



TECHNOLOGY & OPERATIONAL INSIGHTS • SAAS ARCHITECTURE

What Business Leaders Should Know Before Commissioning a SaaS Platform

A practical guide to user roles, multi-tenancy, security, pricing models, data portability, implementation and long-term SaaS ownership.

READING TIME 4 min **CATEGORY** SaaS Architecture **PUBLISHED BY** Client Care Operations

PUBLISHED 20 October 2026

EXECUTIVE SUMMARY

- 01** A SaaS product is an ongoing service, not a website with a login. Define the smallest complete customer journey before the first major build phase.
- 02** Decide early how customers are separated, how permissions and audit trails work, how billing operates, and how the service will be run after launch.
- 03** Settle data ownership, portability and exit terms before signing, and build in stages with verifiable deliverables and acceptance criteria.

IN THIS ARTICLE

- 01** Define the smallest complete customer journey
- 02** Decide how customers are separated
- 03** Plan for roles, permissions, and auditability
- 04** Treat billing as an operating model
- 05** Plan for operations, not just development
- 06** Address data portability and exit before signing
- 07** Build in stages with measurable acceptance criteria
- 08** What to bring to the first development meeting

A software-as-a-service (SaaS) product is not simply a website with a login and a monthly subscription. It is an ongoing service with users, permissions, customer onboarding, operating costs, data responsibilities, software updates, customer support, and a business model that must continue to work as the customer base grows.

For a founder or organization commissioning SaaS development, several decisions deserve attention before the first major build phase.

The conversation should expose tradeoffs before code makes them expensive to reverse.

KEY INSIGHT

Define the smallest complete customer journey

Describe what a customer must do from first access to ongoing use: establish an account, invite colleagues, configure settings, complete a core workflow, see an outcome, manage subscription or service terms, obtain support, and leave with their information if the relationship ends.

A prototype that demonstrates one feature is not necessarily a deployable SaaS product. The minimum viable release should include sufficient account governance, error handling, support pathways, and operational controls for real users to trust it.

Decide how customers are separated

In multi-tenant SaaS, multiple customer organizations may share parts of an application infrastructure. One of the central architecture requirements is ensuring one tenant cannot access another tenant's data. AWS's SaaS guidance distinguishes several isolation models and emphasizes that signing a user in does not, by itself, establish tenant isolation. ^[1] ^[3] ^[4]

A development plan should state how tenant identity is attached to requests, how access is enforced server-side, how organizations are onboarded, and how cross-tenant access is tested. The right model depends on the data, customers, budget, deployment environment, and contractual or legal requirements.

Plan for roles, permissions, and auditability

A business user, manager, administrator, and support operator may need different levels of access. Define permissions around actual business responsibilities rather than screen names alone. Determine who may see, create, approve, export, edit, or delete each kind of record, and who may

change permissions.

For sensitive workflows, plan how relevant actions are recorded, reviewed, corrected, and retained. Audit logs should provide meaningful accountability while avoiding unnecessary exposure of confidential data.

Treat billing as an operating model

Recurring revenue sounds simple until customers need trials, different tiers, usage limits, invoices, failed-payment handling, cancellations, credits, or contract exceptions. Decide whether the product is billed per organization, seat, location, transaction, usage, or a hybrid; and how usage will be measured and explained.

Do not add complex billing logic before deciding which pricing structure makes sense commercially. A founder should understand how customer-support and infrastructure costs may change as adoption grows.

Plan for operations, not just development

AWS's SaaS Lens emphasizes tenant-aware monitoring, repeatable onboarding, and a unified operational experience. These are important because the development team must be able to troubleshoot incidents, deploy updates, understand service performance, and identify unusual usage without confusing one customer's environment with another's. ^[2]

Specify responsibilities for backups, restoration testing, reliability, access review, change management, incident response, and customer communications. A product owner needs a sustainable way to support these activities after launch.

Address data portability and exit before signing

Ask who owns business data, what the vendor can access, where information is stored, how it can be exported, whether audit histories are portable, and what happens when a customer leaves. If customer information is sensitive, assess applicable privacy obligations and contractual responsibilities before production use.

Also clarify intellectual-property ownership, access to source code, third-party service dependencies, licensing terms, and the maintenance arrangement. These are commercial and legal decisions requiring appropriate professional review.

Build in stages with measurable acceptance criteria

A responsible development plan usually progresses through discovery, architecture, a limited working release, structured testing, controlled pilot use, and progressive improvements. Each milestone should leave behind verifiable deliverables and defined acceptance criteria.

