JUST AFTER MIDNIGHT

The ‘perfect’ AWS cloud architecture for SaaS applications

4 very different approaches

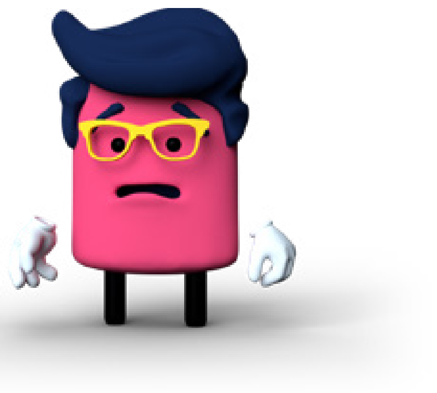
Table of contents

The situation

Myths around 'perfect' architectures and technologies can bias teams and cause serious hold-ups in deploying SaaS products in the cloud.

This can take the form of:

- **Trying to add the latest and greatest even when it doesn't make sense for the project**
- **Pushing for technologies outside a team's skill set**
- **Taking a purist approach to architecture - rather than balancing new and old together**



The solution, and why we wrote this report

Understanding SaaS and AWS (or any provider) from the perspective of your cloud engineers means making smart choices that suit you - because there is no 'perfect' architecture.

By talking with 4 of our own AWS architects, and asking them how they'd approach designing a cloud architecture for a new SaaS product, we were able to gather 4 valuable perspectives, each of which touch on some of these architecture 'myths,' as well as being valuable solutions in themselves.

