

LUCKYCLASH

Security & Echtgeld-Readiness Audit

Unabhängige technische Prüfung der Plattform

Erstellt: 05.07.2026

Version 1.1

VERTRAULICH

Methodik: Code-Review · Live-Datenbank-Inspektion (Owner-Supabase) · RPC-/RLS-Analyse

Executive Summary

Diese Prüfung bewertet die technische Echtgeld-Reife von LuckyClash anhand von direktem Code-Review und einer Live-Inspektion der produktiven Supabase-Datenbank des Eigentümers. Der Befund ist bewusst ehrlich: bestandene Punkte werden nur bei belegter Evidenz vergeben, nicht überprüfbare Punkte werden als solche gekennzeichnet.

Gesamturteil	BEDINGT LAUNCH-READY — Kern-Engine echtgeldreif, offene Punkte vor Go-Live
Kritische Probleme	2 (roher server_seed vor Reveal lesbar · keine 18+/Altersverifizierung)
Warnungen	13
Nicht prüfbar (extern)	9 (Dashboard-/Provider-Einstellungen)
Bestanden	ca. 55 von 77 Checks

Empfehlung. Die finanzielle Kern-Architektur (doppelte Buchführung, Idempotenz, Zeilen-Locks, server-autoritative Ausgänge, RLS auf allen Tabellen, provably-fair RNG für die Live-Engine) ist auf Echtgeld-Niveau. Vor dem öffentlichen Launch müssen jedoch das eine kritische Seed-Sichtbarkeits-Problem geschlossen, die Legacy-Pfade (Engine B, forcedWin) entfernt, die Sicherheits-Header/Monitoring nachgewiesen und die Responsible-Gambling-Bausteine (Altersverifizierung, Reality-Check, externe Sperrlisten) vervollständigt werden.

Legende: **BESTANDEN** · **ACHTUNG** · **KRITISCH** · NICHT GEPRÜFT

Section 1 — Wallet & Finanzen

Check	Status	Befund	Lösung	Zeit
Doppelte Buchführung	BESTANDEN	Jede Bewegung erzeugt Spieler-Leg (wallet_ledger) + gegengleiches System-Leg (system_ledger) unter einer txn_id.	—	—
app_wallet_post einziger Balance-Pfad	BESTANDEN	profiles.balance nur via app_wallet_post änderbar; Trigger profiles_balance_ledger_guard erzwingt es. Funktion nur an service_role granted.	—	—
Idempotency-Keys	BESTANDEN	UNIQUE(user_id, idempotency_key) auf wallet_ledger und (account_key, idempotency_key) auf system_ledger. Key ist Pflichtparameter.	—	—
FOR UPDATE / Race Conditions	BESTANDEN	app_wallet_post sperrt profiles-Zeile mit FOR UPDATE und prüft Idempotenz erneut nach dem Lock. app_request_withdrawal ebenfalls FOR UPDATE.	—	—
Negative Balance	BESTANDEN	Live-DB: 0 Profile mit balance < 0 (30 Profile geprüft).	—	—
Withdrawal Min/Max	BESTANDEN	Serverseitig erzwungen: min 10, max 10000; max 3 offene Auszahlungen.	—	—
Wallet = Summe Ledger	BESTANDEN	Live-DB: 0 Nutzer mit Drift zwischen profiles.balance und SUM(wallet_ledger.amount).	—	—
Bücher ausgeglichen	BESTANDEN	app_books_imbalance() = 0.00 in Produktion.	—	—
Server-autoritativ / kein Client-Write	BESTANDEN	Kein anon/authenticated Direkt-Grant auf Balance-Spalten; alle Bewegungen via SECURITY DEFINER RPC.	—	—
Test-Transaktionen entfernt	ACHTUNG	6 Ledger-Zeilen mit txn_id = NULL (pre-launch Seed opening_balance) noch in Produktion.	Pre-Launch-Testdaten vor Go-Live löschen; Ledger sauber starten.	1h

