Software Requirements Specification (SRS)

Project Title: Credit-Gated Live Streaming & Chat Platform (React)

Version: 1.0
Date: 2025-08-22
Owner: <Your Company / Team>

Revision History

Version	Date	Author	Changes
1.0	2025-08-22	<You>	Initial draft

---

1. Introduction

1.1 Purpose

Define functional and non-functional requirements for a web platform where creators can live stream, viewers can watch, and usage is controlled by user credits. Credits are purchased via cards, crypto, and bank transfers. Chat with streamers is allowed only for users who have credits. Frontend is built in React/Next.js.

1.2 Scope

- Web application (desktop & mobile web) supporting: live streaming, viewer directory, credit wallet, payments, credit-gated chat, moderation, and admin analytics.
- Managed or self-hosted media service; vendor-agnostic design (e.g., Mux/Livepeer/AWS IVS or LiveKit/Nginx-RTMP).

1.3 Objectives

- Launch an MVP with reliable live playback and simple per-minute credit metering.
- Accept card, crypto, and bank transfer payments with webhook-verified credits.
- Provide creator dashboards and an admin console.
- Enforce chat access only for credit-holding users.

1.4 Stakeholders

- Product Owner, Engineering (FE/BE/DevOps), Design, QA, Compliance, Support, Creators, Viewers, Moderators.

1.5 Definitions

- **Credit**: Unit of value for consuming streams/chat features.
- **Ledger**: Immutable, double-entry record of credits.
- **Stream Session**: A single continuous live event by a creator.

---

2. System Overview

Single-page React app backed by a Node.js API (NestJS/Express) with Postgres. Media service handles ingest and playback (WebRTC→HLS). Payments integrate with Stripe (cards/bank) and Coinbase Commerce (crypto). Server performs credit metering during playback and controls chat eligibility.

---

3. User Roles & Permissions

- **Viewer**: register, buy credits, watch, chat (if eligible), follow creators.
- **Creator**: go live, manage streams, see earnings, moderate chat on own streams.
- **Moderator**: mute/ban users in chat, remove streams that violate policy.
- **Admin**: manage users, streams, payouts, disputes, pricing, view analytics.

RBAC enforced via JWT scopes.

---

4. Functional Requirements (FR)

4.1 Authentication & Profile

- **FR-A1**: Users can sign up/sign in via email/password and OAuth (Google/Apple optional).
- **FR-A2**: Email verification required before purchases or streaming.
- **FR-A3**: Profile edit: display name, avatar, bio; creator onboarding flag.

4.2 Creator Streaming

- **FR-S1**: Creators can start/stop a live stream; platform returns ingest key/URL.
- **FR-S2**: Stream metadata: title, category, tags, thumbnail.
- **FR-S3**: Transcoding to multiple bitrates; HLS playback URL generated.
- **FR-S4**: Stream status: scheduled, live, paused, ended.
- **FR-S5**: Optional DVR/recording for VOD (phase 2).

4.3 Discovery & Directory

- **FR-D1**: List all creators with filters: online, category, language.
- **FR-D2**: “Online now” shows all currently live streams.
- **FR-D3**: Search by creator name/stream title/tags.
- **FR-D4**: Creator profile page with live status and past sessions (if enabled).

4.4 Playback & Metering

- **FR-P1**: Viewers can watch live streams via HLS player in the browser.
- **FR-P2**: The frontend obtains a short-lived playback token (JWT) from the server.
- **FR-P3**: While playing, the client pings `POST /meter` every 10s; server debits credits based on configured rate (e.g., X credits/min).
- **FR-P4**: Playback stops if balance < required debit for next interval; a top-up dialog is shown.
- **FR-P5**: Metering intervals are idempotent by `session_id + interval_index`.

4.5 Wallet & Credits

- **FR-W1**: Each user has a wallet with current balance and a double-entry ledger.
- **FR-W2**: Ledger entries: DEPOSIT, DEBIT, REFUND, ADJUSTMENT; each with reference (payment_id or meter_event_id).
- **FR-W3**: On successful payment webhook, create a DEPOSIT.
- **FR-W4**: Expose `GET /wallet/balance`, `GET /wallet/ledger` with pagination.