Our AWS architects

Zek

Arif

Ray

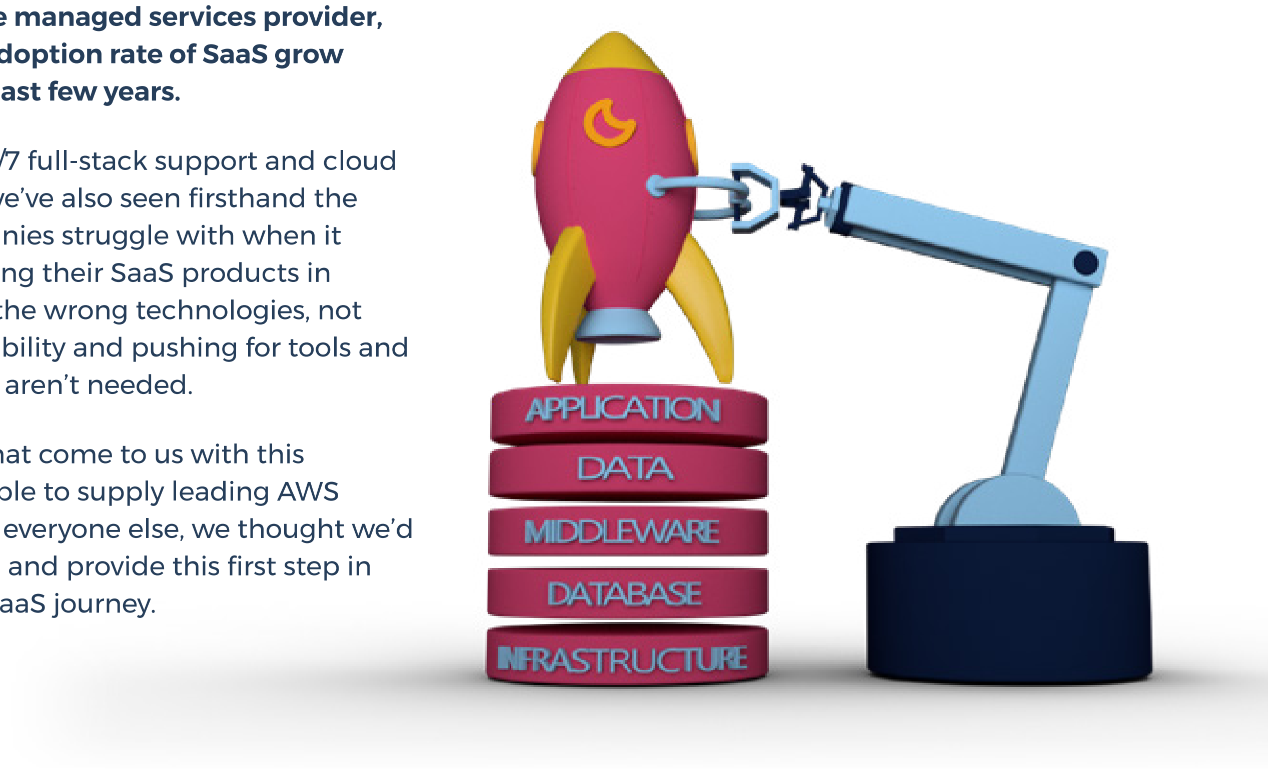
Hang

Just After Midnight & SaaS

As a cloud-native managed services provider, we've seen the adoption rate of SaaS grow rapidly over the last few years.

And as both a 24/7 full-stack support and cloud design partner, we've also seen firsthand the problems companies struggle with when it comes to deploying their SaaS products in the cloud: using the wrong technologies, not building for scalability and pushing for tools and features that just aren't needed.

For companies that come to us with this problem, we're able to supply leading AWS solutions. But for everyone else, we thought we'd share the wealth, and provide this first step in beginning your SaaS journey.



SaaS and AWS

SaaS is one of the fastest-growing delivery models, tipped to generate 140 billion in 2022 (Gartner). And with offerings like the AWS SaaS Factory Programme, and the SaaS Enablement Framework, AWS has positioned itself as the go-to cloud service provider for SaaS companies.

Why 'front-line' perspectives

Often, the reason projects fail is the disconnect between key stakeholders: the cloud/solution architects and the business decision makers.

By taking a look at 4 cloud/solution architects' perspectives on how to build the right AWS architecture for SaaS platforms/products, you learn 4 real-world approaches as well as seeing things from the architect's perspective.

