



# Fundraisely

Introduction	2
High-Level Overview	2
Definitions, acronyms and abbreviations	3
Architecture constraints	3
Architecture Overview	3
C4 L1 Diagram: High-Level Architecture	3
C4 L2 Diagram: Zoom into the Fundraisely System	4
Contract Overview	4
Technology Stack	6
Backend	6
Frontend	6
Infrastructure	6
Automated Testing	6
Integrations	6



# Introduction

## High-Level Overview

FundRaisely is a full stack platform that enables charities, clubs, and grassroots communities to manage fundraising campaigns & events. With a donor and Sponsor CRM and AI prize finder, clubs can also host games of skill and games of chance within the platform and know they are fully compliant and transparent and verifiable on chain.

To launch we are building a multiplayer quiz system, optimised for fundraising, which leverages the Stellar blockchain to deliver transparent, regenerative fundraising.

Through this grant, we are launching the "Web3 Impact Event", a fundraising campaign powered by Glo Dollar, with transparent on-chain fund distribution and bridging to real-world charities via Coala Pay and The Giving Block while also onboarding our first Web2 Clubs and Communities.



## Definitions, acronyms, and abbreviations

- **USDGLO**: Non-profit - Glo Dollar stablecoin on Stellar, used for game fees and payment to Coala Pay for donations.
- **USDT**: Tether (used for bridging to The Giving Block)
- **Fundraising Extras**: Optional power-ups or features players can buy (e.g., clues, second chances) to help increase the funds raised by the club. Think of it like in game extras!
- **Allbridge**: Cross-chain bridge used to move funds from Stellar to EVM for donation.
- **Games of Skill** - Like Quizzes, Escape Rooms, Treasure Hunts, which require no licence or permit to run and compliance is easy to manage.
- **Games of Chance** - (gambling activities) Like Bingo, Raffles and Lotteries that do require licence or permit to run and compliance is more difficult to manage.

## Architecture constraints

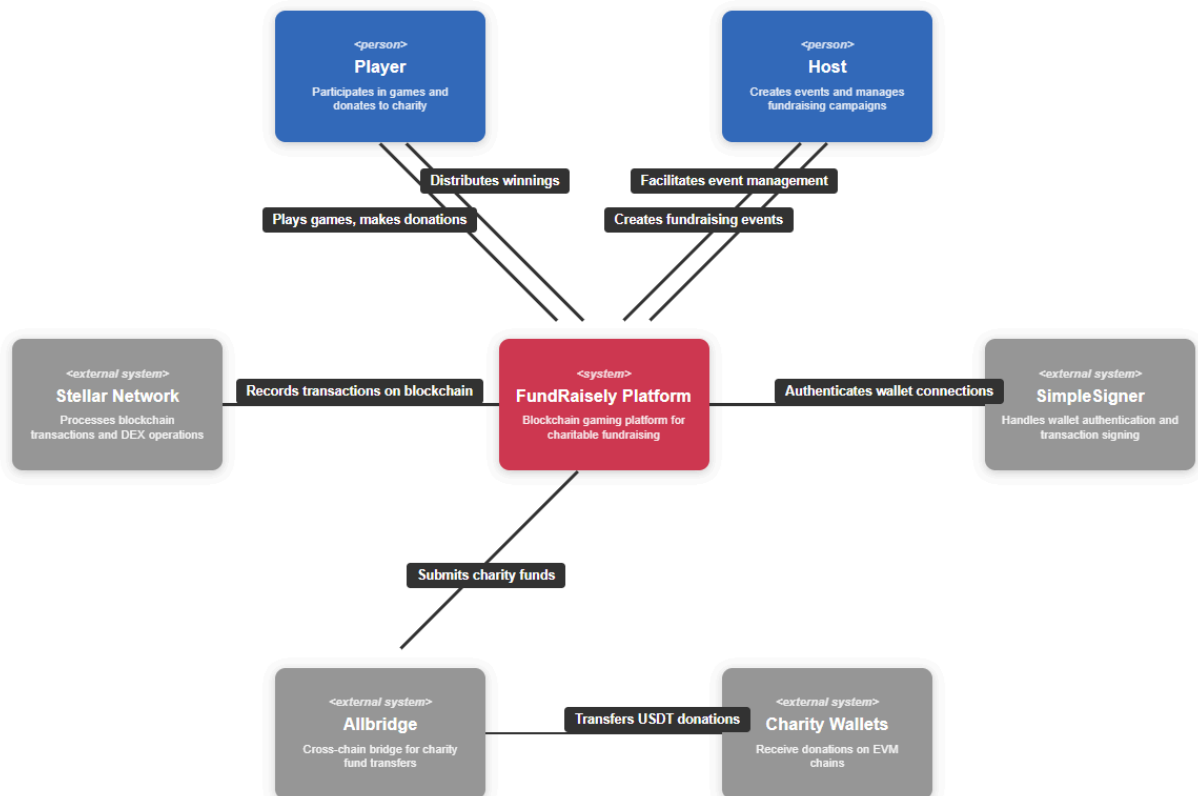
- Must support a minimum 40% donation to a registered charity.
- Max 5% host share, max 35% prize share (enforced in smart contract)
- Stellar-native support for USDGLO and USDT
- Cross-chain bridge must deliver USDT to The Giving Block
- An alternative now for donations on Stellar is Coala Pay.
- Must work in low-connectivity or mobile-first environments.
- SimpleSigner as wallet auth to reduce onboarding friction.



