# Let’s talk about pipelines in the post-AWS CodeCommit Era

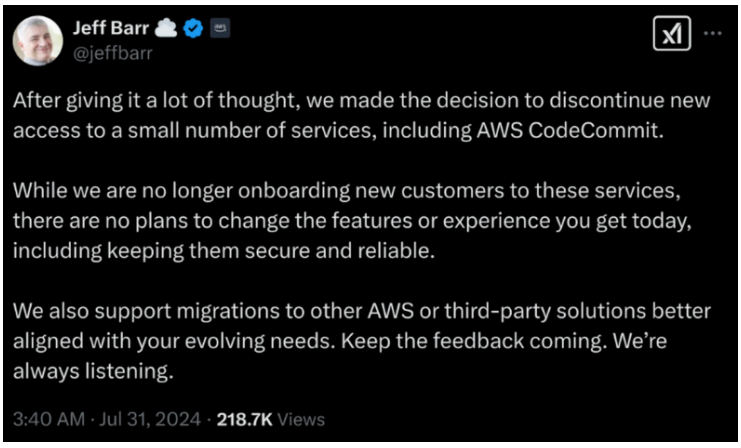
*26 February 2025 - 7 min. read*

AWS CodeCommit | CI/CD | GitHub | GitLab | Landing Zone

The DevOps philosophy and the evolution of software development led to the automation of application build and deployment operations. Thanks to application and infrastructure pipelines, developers no longer carry USB drives to the system administrators' office to deploy the new software release!

For those of us who develop cloud-native applications on AWS every day, the first step to kick off a project is creating a Git repository.. Since July 9, 2015, the most natural step has always been to click on the CodeCommit section of the AWS console and select “Create Repository.”

Just over 9 years later (on July 31, 2024), we learned that this habit would have to change following Jeff Barr’s announcement on X.




The news surprised everyone: essentially, new AWS accounts created after July 31, 2024, will no longer be able to access the service, however, accounts in AWS

organizations where repositories were already present will still be able to access the service normally.

The service is not in the process of being decommissioned, as was hypothesized in the days following the announcement; security updates, bug fixes, and maintenance are guaranteed, but no new features will be developed.

Although AWS CodeCommit has never provided advanced options for collaborative coding, its strength has always been its integration with the various automation and deployment tools offered by AWS (like CodePipeline, CodeDeploy, and Amplify).

There's already been a lot of discussion about the reasons that may have led AWS to make this decision. Our goal, instead, is to explore the different available alternatives, highlighting the pros and cons based on various scenarios

## Why AWS CodeCommit: its strengths

A key aspect is its integration with IAM, which allows you to manage Git permissions using already-registered users: defining the allowed Git operations (pull, push, merge, etc.) directly within IAM policies, making it quick and easy to manage.

You can also create notification and approval workflows directly integrated with the AWS Cloud (like Amazon SNS, Amazon EventBridge). On the other hand, switching to a different Git provider might require creating integrations that are no longer directly configurable on AWS.

Moreover, the service's cost is incredibly low, enabling anyone to get started without navigating a maze of features offered by countless different usage plans.

Having governance over AWS ecosystem services in one centralized point, federated with the corporate Identity Provider, including management of development tools, provides a holistic view, simplifies processes, and facilitates the implementation of a Landing Zone. The integration of CodeCommit with IAM is a key piece in putting together the puzzle of permissions and access management.

## Weaknesses

Despite all the positive aspects described, it was undeniable that AWS CodeCommit had some pretty evident limitations.

The first was the limited number of integrations with tools outside of AWS. Nowadays, it's crucial for a code repository tool to offer various integrations beyond just managing branches and resolving conflicts. Almost all the main vendors now provide solutions directly integrated with third-party products, offering users a seamless 360° experience on their platforms.

The second major limitation was the poor implementation of features for collaborative development and code reviews. While the possibility of performing 'pull & merge requests' was available, its implementation was quite basic (not to mention the user experience). Today, collaborative development tools offer much more advanced user interfaces, alongside a vast range of complex and advanced features for code review and comparison.

AWS probably never intended CodeCommit to directly compete with specialized collaborative development tools. Instead, it aimed to provide an integrated option in the context of centralized governance of accounts and resources. And that's precisely why the sudden announcement caught us off guard.

## Alternative codebases: GitHub, Gitlab, Bitbucket

"As we've already mentioned, AWS CodeCommit is no longer available for newcomers to AWS, but the service isn't being discontinued: there's no need to panic or come up with emergency strategies, but it's still a good idea to start evaluating the available alternatives.