#1 ZEK'S APPROACH

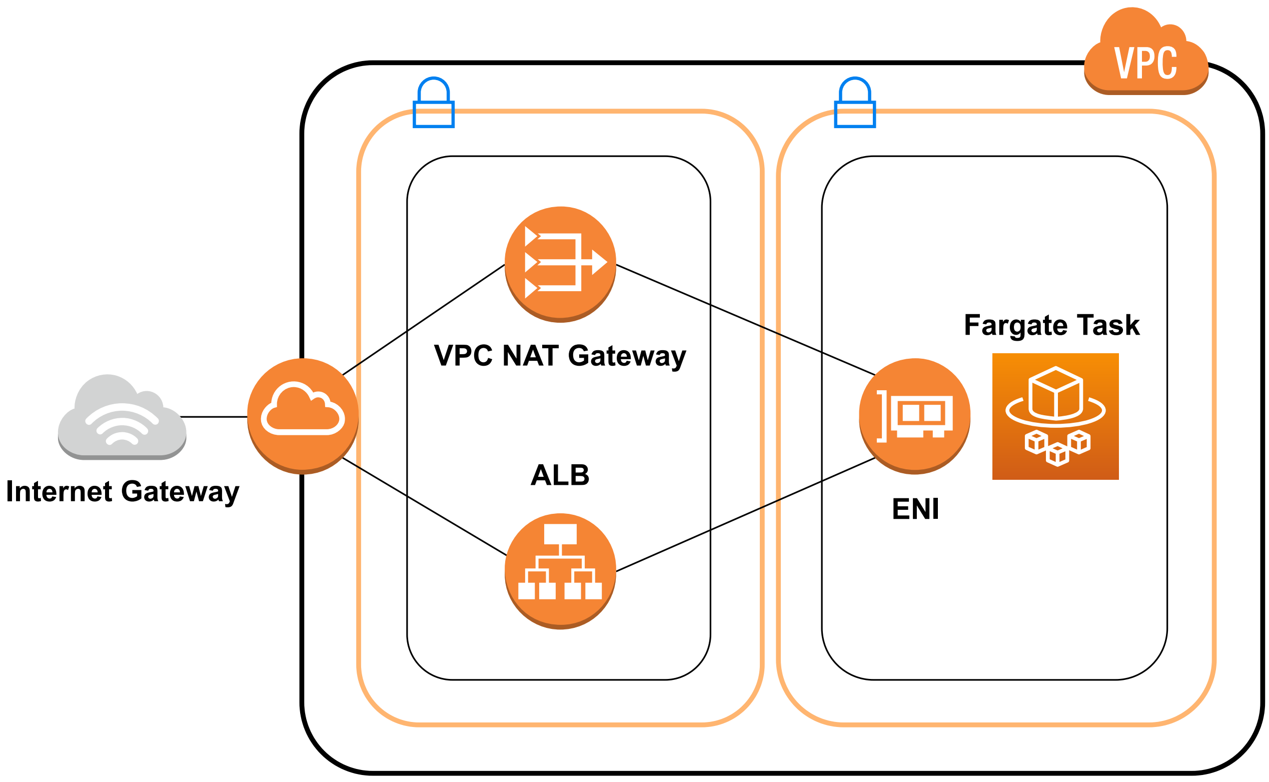
Plan for pace, not for perfection

Zek Chak
Lead Solution Architect

Zek is a lead solution architect based in our Singapore office. Before joining Just After Midnight, Zek delivered projects for the likes of Microsoft and Twitter. His approach is tactical, showing that taking on complexity bit by bit yields better results down the line.

> **It's hard to give a single answer to 'what's the perfect SaaS architecture' - but what I'd say is you should be planning for pace, not perfection.**
>
> **This means starting simple BUT giving yourself the wriggle room to incorporate more complex technologies down the line.**
>
> **Often, teams try to for an idealised version of their app - one that can do everything right from the get-go.**
>
> **And this is really running before you can walk.**

Zek's solution



Looking at the above, you can see I've gone with a monolithic approach managed with AWS Fargate.

The reasons for this are:

1. A monolithic application is often better than microservices starting out because, in the early stages, it's harder to see how your app should be split out into microservices
2. A monolithic application is easier to work with on the fly, as it can be changed and updated a lot easier than a set of microservices
3. A containerised monolith can be refactored to microservices relatively easily
4. I've gone with Fargate, as opposed to Kubernetes, because Fargate does the container management for you

So you can see, this approach really minimises the technical complexity you take on in the initial phase, and opens up room for maneuvering.

But... it leaves the door open to hyperscale technologies down the line.

This is not the only example. This is also broadly true when it comes to data: many teams will opt for powerful, NoSQL databases without stopping to think that a lot of their data is relational.

Again, starting out with a simple DB like RDS, and then moving to Redshift or similar when the need arises is often a better way.

That's my two cents - start at an even pace, and acquire more technical complexity as and when.

Zek's key takeaway

In a world where everyone thinks microservice architectures are the default for new developments, you can gain advantage starting out with a refactor-ready monolith - depending on your use case.

