development Kit (CDKTF), instead of the Terraform Provider directly to provision infrastructure using the language of their choice (JavaScript, TypeScript, Python, Java, Go, and C#), rather than having to use HCL.

Using Terraform to incorporate Atlas into your IDP has several advantages. Atlas's native integrations with Terraform and the CDKTF enable developers to have workflow- and language-specific methods for incorporating Atlas into existing CI/CD pipelines, providing a more developer-friendly way of using Terraform. Terraform can be used to manage a large number of MongoDB Atlas deployments consistently, in or across any of the three major cloud providers: AWS, GCP, or Azure. Developers using Terraform also have access to a large Terraform community, repositories, and quickstarts, which can help them get started and troubleshoot faster.

## AWS CloudFormation Resources

AWS CloudFormation is an IaC tool offered by AWS. It enables users to provision and manage Atlas infrastructure as code through AWS-native workflows. This tool is ideal for teams whose developers prefer to work with AWS-native tooling within their IDP. MongoDB Atlas offers users multiple resources from CloudFormation, including:

- JSON/YAML resources from the CloudFormation Public Registry which can be executed via the AWS CLI / AWS Management Console. This is a great option for teams comfortable working in JSON/YAML and want to work in a way that is tightly ingrained with AWS.

- AWS's Quick Starts, formerly known as Partner Solution Deployments which create an Atlas project with a standard, single-Region, M10 cluster and enable users to automatically set up a MongoDB Atlas environment in AWS. This method is great for teams who want to quickly and easily provision multiple resources at a time using templates that have been reviewed by AWS and MongoDB Atlas.

- The AWS CDK which makes it easier for developers to define and manage their Atlas infrastructure natively in their programming language (JavaScript, TypeScript, Python, Java, C#, and Go. ) of choice.

Using AWS CloudFormation to incorporate Atlas into your IDP is an excellent option for teams that are most comfortable working with and have their IDP built with AWS-based tools and workflows. AWS CloudFormation can be used to manage a large number of MongoDB Atlas deployments consistently within AWS and gives developers the flexibility to work in a manner that best fits their team's needs.

# Accelerate Developer Velocity with Atlas and your IDP

Achieve even greater developer autonomy and application output speed by leveraging Atlas, MongoDB's developer data platform, within your Internal Developer Platform. You can get started with creating a MongoDB Atlas account here.

You can learn more about the DevOps tools we integrate with here:

- Atlas Kubernetes Operator
- HashiCorp Terraform MongoDB Atlas Provider
- AWS CloudFormation

To start using any of our DevOps tools, you can go to the documentation here.

Additionally, you can watch a demo of how to integrate each of these tools into your IDP here.

Go build Atlas into your IDP today!