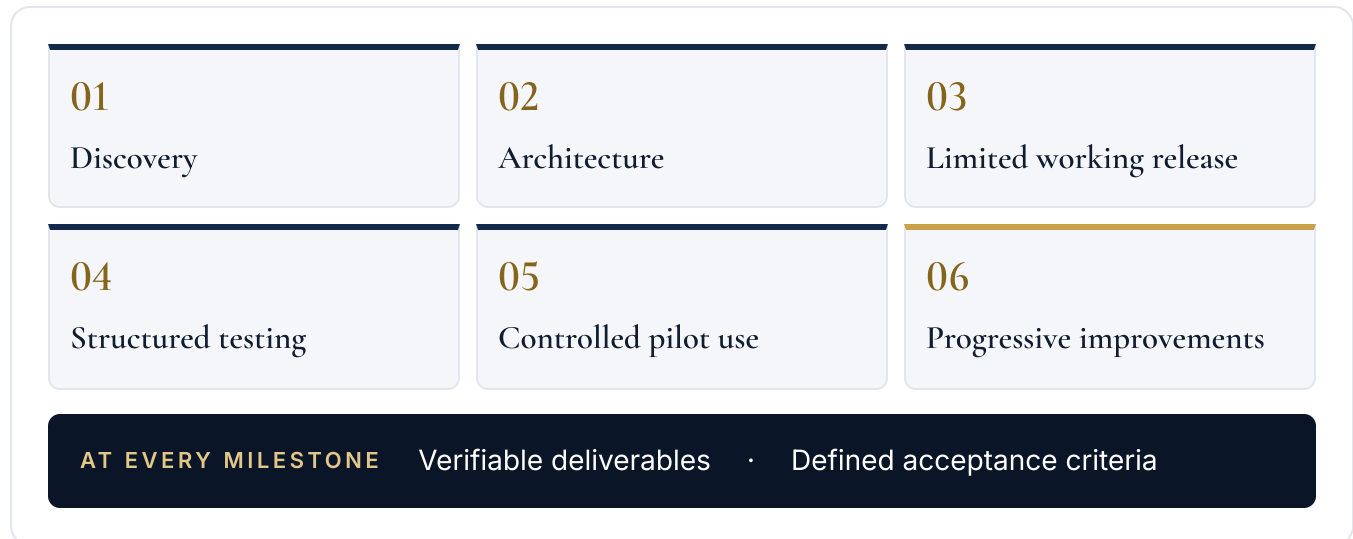


Figure 1 A staged development plan, with acceptance criteria at every milestone

Original CCO diagram based on this article.

ILLUSTRATIVE EXAMPLE

A multi-site services provider commissions a shared task platform. Instead of immediately building billing, advanced analytics, and AI, the first release establishes organizations, role permissions, task assignment, follow-up status, and auditability. More ambitious capabilities follow only once actual usage supports them.

What to bring to the first development meeting

Prepare a one-page business objective, a list of user roles, three core workflows, relevant systems and data, integration needs, constraints, expected pilot size, and a realistic budget range. The conversation should expose tradeoffs before code makes them expensive to reverse.

CCO PERSPECTIVE

A scalable software product needs a scalable operating model. Architecture, ownership, implementation, and ongoing service delivery belong in the same plan.

Client Care Operations · Technology & Operational Insights

PRACTICAL NEXT STEP

CCO SERVICE

Software Engineering & Technology Consulting

Software designed around the way your organization works.

www.clientcareoperations.com/services/software-engineering

FREE PLANNING TOOL · PDF · WORD

Software Requirements Discovery Checklist

A structured starting point for commissioning custom software, SaaS products, or systems integration.

www.clientcareoperations.com/resources#tools

Prefer to talk it through? Discuss a SaaS project before it is commissioned:

www.clientcareoperations.com/contact

Sources and further reading

- [1] **AWS, SaaS Lens: Infrastructure protection**
<https://docs.aws.amazon.com/wellarchitected/latest/saas-lens/infrastructure-protection.html>
- [2] **AWS, SaaS Lens: General design principles**
<https://docs.aws.amazon.com/wellarchitected/latest/saas-lens/general-design-principles.html>
- [3] **AWS, SaaS Lens: Core isolation concepts**
<https://docs.aws.amazon.com/wellarchitected/latest/saas-lens/core-isolation-concepts.html>
- [4] **AWS, SaaS Lens: The isolation mindset**
<https://docs.aws.amazon.com/wellarchitected/latest/saas-lens/isolation-mindset.html>

Educational guidance only. This article is not a security assurance, product warranty, or substitute for software contracting advice. References are ordinary outbound links; no endorsement by, or partnership with, the referenced organizations is implied.