#2 ARIF'S APPROACH

A realist approach, that centers engineers

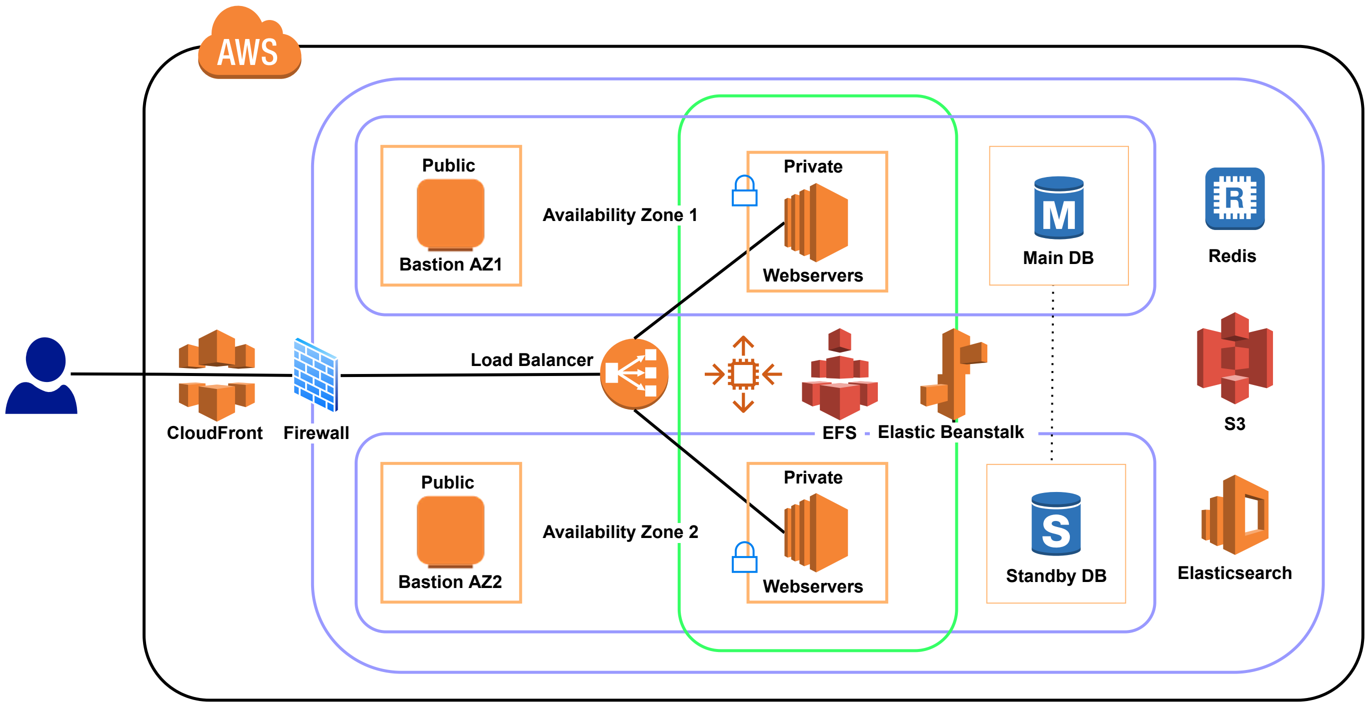
Arif Ali

Technical Director

Arif came to Just After Midnight as technical director after nearly a decade with award-winning agency Reading Room. In his long career, he's seen technology fads come and go. And his approach speaks to the constants in architecting cloud solutions.

> **My approach solves a very common problem. When people ask 'what's the best approach for a SaaS architecture on AWS,' they might expect 'Fully serverless! Lambda all the way!'**
>
> **But the de facto answer is a lot closer to, 'whatever your team's comfortable with.'**
>
> **This doesn't mean you shouldn't upskill or recruit for a new project. But the reality is that most teams will benefit from an architecture that speaks to their existing skills.**
>
> **And in a lot of cases, it's better to build what you know than to overshoot.**
>
> **With that in mind, I've come up with something that combines some nifty cloud-native practices with other, more traditional elements - those that the typical team will be able to pick up and run with.**

Arif's solution



Looking at the above, you can see I've gone with the requirements for a customer-facing SaaS web app, where availability is key.

This design will sit well with a typical team because:

1. It leans into EC2 (which engineers will have a lot of experience with)

2. It's not too modular

3. Most of the problems/needs facing SaaS companies (high availability, data and isolation, scalability) have been solved using AWS services you'll be nice and familiar with

However, it's also open to a number of cloud-native features:

1. A serverless approach will gel with the majority of tools and services here

2. This architecture is open to 3rd-party integrations, which are pretty much essential in today's world of distributed apps

There are also downsides - which is unavoidable considering this is a compromise between blue-sky and traditional thinking, but then again, that trade-off is what this approach is all about.

1. The lack of modularity to this architecture, while being easier to work with, does leave it more vulnerable to failure

2. There is more management and manual work involved, patching etc

3. Beanstalk is quite common - but it's not safe to assume everyone is going to be totally up-to-date with it

Arif's key takeaway

Your team's knowledge base can be just as important as your key requirements. If you don't have the staffing profile to get the exact stack you want, it's better to work with your developers and architects than against them.

And you can often incorporate elements of new technologies alongside more traditional services.

