System Architecture

The architecture consists of three primary components:

Frontend (Next.js + Web3.js)

Wallet Integration: Users (businesses and influencers) connect wallets via MetaMask/WalletConnect.

Campaign Management: Businesses create campaigns, selecting payout metrics.

Event Tracking UI: Displays influencer performance in real-time.

Smart Contract Interaction: Calls smart contract functions for campaign creation, tracking, and payout execution.

Backend (Node.js + Firebase + Web3.js)

Stores Campaign Metadata: Keeps details like influencer wallet, campaign type, payout structure.

Tracks Event Metrics: Listens for UTM-based tracking and logs user interactions.

Calls Smart Contract Functions: Aggregates performance data and triggers the final payout.

Smart Contract Layer (Solidity on Polygon)

Factory Contract: Deploys new campaign contracts dynamically.

InfluencerPayout Contract: Holds funds, tracks performance, and processes payouts.

Event Emissions: Emits logs for transparency and auditability.

Smart Contract Architecture

CampaignFactory (Factory Contract)

Deploys "InfluencerPayout" contracts dynamically.

Stores references to deployed campaign contracts.

Ensures each campaign contract is funded properly.

Functions:
`createCampaign()` → Deploys new "InfluencerPayout" contract and locks funds.   `getCampaigns()` → Returns list of deployed campaigns.

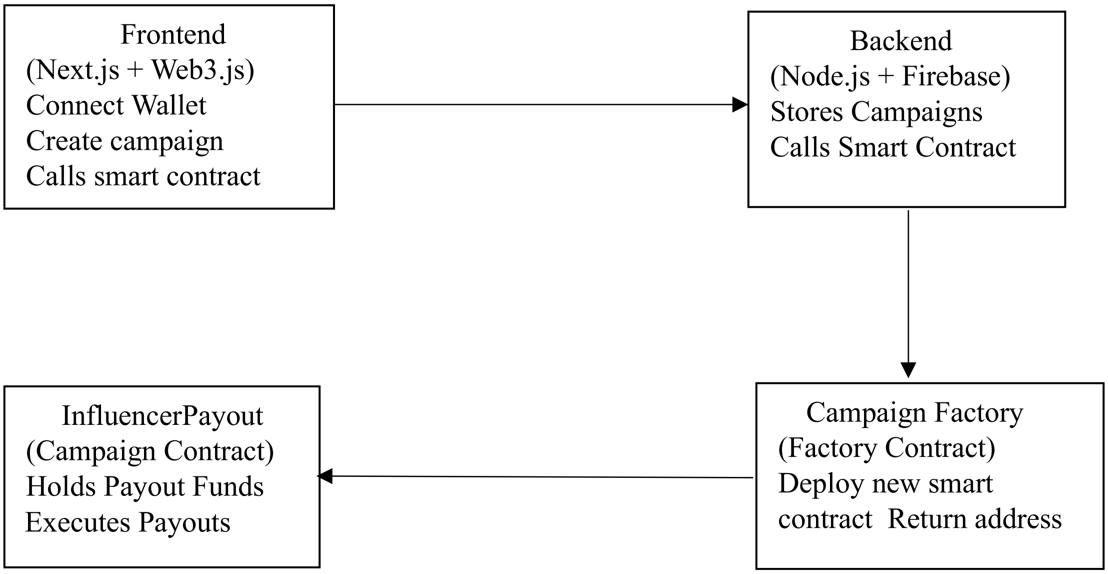
InfluencerPayout (Campaign Contract)

Holds campaign funds in escrow.

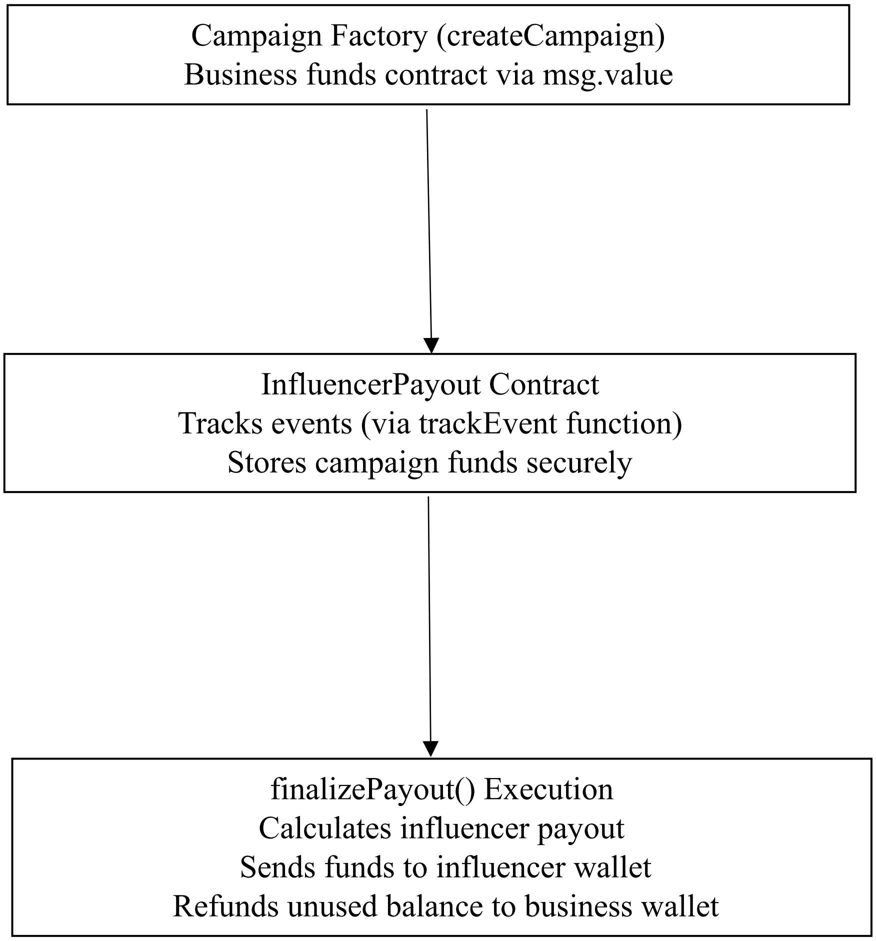
Executes payments based on predefined payout criteria.

Functions:
`finalizePayout()` → Calculates payout and releases funds. Also returns remaining funds to the business owners.

Architecture Diagram



Payout execution and fund flow



Campaign Factory (createCampaign)
Business funds contract via msg.value

InfluencerPayout Contract
Tracks events (via trackEvent function)
Stores campaign funds securely

finalizePayout() Execution
Calculates influencer payout
Sends funds to influencer wallet
Refunds unused balance to business wallet