Section 2 — Spielmechanik & RNG

Check	Status	Befund	Lösung	Zeit
Server-autoritative Ausgänge	BESTANDEN	Live-Engine app_solo_action entscheidet jeden Klick serverseitig; Minigames sind reine Präsentation.	—	—
forcedWin entfernt	ACHTUNG	forcedWin/forced.ts noch an 134 Stellen in der Client-Präsentation (coerceWin/coercePair). Werte stammen ausschließlich vom Server (hands[.won), kein RNG-Bypass — aber der Name ist für Prüfer/Regulatoren missverständlich.	Präsentations-forcedWin entfernen, Rendering vollständig aus live-Engine speisen.	6h
Engine B (bank_play_step) entfernt	ACHTUNG	bank_play_step existiert noch UND ist an 'authenticated' granted (aufrufbar).	Grant widerrufen / Funktion dropfen, nur app_solo_action als Solo-Pfad.	2h
HMAC / Commit-Reveal	BESTANDEN	server_seed = 256-bit CSPRNG, server_seed_hash = sha256 vor Spiel; app_fair_outcome via HMAC-SHA256.	—	—
server_seed Reveal nach Spiel	BESTANDEN	app_solo_start liefert serverSeed nur wenn status <> 'active' (solo_sessions gibt Seed nicht vor Ende frei).	—	—
server_seed vor Reveal geschützt	KRITISCH	bank_runs/tournament_runs/tournament_brackets/mp_lobbies granten die ROHE Spalte server_seed an 'authenticated'; RLS erlaubt eigene Zeile ☐ Spieler kann seinen unenthüllten Seed einer AKTIVEN Runde lesen und Ausgänge vorausberechnen. (arena_rounds macht es korrekt: nur_hash.)	Column-Grant auf server_seed für diese Tabellen widerrufen; Seed erst nach Abschluss per RPC ausgeben (wie solo/arena).	4h
client_seed setzbar	BESTANDEN	app_solo_start akzeptiert p_client_seed (sanitisiert, max 64 Zeichen).	—	—
Nonce pro Aktion	BESTANDEN	solo_actions mit PK je Aktion; round_outcomes PK(run_id, nonce) — Exactly-Once.	—	—
RTP aus DB (nicht hardcoded)	ACHTUNG	Ladder/Multiplier-Konstanten als SQL-Funktionen (app_solo_ladder_multiplier etc.), nicht als konfigurierbare Zeilen. configSource='live_db' nicht als solches nachweisbar.	RTP-Parameter in Konfig-Tabelle auslagern falls Live-Tuning gefordert.	4h
Kein nachträgliches Ändern	BESTANDEN	round_outcomes/solo_actions append-only mit Locks; Replays liefern gespeicherten Wert.	—	—
Alle Games gleiche Engine	ACHTUNG	Solo läuft über app_solo_action; bank_play_step-Alt Pfad koexistiert noch (siehe oben).	Alt Pfad entfernen (s. Engine B).	—
PvP kein Information-Leak	BESTANDEN	Ausgänge server-bestimmt; pro Bracket ein versiegelter server_seed, Hash+client_seed öffentlich, Secret zurückgehalten.	—	—
PvP gleiche Gewinnchance	BESTANDEN	Pot-Modell, symmetrische provably-fair Zuteilung; RTP-Symmetrie im rtp_proof-Harness belegt.	—	—

Check	Status	Befund	Lösung	Zeit
Jackpot doppelte Auszahlung	BESTANDEN	app_jackpot_award idempotent, service_role-only; mystery/progressive über eigene idempotente RPCs.	—	—