JUST AFTER MIDNIGHT

#3 RAY'S APPROACH

The optimal, next-gen build - with a dash of thriftiness

Ray Barlow
DevOps Engineer

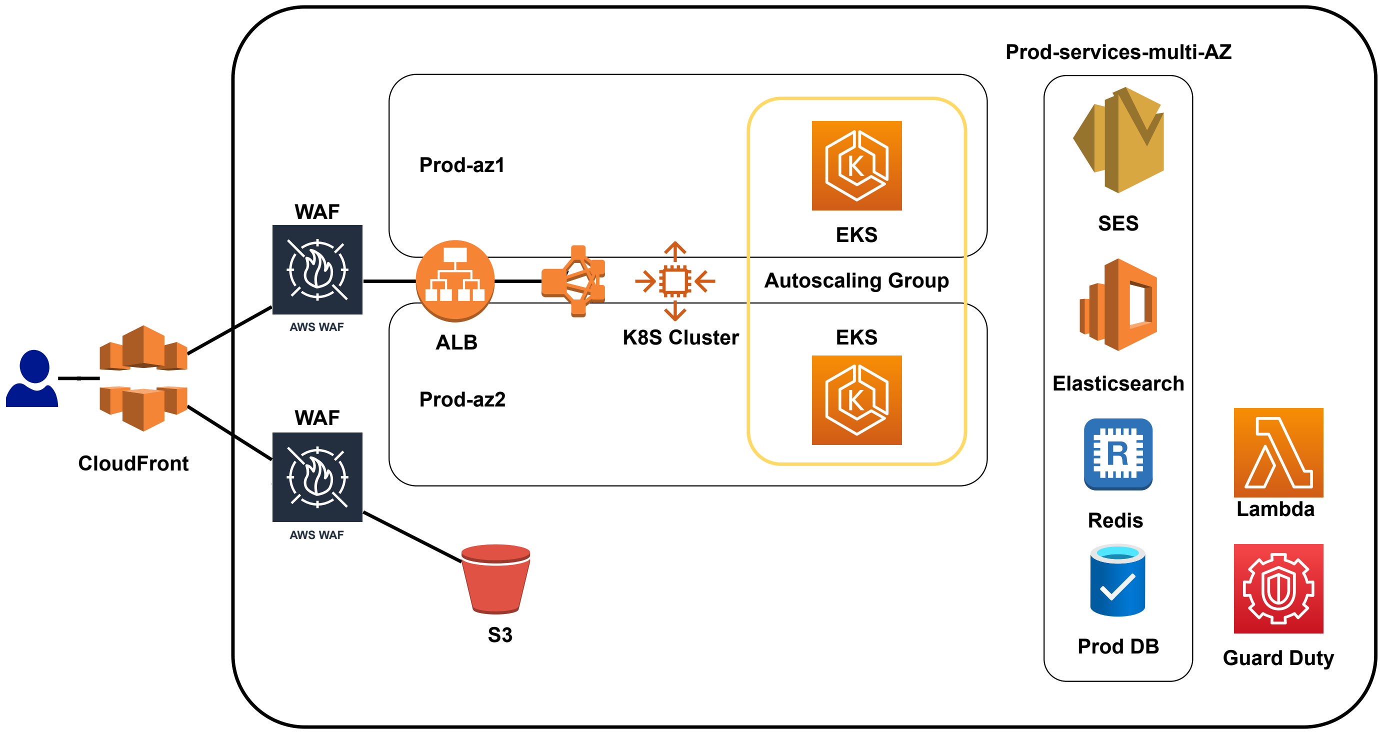
With over 30 years' experience in various senior software roles, Ray brings a wealth of knowledge to Just After Midnight. His approach finds that sweet spot between full-cloud native hyperscale and less expensive builds that many teams will want to hit.

"I've gone for a more blue-sky approach, assuming the right skills, and a middle-sized company. My approach makes use of the latest solutions; but I've reigned things in a little bit to keep costs down.

This speaks to the fact that, for many, a global hyperscale cloud isn't necessary. But you'll still be wanting to explore the flexibility, resilience and deployment speed of newer, cloud-native technologies.

Containers are a great halfway house for this, allowing you to gain a lot of benefits, without breaking the bank."

Ray’s solution



In a balance between absolute scalability and cost, I’ve erred on the side of thriftiness, because, unless you really need to hyperscale, the cost of going pure Lambda (AWS’s Serverless solution) is a little too high - when you can build a great, scalable containerised app with Kubernetes.

I’ve gone for Kubernetes over ESC because:

1. It offers more control overall (which is what you’re looking for in an orchestration tool)

2. I’m keeping this open to SaaS companies who might want to integrate multi-cloud at some point in their journey, ECS being more ‘sticky’ to the AWS ecosystem

However, as you can see, I’ve also found a place for Lambda adding functionality between CloudFront and S3.

This is really a modern, cloud-native build for a company that wants to scale but not hyperscale.

It’s a little less of a philosophy than Arif or Zek’s answer, but really there’s no right answer to this question - and sometimes bigger, newer shinier really is better!