4.6 Payments (Cards, Crypto, Bank Transfers)

- **FR-PM1**: Card & bank transfers via Stripe Checkout/Payment Links.
- **FR-PM2**: Crypto via Coinbase Commerce (or equivalent).
- **FR-PM3**: Webhooks verify payment signatures; idempotent processing.
- **FR-PM4**: Pricing tiers & currency mapping defined in admin.
- **FR-PM5**: Refunds and chargebacks create negative ledger adjustments.

4.7 Chat (Credit-Gated)

- **FR-C1**: Chat is available only if user's wallet balance > 0.
- **FR-C2**: Chat channel per stream; supports text and basic emojis.
- **FR-C3**: Rate limits: max 10 msgs/30s per user; flood protection.
- **FR-C4**: Moderation actions: mute (temp), ban, delete message.
- **FR-C5**: Optional credit cost per message (configurable; default 0) (phase 2).

4.8 Notifications

- **FR-N1**: In-app toasts for low balance, stream start of followed creators.
- **FR-N2**: Email/Push (phase 2) for scheduled stream reminders.

4.9 Admin & Moderation Console

- **FR-AD1**: Manage users, creators, streams, prices, categories.
- **FR-AD2**: View payments, disputes, and ledger anomalies.
- **FR-AD3**: Content policy flags & takedown with audit trail.

4.10 Analytics & Reporting

- **FR-AN1**: Stream metrics: CCU, watch time, bitrates, drop rates.
- **FR-AN2**: Revenue reports by source (card/crypto/bank) and period.
- **FR-AN3**: Creator earnings dashboard; export CSV.

5. Non-Functional Requirements (NFR)

- **NFR-1 Performance**: Player start time < 2.5s p95; live latency < 5s (HLS-LL) or < 500ms (WebRTC rooms), depending on mode.
- **NFR-2 Availability**: 99.9% monthly uptime target.
- **NFR-3 Scalability**: Support 100k MAU MVP; scale to 10k concurrent viewers.
- **NFR-4 Security**: JWTs signed with RS256; TLS 1.2+; OWASP Top 10 mitigations.
- **NFR-5 Privacy**: GDPR data export/delete; data retention policies.
- **NFR-6 Compliance**: PCI scope offloaded to provider; KYC/AML if creator payouts added (out of scope for MVP).
- **NFR-7 Observability**: Centralized logs, metrics, traces; alerting on SLOs.
- **NFR-8 Accessibility**: WCAG 2.1 AA; keyboard navigation.

---

6. External Interfaces

- **Media Service API**: Create/Delete livestream, get playback URL, webhooks for status (started/ended).
- **Payment Providers**: Stripe Checkout/PaymentIntents; Coinbase Commerce charge API; webhook endpoints for events.
- **Email/Push**: SendGrid/FCM (phase 2).

---

7. Data Model (Core Tables)

- `users(id, email, password_hash, role, status, created_at)`
- `profiles(user_id, display_name, avatar_url, bio, is_creator)`
- `streams(id, creator_id, title, category, status, ingest_url, playback_url, started_at, ended_at)`
- `stream_sessions(id, stream_id, status, created_at, ended_at)`
- `wallets(user_id, balance)`
- `ledger_entries(id, user_id, type, amount, currency, reference_type, reference_id, created_at)`
- `payments(id, provider, provider_ref, status, amount, currency, user_id, created_at)`
- `prices(id, product_code, credits, amount, currency, active)`
- `follows(user_id, creator_id)`
- `chat_messages(id, stream_id, user_id, message, created_at)`
- `moderation_actions(id, target_type, target_id, action, reason, actor_id, created_at)`

Indexes on `ledger_entries(user_id, created_at)`, `streams(status, started_at)`, `chat_messages(stream_id, created_at)`.

---

8. API Specification (MVP)

Auth
- `POST /auth/register`, `POST /auth/login`, `POST /auth/verify`

Users & Profiles
- `GET /me`, `PATCH /me/profile`

Streams
- `POST /streams` (creator) — returns ingest key/URL.
- `PATCH /streams/:id` — update metadata/status.
- `GET /streams?status=live|ended&category=...` — discovery.
- `GET /streams/:id` — details.
- `POST /streams/:id/token` — signed playback token (JWT).

Metering
- `POST /meter` — body: `{session_id, interval_index, playback_ms}` → debits.
- Response: `{remaining_credits, next_allowed_ms}`; idempotent by `(session_id, interval_index)`.