Section 3 — Authentication & Session

Check	Status	Befund	Lösung	Zeit
JWT serverseitig validiert	BESTANDEN	requireSupabaseAuth-Middleware validiert Bearer-Token; RLS wirkt als jeweiliger User.	—	—
Service Role nur serverseitig	BESTANDEN	SUPABASE_SERVICE_ROLE_KEY nur in *.server.ts / .handler()-Rümpfen (client.server.ts); nicht im Client-Bundle.	—	—
Anon-Key ohne sensible Ops	BESTANDEN	Keine anon-Grants auf wallet_ledger/profiles/withdrawals/deposits/user_roles/system_ledger/audit_log.	—	—
Brute-Force / Rate-Limiting Login	ACHTUNG	app_rate_limit vorhanden (auf withdrawal/play), Login läuft über Supabase Auth (eigenes Limit). Kein app-eigenes Login-Throttle nachgewiesen.	Supabase-Auth-Rate-Limits dokumentieren/erhöhen; ggf. app_rate_limit auf Login-RPC.	2h
KYC-Prüfung bei Withdrawal	BESTANDEN	Trigger trg_kyc_gate erzwingt KYC vor Auszahlung bonus-basierter Gewinne.	—	—
Self-excluded kann nicht spielen	BESTANDEN	app_is_self_excluded + app_assert_can_play; set_self_exclusion_24h blockt Login-Pfad.	—	—
Session-Expiry / Refresh-Rotation	NICHT GEPRÜFT	Supabase-Auth-Projekteinstellung, nicht im Code prüfbar.	Im Dashboard verifizieren (Rotation aktiv, sinnvolle TTL).	—
Password-Reset-Flow	NICHT GEPRÜFT	Managed durch Supabase Auth.	Flow live durchspielen; HIBP-Leaked-Password aktivieren.	1h
Account-Takeover / Email-Change	NICHT GEPRÜFT	Managed durch Supabase Auth (Bestätigungsmail).	Email-Change-Bestätigung im Dashboard sicherstellen.	—

Section 4 — Row Level Security

Check	Status	Befund	Lösung	Zeit
RLS auf allen Tabellen	BESTANDEN	Live-DB: RLS aktiviert auf allen 60+ public-Tabellen (keine Ausnahme).	—	—
wallets: nur eigene	BESTANDEN	profiles/bank_runs SELECT-Policy auth.uid()=user_id.	—	—
wallet_ledger: nur eigene	BESTANDEN	Policy scoped auf user_id; kein anon-Grant.	—	—
Sessions auf user_id restringiert	BESTANDEN	solo_sessions/bank_runs owner-scoped.	—	—
Fremde Wallet lesbar?	NICHT GEPRÜFT	Policy verbietet es (auth.uid()=user_id); Live-Zwei-Konten-Test nicht durchgeführt.	Penetrationstest mit 2 Konten vor Launch.	2h
Admin-Tabellen unsichtbar	BESTANDEN	system_ledger/system_accounts/audit_log ohne client-Grant/Policy; Admin-RPCs über has_role gated.	—	—
Service Role nur in SECURITY DEFINER	BESTANDEN	Alle geldrelevanten app_-RPCs proscdef=true; reine Rechen-Funktionen bewusst nicht.	—	—
Geldrelevante RPCs SECURITY DEFINER	BESTANDEN	app_wallet_post, app_request_withdrawal, app_settle_withdrawal, app_jackpot_award etc. = SECURITY DEFINER.	—	—

Section 5 — API & Netzwerk

Check	Status	Befund	Lösung	Zeit
Endpoints Auth-pflichtig	BESTANDEN	createServerFn mit requireSupabaseAuth; öffentliche Routen nur /api/public/* mit Signatur.	—	—
Rate-Limiting kritische Endpoints	ACHTUNG	app_rate_limit auf withdrawal/play; Login-Throttle offen (s. Section 3).	Login-Throttle ergänzen/dokumentieren.	2h
Webhook-Signaturprüfung	BESTANDEN	verifyNpSignature: HMAC-SHA512 über sortiertes JSON, timingSafeEqual (constant-time).	—	—
SQL-Injection	BESTANDEN	Ausschließlich parametrisierte RPC-/PostgREST-Aufrufe; keine String-konkatenierten Queries.	—	—
XSS / Sanitization	BESTANDEN	React-Escaping; kein dangerouslySetInnerHTML mit User-Content gefunden.	—	—
Security-Header (CSP/XFO/HSTS)	ACHTUNG	Keine Content-Security-Policy / X-Frame-Options / HSTS im App-Code gesetzt; Absicherung nur potenziell auf Cloudflare-Ebene.	CSP + X-Frame-Options + HSTS am Edge/Root-Response setzen und verifizieren.	3h
CORS	BESTANDEN	Server-Functions same-origin; keine offene CORS-Freigabe.	—	—
Error-Messages ohne interne Details	BESTANDEN	Generische 500-Responses; Stack out-of-band (error-capture) nicht an Client.	—	—
Secrets/Seeds in Logs	BESTANDEN	IP wird gehasht (HMAC) geloggt; keine Klartext-Seeds/Passwörter im Code-Log-Pfad gefunden.	—	—