Ray’s key takeaway

Hyperscale is a relative term. A cloud-native SaaS build can make use of containers at a lower cost than a pure Lambda solution - and unless you’re serving a global market, this will be more than enough scalability to meet demand.

#4 HANG'S APPROACH

Totally serverless, hyperscale SaaS

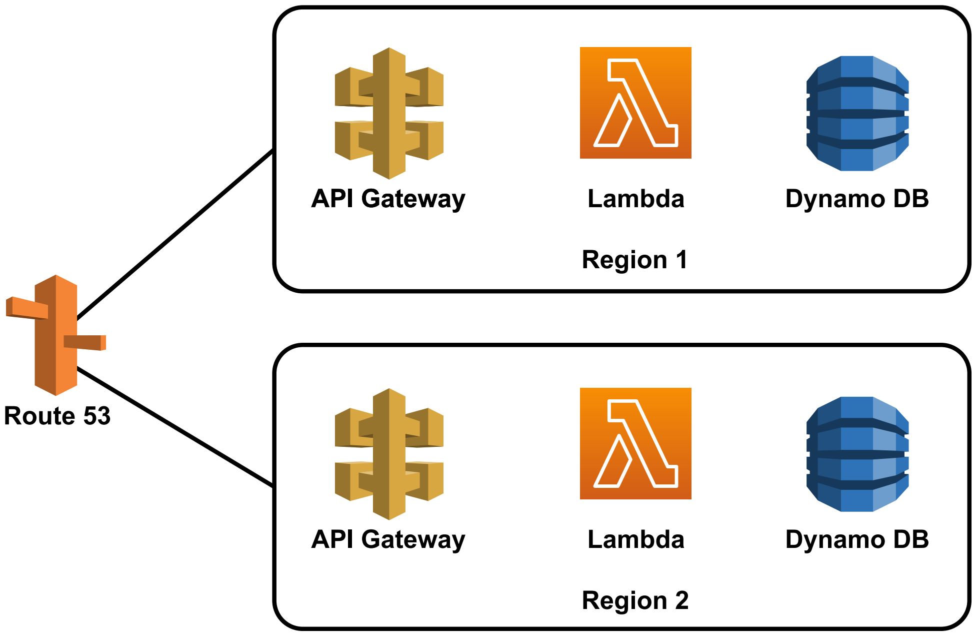
Hang Tian
DevOps Engineer

With over six years in the IT industry, Hang brings a deep knowledge of software development to her role as DevOps Engineer. Hang's approach would fit a truly global offering with hyperscale needs, and it proves that, for the right team, that 'perfect architecture' really can be the right fit.

"As Arif put it, a pure serverless approach is often the default response to the 'ideal' SaaS architecture question.

But sometimes, it's the right one. For anyone who's really trying to reach hyperscale capacity and serve a global market, the build costs of a purely serverless architecture are outweighed by the savings."

Hang's solution



My approach makes best use of all the major benefits of serveless:

1. More efficient than auto-scaling, as it only runs per request
2. Totally offloads infrastructure management
3. Keeps customers happy by delivering faster and more frequent changes as a result of breaking down your application into small functions

And the multi-region aspect allows the solution to meet the resilience and regulatory standards of a truly global product.

Obviously everything varies case by case, but the above is a pretty good starting point.

Hang's key takeaway

A globally focussed hyperscale solution should go all the way when it comes to technology. And if you plan to serve that kind of market from the beginning, you can really build around this objective.

JUST AFTER MIDNIGHT

How can we help

These 4 different perspectives offer a valuable window into how architects think about cloud, SaaS and problem solving.

But if you want the absolute right build for your SaaS product, then there's nothing like an MSP with a great track record in SaaS - that's us.

We're a next-gen managed service provider who specialises in 24/7 support, cloud and customer-facing applications.

We've delivered leading SaaS infrastructure solutions for many in Digital HR, LegalTech and security.

And we provided the full-stack 24/7 support for the launch of national fitness icon Joe Wicks' The Body Coach app, covering the cloud infrastructure, application and 3rd-party integrations.

So whether you want an architecture built from the ground up, or you're looking for a support partner, just get in touch.

Contact us

Adam Dunn

Head of Strategic Alliances

justaftermidnight247.com

/just-after-midnight

@JAM_24_7

adam.dunn@justaftermidnight.io