EverstoneBTC

BITCOIN-ANCHORED MEMORIAL ARCHITECTURE

MEMORIAL TRANSACTION WORKFLOW

From tribute form to permanent Bitcoin proof.

Everstone creates a memorial record, accepts payment through Square or OpenNode, broadcasts an OP_RETURN anchor, then confirms and publishes the permanent proof after Bitcoin block confirmation.

1

Create memorial

User submits `/create`: name, dates, epitaph, bio, image/gallery, tier, email, public flag.

2

API validates + stores

`POST /api/memorials` enforces tier limits, generates slug + `bundleSecret`, writes Prisma `DRAFT`.

3

Checkout created

Payment route verifies token, calculates tier price + live Bitcoin network fee using mempool fee rates.

4

Customer pays

Square for card / Apple Pay.
OpenNode for Bitcoin / Lightning.

5

Webhook verifies

Provider webhook signature is checked. On completed payment, memorial transitions `UNPAID → PAID`.

6

Anchor lock

`withAnchorLock(memorial:id)` prevents concurrent treasury UTXO collisions and duplicate broadcasts.

7

Bitcoin transaction

Treasury wallet funds a real mainnet TX with OP_RETURN payload `EVST1:<memorialId>`.

8

Broadcast

TX hex posts to `mempool.space/api/tx`. Returned txid is saved.

9

Recovery + DB update

Anchor recovery log captures txid. Memorial becomes `BROADCAST` with txid; if anchoring fails it stays `PAID` for retry.

10

Confirmation cron

Every 5 minutes: `check-confirmations.sh` asks mempool for status of `BROADCAST` txids.

11

Confirmed forever

When mined: status `CONFIRMED`, `blockHeight` saved, explorer sorting updates.

12

Notify + publish

Receipt bundle email and one-time Twitter / Telegram memorial broadcast fire after confirmation.

EverstoneBTC

OPERATIONAL REFERENCE

IMPLEMENTATION NOTES

What each layer owns.

This page is the build-accuracy layer: routes, states, external services, and failure behavior that matter when debugging a memorial transaction.

Transaction state machine

DRAFT	Created by `POST /api/memorials`. Slug and private bundle secret exist, but no payment has completed.
UNPAID	Payment status before provider confirmation. Checkout is protected by the memorial's bundle/payment token.
PAID	Square/OpenNode webhook completed and signature verified. If Bitcoin anchoring fails, the record intentionally stays here for retry.
ANCHORING	Admin/manual anchor endpoint uses optimistic locking before calling the Bitcoin anchoring function.
BROADCAST	Bitcoin transaction has been broadcast and txid saved, but not yet mined into a block.
CONFIRMED	mempool.space reports a block. Everstone saves `blockHeight`, sends receipt bundle, and runs one-time social broadcasts.

Core components

Frontend	`/create`, `/m/[slug]`, `/explore`, `/memorial-success`.
Database	Prisma stores memorial content, payment status, txid, public flag, block height, referral metadata, and bundle secret.
Payments	`/api/payments/create-square` for card. `/api/payments/create` for OpenNode BTC/Lightning.
Anchoring	`lib/anchoring.ts` builds, signs, and broadcasts the treasury-funded Bitcoin transaction.
External APIs	mempool.space for fee rates, UTXOs, tx broadcast, and confirmation status. Square/OpenNode for payment confirmation.
Cron	`/home/sven/everstone-cron/check-confirmations.sh` runs the confirmation pass every five minutes.

Bitcoin data boundary

The chain anchor is intentionally compact: `EVST1:<memorialId>`. The full human memorial remains in Everstone's app/database and downloadable bundle; Bitcoin provides the permanent timestamp/proof pointer.

Receipt timing

Receipt email and social broadcast are deferred until block confirmation, not merely provider payment or mempool broadcast. That avoids announcing records before the anchor is mined.

Operational failure mode

Provider webhooks return safely and avoid retry storms. If anchoring fails after payment, the memorial remains `PAID` so cron/manual retry can recover without charging again.