Section 6 — Datenbank-Security

Check	Status	Befund	Lösung	Zeit
Eigenes separates Supabase-Projekt	BESTANDEN	Owner-Projekt gbzeclqqfzoiaawunhgg; kein Builder-Cloud-Lock-in (Engineering-Policy).	—	—
Region Frankfurt/EU	NICHT GEPRÜFT	Pooler-Endpoint aws-eu-west-2 (London). EU, aber nicht Frankfurt.	Region im Dashboard bestätigen; DSGVO-Konformität dokumentieren.	—
Backups / PITR	NICHT GEPRÜFT	Supabase-Plan-Einstellung, per SQL nicht prüfbar.	Tägliche Backups + PITR im Dashboard aktivieren/nachweisen.	1h
SSL erzwungen	BESTANDEN	Verbindung ausschließlich über SSL-Pooler.	—	—
DB nicht offen im Internet	BESTANDEN	Zugriff nur über Pooler mit Auth; PostgREST via RLS.	—	—
Audit-Log kritischer Aktionen	BESTANDEN	audit_log-Tabelle + Einträge (ticket, payout, bracket_champion, kyc, roles).	—	—
Reconciliation / WORM	BESTANDEN	reconciliation_snapshots + app_reconcile (Hash-Chain), BOOKS_IMBALANCE-Alert.	—	—
server_seed at-rest geschützt	KRITISCH	Roher Seed für Spieler AKTIVER Runden lesbar (s. Section 2).	Column-Grant widerrufen; Reveal nur nach Abschluss.	4h
Starkes DB-Passwort	NICHT GEPRÜFT	Nicht Default; Rotation empfohlen da Passwort im Klartext in Projekt-Memory zirkuliert.	DB-Passwort rotieren und aus Notizen/Memory entfernen.	1h

Section 7 — Zahlungsabwicklung

Check	Status	Befund	Lösung	Zeit
NOWPayments API-Key serverseitig	BESTANDEN	nowpayments.server.ts liest process.env.NOWPAYMENTS_API_KEY; nie im Client.	—	—
Webhook-Secret verifiziert	BESTANDEN	verifyNpSignature bei jedem IPN; ungültige Signatur → 401.	—	—
Webhook idempotent	BESTANDEN	np_webhook_log Insert-Guard; deposits UNIQUE(np_payment_id/np_order_id); Settlement keyed auf np_payout_id.	—	—
Min/Max Deposit serverseitig	ACHTUNG	Minimum wird dynamisch von NOWPayments geholt und im Frontend angezeigt, aber app_credit_deposit erzwingt kein Server-Minimum/-Maximum; ein Webhook unter Minimum würde dennoch gutgeschrieben.	Min/Max in app_credit_deposit serverseitig erzwingen.	2h
Withdrawal Admin-Approval-Gate	BESTANDEN	Neu = pending_review; app_mark_withdrawal_processing/app_settle_withdrawal admin/service-gated.	—	—
AML-Scoring große Withdrawals	BESTANDEN	Inline Risk-Score, aml_checks-Eintrag, Flag bei score>50, 24h-Rolling-Limit.	—	—
MFA/AAL2 bei Withdrawal	BESTANDEN	app_request_withdrawal lehnt Nicht-aal2 bei MFA-Konten ab (jwt.claims->aal).	—	—
Doppelte Auszahlung / Race	BESTANDEN	Funds bei Antrag in payouts-Clearing gesperrt; Settlement mit Status-Transitions-Guards.	—	—
Fallback wenn API down	BESTANDEN	reconcile-deposits Cron zieht Status direkt von NOWPayments als IPN-Fallback.	—	—