Wallet
- `GET /wallet/balance`, `GET /wallet/ledger?cursor=...`

Payments
- `POST /payments/checkout` — create Stripe Checkout session or Coinbase charge.
- `POST /webhooks/stripe`, `POST /webhooks/coinbase` — verify signature; on success create `DEPOSIT` and update `payments`.

Chat
- `GET /streams/:id/chat/token` — websocket/SSE token if balance > 0.
- `WS /chat` — join room, send messages; server enforces rate limits and eligibility.

Admin
- `GET /admin/analytics`, `GET /admin/payments`, `PATCH /admin/users/:id`, `POST /admin/prices`.

---

9. Key Workflows

9.1 Payment → Credits

1) FE calls `/payments/checkout` with selected price.
2) Redirect to provider (Stripe/Coinbase).
3) Provider → webhook → `/webhooks/<provider>` (signed).
4) BE verifies, upserts `payments` (status=SUCCEEDED), creates `ledger DEPOSIT`, updates

`wallets.balance`.
5) FE polls `GET /wallet/balance` or receives websocket event.

9.2 Start Live Stream (Creator)

1) Creator clicks “Go Live” → `POST /streams`.
2) BE creates livestream via media API; returns ingest key/URL.
3) Creator pushes via OBS/WebRTC publisher.
4) Media webhook → platform `status=live`; directory shows stream.

9.3 Playback with Metering

1) Viewer opens stream → FE requests `/streams/:id/token`.
2) FE starts HLS; every 10s posts `/meter` with `session_id` & `interval_index`.
3) BE debits credits atomically; returns remaining balance.
4) If insufficient, FE pauses and prompts top-up.

9.4 Credit-Gated Chat

1) FE requests chat token → server checks `wallet.balance > 0`.
2) If true, issue room token; else show top-up prompt.
3) Messages go via WS; moderator controls apply.

---

10. Security & Compliance

- JWT with RS256; short-lived playback tokens (≤ 5 min) with refresh.
- Webhooks: verify signatures; idempotency keys for retries.
- Least-privilege service accounts; secret rotation; WAF & rate limits.
- PII encryption at rest; audit logs for admin actions.
- PCI scope minimized by using hosted checkouts; no card data stored.

---

11. Environments & Deployment

- **Envs**: dev, staging, prod.
- **Infra**: Containerized (Docker), CI/CD, IaC (Terraform).
- **Storage**: Postgres (Neon/RDS), Object storage for thumbnails/VOD.
- **CDN**: For HLS segments & static assets.

---

12. Observability

- Metrics: API latency, error rates, metering success %, ledger mismatches, CCU.
- Logs: structured JSON; correlation IDs.

- Alerts: low ledger reconciliation, webhook failures, payment error spikes.

---

13. Acceptance Criteria & QA

- Play a test stream with 100 concurrent viewers; p95 start < 2.5s.
- Credit purchase updates balance within ≤ 30s of payment success.
- Metering stops playback when credits exhausted; no double-debits across retries.
- Chat denied when balance = 0; allowed when balance > 0.
- Webhook signature verification tested; idempotency validated.
- Admin can see real-time live list and revenue dashboard.

---

14. Phased Delivery

Phase 1 (MVP, 2–4 weeks): Auth, directory, go live, HLS playback, basic metering, Stripe + Coinbase, wallet, chat gating, admin basics.
Phase 2 (4–8 weeks): Recording/VOD, schedules/notifications, creator analytics, refunds, moderation tools.
Phase 3: Mobile web hardening, A/B pricing, affiliate links, payouts/KYC (if needed).

---

15. Risks & Mitigations

- Payment/webhook failures → retries + dead-letter queue.
- Media vendor outages → abstracted media adapter + fallback.
- Chargebacks → ledger adjustments & dispute workflow.
- Abuse/spam → chat rate limits, captcha on spikes, moderation.

---

16. Out of Scope (MVP)

- Native mobile apps.
- Creator fiat payouts and tax forms (KYC/AML).
- DRM and pay-per-message fees (configurable later).

---

17. Appendix: Glossary & Config

- **Credit Rate**: default 1 credit per 10s (configurable per creator/category).
- **Currencies**: USD (primary); FX table maps to credits.
- **Chat Limits**: 10 msgs/30s; 200 chars/msg.

End of SRS