DevForth

Before You Launch a Casino: A Founder's Technical Checklist

Practical lessons from casino development to protect your revenue, earn player trust, and ask your engineering team the right questions.

November 2026

Contents

Protect your margins when offering bonuses . . . 3
- Keep promotional credit separate from withdrawable funds . . . 3
- Check the economics behind every offer . . . 3
- Count wagering when a bet settles . . . 3
- Close the free-spin and reward-farming loophole . . . 4

Secure the back office where money moves . . . 4
- Require a second factor for critical financial actions . . . 4
- Keep an audit trail your team can rely on . . . 4
- Give each role only the access it needs . . . 4

See where players get stuck . . . 5

Make the path to a first deposit simple . . . 5

Back up the business, and practise restoring it . . . 5

Plan for incidents outside office hours . . . 5

Prevent concurrent requests from spending the same balance . . . 6

Keep secure bet handling fast . . . 6

Talk through your launch with our team . . . 6

You know how to bring players to a casino. The next challenge is building a platform that earns their trust, protects your margins, and keeps working when real money is on the line.

Whether you are moving from affiliate marketing into operating your own casino or preparing your next launch, the details below deserve attention before you spend on player acquisition.

This checklist draws on our experience building and supporting gambling platforms at DevForth. Use it to review your launch plan with your engineering team: how bonuses work, who can move money, what happens when something fails, and how the platform behaves under load.

Protect your margins when offering bonuses

Keep promotional credit separate from withdrawable funds

A welcome bonus can attract players. It can also attract automated abuse. A small reward, claimed repeatedly across large numbers of accounts, can become an expensive leak before your team identifies the pattern.

For deposit-linked offers, tie bonus credit to a real deposit and clearly defined wagering requirements. Your platform needs to distinguish promotional credit from withdrawable funds and apply the published bonus rules consistently.

Check the economics behind every offer

Consider a $100 bonus played on a game with a 95% return to player (RTP). The expected loss per dollar wagered is 5%, so $2,000 in total wagers corresponds to an expected loss of $100:

$100 ÷ (1 – 0.95) = $2,000 in wagers.

That is an average-value illustration, not a guarantee that a player will lose the bonus or a complete formula for setting withdrawal conditions. Individual outcomes vary, and higher-RTP games change the economics. Review wagering requirements and game contributions against the actual content in your casino.

Count wagering when a bet settles

Crediting wagering progress when a bet is placed creates a loophole. A bet may later be cancelled or remain unresolved for days, particularly in sports betting. A player could meet the apparent wagering requirement without completing the underlying bets.

Count eligible wagering on settlement, once the outcome is known. Cancelled bets should not advance the requirement.

We have seen unreliable provider settlement events offered as a reason to count bets early. Treat

that as an integration problem to resolve before launch: bonus accounting depends on a reliable bet lifecycle.

Close the free-spin and reward-farming loophole

Some third-party slots let players accumulate free spins or other rewards through play. A player may build up those rewards with bonus funds, exhaust the bonus, then return to real-money play and collect the stored rewards without the original wagering restrictions.

Review how each game's rewards interact with your bonus rules:

- Exclude games with these mechanics from bonus play where appropriate. This is the simplest control.
- If you allow them, build detection for reward farming and define how your team will respond.
- Make the rules clear before play. Account blocks or disputed winnings can damage player trust.

Secure the back office where money moves

Require a second factor for critical financial actions

In 2020, one of our clients lost more than $100,000 after malware compromised a VIP manager's computer. The manager had permission to issue large bonuses, and that access was used to extract funds.

After that incident, we introduced two-factor authentication (2FA) for money-related actions and prevented at least five similar incidents. The lesson: protect the financial action itself, not just the initial login.

We build client back offices with AdminForth, our free, open-source framework. It supports authenticator-app codes and passkeys, with security controls built into the workflows staff use every day.

Keep an audit trail your team can rely on

When money goes missing, you need clear answers: who issued a VIP bonus, approved a withdrawal, or changed campaign settings, and when?

Keep immutable audit logs for critical actions so your team can reconstruct events without relying on memory or records that can be edited after the fact.

Give each role only the access it needs

Role-based access control (RBAC) defines what each member of staff can do. Match those permissions to their responsibilities, especially when they can affect balances, bonuses, or withdrawals.

A convenient back office should not require broad financial access for routine tasks.

See where players get stuck

Acquisition spend is wasted when a player reaches your site but cannot finish signing up. The cause may be as small as an unclear button or an error that appears only on a particular device or in one country.

Use session recordings, such as Hotjar, to review the first steps players take on your platform. Have QA and developers watch the journeys, investigate failed attempts, and fix the points of friction. A recording can reveal a problem that a conversion-rate chart alone cannot explain.

Make the path to a first deposit simple

Aim for a short, clear journey from signup to deposit. Three clicks is a useful design target where your required checks allow it. Every extra field, popup, or unexplained step gives a prospective player another reason to leave, including someone who could become a valuable VIP.

Offer familiar third-party sign-in options through OAuth, keep the signup form focused, and defer optional details such as a phone number until they are needed.

Build required know-your-customer (KYC) checks into the journey at the points your licensing and fraud controls require. Where verification is required before a deposit, make that step clear and easy to complete.

Back up the business, and practise restoring it

Player balances, transaction history, account access, and VIP status are core business records. Losing them can put the entire operation at risk.

Set up backups from day one and test the recovery procedure. Ask your team to demonstrate a restore: a successful backup job is only part of the work.

Plan for incidents outside office hours

A casino runs around the clock. Monitoring needs to detect an outage quickly and alert someone who can act, including at night and on weekends. Agree on who owns the response before launch.

At DevForth, planned development happens on weekdays; production incident response extends beyond those hours. If a platform goes down on a weekend, monitoring alerts the responsible developer by phone. The response includes restoring service and addressing the cause to reduce the chance of another outage.

Prevent concurrent requests from spending the same balance

One balance check can look correct in isolation and still fail when several requests arrive together. This is a time-of-check-to-time-of-use (TOCTOU) race: the balance changes between being checked and being debited.

Imagine an account with $100 sending ten simultaneous withdrawal requests for $100 each. Without proper concurrency controls, several requests could see the same available balance and pass the check. If eight succeed, the platform could pay out $800 against a $100 balance and record a balance of −$700.

A negative balance is a warning that warrants investigation; this race condition is one possible cause.

One approach is to serialize all balance-decreasing operations in a queue per player. Bets and withdrawals are processed one at a time, with each operation checking the updated balance before proceeding. This coordination must cover every service that can debit that player's funds.

Ask your team to demonstrate what happens when bets and withdrawals arrive simultaneously. This is a launch-critical money flow.

Keep secure bet handling fast

Serializing balance operations protects funds, but slow processing can make gameplay feel unresponsive. If checking a balance and recording a bet takes 300–400 ms, rapid-play games such as Plinko can expose the delay.

Database design and the way balance operations are stored have a direct effect on performance. Ask your developers to measure this path under realistic load. Use 100 ms as a practical review threshold for the balance-check-and-bet operation: if it takes longer, investigate the bottleneck.

The goal is responsive play with the same financial correctness at peak traffic as during a quiet test.

Talk through your launch with our team

DevForth is a gambling-focused, end-to-end software development team. We build casino platforms and original games, covering architecture, aggregator and payment integrations, back-office tools, and production support.

Tell us about your project: what you want to launch, what you already have, and where you need engineering support.