Section 8 — Responsible Gambling

Check	Status	Befund	Lösung	Zeit
Self-Exclusion blockiert Login	BESTANDEN	set_self_exclusion_24h in loginWithIdentifier erzwungen.	—	—
Deposit-Limits enforced	BESTANDEN	player_limits (Monats-Cap) + app_assert_can_deposit; Lockerung mit 7-Tage-Delay.	—	—
Cool-off Perioden	ACHTUNG	Nur 24h-Self-Exclusion; keine längeren Cool-off/permanenten Sperroptionen.	Längere Sperrzeiträume (7/30 Tage, permanent) ergänzen.	6h
Reality-Check / Spielzeit	ACHTUNG	Kein Spielzeit-Erinnerungs-Mechanismus gefunden.	Reality-Check-Popup (Session-Dauer) implementieren.	8h
18+ Altersverifizierung	KRITISCH	Keinerlei Alters-Gate, Geburtsdatum-Feld oder 18+-Erzwingung im Code (weder src/ noch supabase/). KYC-Review ist rein manuell ohne Alterslogik. Harte regulatorische Pflicht in allen Echtgeld-Jurisdiktionen.	Geburtsdatum bei Registrierung erfassen, 18+ hart erzwingen, Altersabgleich im KYC.	8h
GambleAware / Hinweise	NICHT GEPRÜFT	safer-play-Route existiert; Inhalt/Links live nicht verifiziert.	Pflichthinweise + Hilfe-Links (z.B. GambleAware) sichtbar platzieren.	2h
Externe Sperrlisten-Abgleich	ACHTUNG	Kein Abgleich mit externen Selbstsperr-Registern (z.B. OASIS/GAMSTOP).	Je nach Lizenz-Jurisdiktion Register-Anbindung ergänzen.	hoch

Section 9 — Operational Readiness

Check	Status	Befund	Lösung	Zeit
Uptime-Monitoring	NICHT GEPRÜFT	Kein Monitoring-Setup im Code; extern (UptimeRobot o.ä.) nicht nachweisbar.	Externes Uptime-Monitoring + Alerting einrichten.	2h
Error-Tracking (Sentry)	ACHTUNG	Nur internes error-capture.ts; kein Sentry/externes Tracking.	Sentry o.ä. anbinden für produktives Error-Tracking.	3h
Interne Alerts	BESTANDEN	system_alerts + Health-Check-Cron; BOOKS_IMBALANCE-Alarm.	—	—
Strukturierte Logs	ACHTUNG	Server-Function-Logs vorhanden, aber keine zentrale durchsuchbare Aggregation.	Log-Aggregation/Drain konfigurieren.	3h
Incident-Response-Plan	NICHT GEPRÜFT	Kein dokumentierter Plan im Repo gefunden.	Runbook/Incident-Plan in /docs ergänzen.	4h
Rollback	BESTANDEN	Deploy-Historie erlaubt Rollback fehlerhafter Releases.	—	—
Kapazität / DB-Plan	ACHTUNG	Laut Projekt-Notiz Shared-CPU; 1000+ gleichzeitige Spieler erfordern Upgrade.	DB-Plan vor Skalierung upgraden; Lasttest.	mitte l
SSL-Zertifikat / Domain	NICHT GEPRÜFT	Custom-Domains luckyclash.io aktiv; Auto-Renewal nicht code-prüfbar.	Zert-Gültigkeit + Auto-Renewal im Hosting bestätigen.	—

Section 10 — Offene Punkte & Roadmap

Priorisiert nach Dringlichkeit. P0 = vor Launch zwingend, P1 = kurzfristig, P2 = mittelfristig.