First, we need to figure out which of the many options best suits our needs and resources.

AWS directly recommends some vendors, including the well-known GitHub, GitLab, and BitBucket. The main driver behind this suggestion is, once again, the integration between these platforms and AWS services. The deciding factor might boil down to one simple question:

'What's the actual use case for my repository?'

In the next section, we'll try to match the most common answers to this question with the various products available on the market.

There are very few services (both AWS and non-AWS) that the three platforms mentioned above don't integrate with. In fact, all three can be configured directly as a source step for AWS CodePipeline through the creation of a CodeStar connection. For both GitHub and GitLab, you can use either the SaaS or self-hosted version.

If we decide to use one of these solutions, the question naturally arises: 'Do I really need AWS CodePipeline?'

If we previously used AWS CodeCommit because it was convenient and right there in the same AWS CodePipeline console, now that we'll have to use an external service, it might not be worth sticking with AWS pipelines.

Both GitHub, GitLab, and BitBucket offer integrated pipeline services. Compared to AWS, these services are designed to leverage a wider range of technologies and the most common third-party products. Switching to these types of pipelines could simplify all those intermediate steps that often needed to be orchestrated using complex AWS CodeBuild workflows.

At the same time, AWS has shifted its development efforts toward tools aimed at improving the developer experience and offering better integrations—for example, with GitLab. It's now possible to configure runners using AWS CodeBuild as the compute platform, allowing for more granular permissions control compared to the past when interacting with cloud resources. Who hasn't encountered EC2 runners with full AdministratorAccess permissions?

Even GitHub Actions are now integrated, and integration with BuildKite has just been announced.

## The "homemade"

"There's also the 'do-it-yourself' alternative: even a simple Git repository accessible via SSH can trigger hooks and perform actions. However, managing your own repository, and investing time to ensure high availability, security, and maintenance, doesn't really offer advantages compared to managed services that already guarantee these aspects and allow you to focus on real business needs. That said, this option shouldn't be completely ruled out for very specific requirements or unique cases.

## Conclusion

A few months ago, after Jeff Barr's announcement, many likely thought they needed to scramble to choose a new codebase in a hurry.

In reality, this isn't about discontinuing the service, but it's still a good time to start evaluating the different options the market has to offer as alternatives to a service that has gained significant success for anyone operating in the AWS ecosystem.

With its PROS and CONS, AWS CodeCommit leaves us, in its departure, an important legacy of insights about pipeline management—extremely valuable during the decision-making process for choosing a suitable alternative.

All the alternative platforms to CodeCommit provide a centralized point of management. However, they obviously don't integrate with IAM directives and potential federations with the identity tools already in use across the AWS ecosystem.

Integration with third-party IDPs is, in fact, another important aspect. Many Git repositories now natively integrate and allow SSO configuration to provide users with a seamless authentication mechanism that makes the tool easier to use. However, using SAML authentication in the case of BitBucket, GitLab, or GitHub is an option included only in paid plans with enterprise features. This is something to consider compared to the low costs of CodeCommit, which natively leveraged the federation already set up in IAM.

Once a new platform is chosen, even if everything works on paper, it's advisable to start developing new projects to get familiar with the tools offered. This helps plan a gradual migration of existing repositories, aiming for unified management. The effort required for porting existing pipelines could be much higher than simply starting with a new, simple, less critical project.

Have you already thought about which tool will host your repositories in the future? Let us know in the comments!

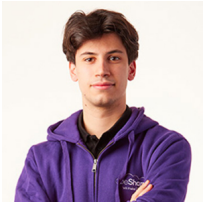
---

## About Proud2beCloud

**Proud2beCloud** is a blog by beSharp, an Italian APN Premier Consulting Partner expert in designing, implementing, and managing complex Cloud infrastructures and advanced services on AWS. Before being writers, we are Cloud Experts working daily with AWS services since 2007. We are hungry readers, innovative builders, and gem-

seekers. On Proud2beCloud, we regularly share our best AWS pro tips, configuration insights, in-depth news, tips&tricks, how-tos, and many other resources. Take part in the discussion!

---



**Riccardo Fragnelli**

DevOps @ beSharpI was born on-prem as a Dev before landing on the “Cloud side of IT”. With AWS I discovered a whole new branch of IT that fascinates me more and more; I’m always ready for the next big thing!I’m the fussiest man I know on earth and quite lazy. I like spending my free time jumping between video games and RPGs.

---