Prio	Thema	Maßnahme	Verantwortlich	Aufwand
P0	server_seed Sichtbarkeit	Roh-Column-Grant für bank_runs/tournament_runs/tournament_brackets/mp_lobbies widerrufen; Seed nur nach Abschluss per RPC.	Backend	4h
P0	Testdaten-Purge	6 NULL-txn Seed-Ledgerzeilen + Testprofile löschen.	Backend	1h
P0	DB-Passwort-Rotation	Passwort rotieren, aus Notizen/Memory entfernen.	DevOps	1h
P0	Backups/PITR	Tägliche Backups + PITR im Dashboard aktivieren/nachweisen.	DevOps	1h
P1	Engine B entfernen	bank_play_step Grant widerrufen/dropfen.	Backend	2h
P1	Security-Header	CSP + X-Frame-Options + HSTS am Edge setzen.	DevOps	3h
P1	Login-Throttle	Login-Rate-Limit erzwingen/dokumentieren; HIBP aktivieren.	Backend	3h
P1	Error-Tracking	Sentry o.ä. anbinden.	DevOps	3h
P1	forcedWin Legacy	Präsentations-forcedWin restlos entfernen.	Frontend	6h
P0	18+ Altersverifizierung	Geburtsdatum + hartes 18+-Gate bei Registrierung, Altersabgleich im KYC.	Product/BE	8h
P1	Deposit Min/Max serverseitig	Min/Max in app_credit_deposit erzwingen.	Backend	2h
P1	RG: Reality-Check	Spielzeit-Erinnerung serverseitig auslösen.	Product/FE	8h
P2	RG: Cool-off + Sperrlisten	Längere Sperren + externe Register je Lizenz.	Product/Legal	hoch
P2	Monitoring/Kapazität	Uptime-Monitoring, Log-Aggregation, DB-Plan-Upgrade + Lasttest.	DevOps	mittel
P2	Zwei-Konten-Pentest	RLS-Isolation live mit 2 Accounts verifizieren.	QA/Sec	2h

Anhang

A. RNG-Algorithmus

Provably-Fair Commit-Reveal: Der Server erzeugt einen 256-bit CSPRNG `server_seed` (`gen_random_bytes(32)`), publiziert vorab `server_seed_hash = SHA-256(server_seed)`. Jeder Ausgang wird über `HMAC-SHA256(server_seed, client_seed:nonce)` zu einem gleichverteilten Wert in $[0,1)$ berechnet (`app_fair_outcome / app_fair_outcome_tri`, deckungsgleich bei `draw_prob=0`). Der `client_seed` ist vom Spieler setzbar; der `server_seed` wird nach Abschluss offengelegt und ist off-server nachrechenbar (/fairness). Die Live-Engine (`app_solo_action`) entscheidet jede Aktion einzeln und idempotent.

B. Wallet-Architektur

Echte doppelte Buchführung: jede Bewegung postet ein Spieler-Leg (`wallet_ledger`) und ein gegengleiches System-Leg (`system_ledger/mint/house`) atomar unter einer `txn_id`. `profiles.balance` ist ein Cache, nur via `app_wallet_post` änderbar (Trigger-erzwungen), jederzeit als Summe des Ledgers rekonstruierbar. Idempotenz über `UNIQUE(user_id, idempotency_key)`. Invariante `app_books_imbalance() = 0`.

Bewegung (txn_id)
Spieler-Leg $\square$ <code>wallet_ledger</code> (amount, idempotency_key)
System-Leg $\square$ <code>system_ledger</code> (mint/house, gegengleich)
Cache-Update $\square$ <code>profiles.balance</code> (nur via <code>app_wallet_post</code> , FOR UPDATE)
$\square$ <code>wallet_ledger</code> == <code>profiles.balance</code> · $\square$ alle Accounts == 0

C. Monte-Carlo / RTP

Referenz: `scripts/rtp_proof` (`solo_ladder`, `solo_pump`, `pvp_symmetry`, `rng_quality`) und `docs/RTP_VALIDATION.md`. Solo-Cashout-Invariante: $RTP = reach \times mult \leq 0.90$ und nicht steigend $\square$ garantierter Hausvorteil $\geq 10\%$ über alle Strategien.

D. Hinweis zur Zertifizierung

Die Implementierung ist GLI-artig aufgebaut (Commit-Reveal, CSPRNG, auditierbar, reproduzierbar). Ein formales "certified RNG"-Siegel erfordert die Abnahme durch ein akkreditiertes Prüflabor (GLI/BMM/iTech) sowie Lizenz-/Compliance-Prozesse je Jurisdiktion — ein externer Vorgang. Dieses Dokument ist eine technische Selbstprüfung, keine unabhängige Zertifizierung.