# Architecture Overview

## C4 L1 Diagram: High-Level Architecture

System Context diagram for FundRaisely Platform

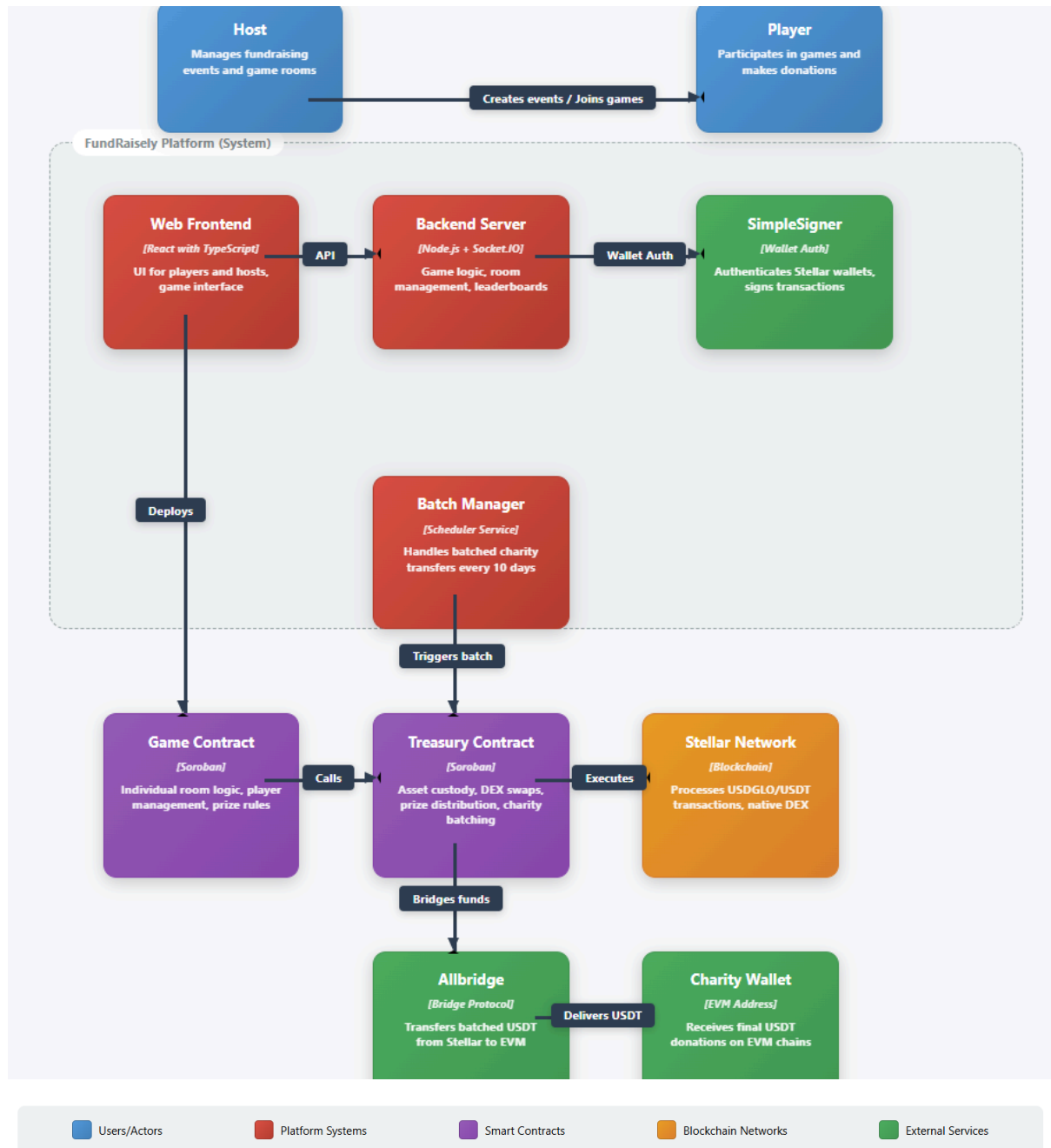




## C4 L2 Diagram: Zoom into the Fundraisely System

### Level 2: FundRaisely Gaming Platform

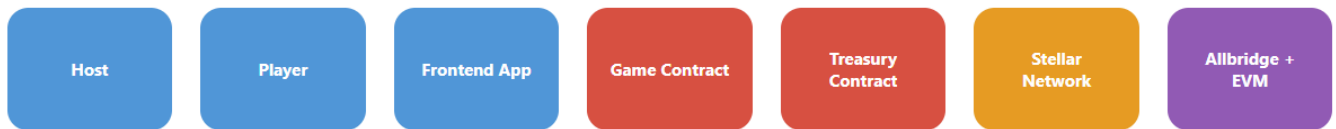
Container View - Stellar Architecture





# Contract Overview

## Stellar Gaming Platform Smart Contract Flow



### GAME SETUP PHASE

1. Host creates game config (entry fee, prize distribution %, charity wallet address)
2. Host deploys new Game Contract (individual contract per room)
3. Game Contract initialized with parameters and authorized with Treasury Contract

### PLAYER ENTRY PHASE

4. Player joins room, selects entry fee payment method (USDGLO or USDT)
5. Player pays entry fee via SimpleSigner → Game Contract → Treasury Contract
6. If USDGLO: Treasury Contract swaps to USDT via Stellar DEX
7. Treasury Contract holds all entry fees + any extra prizes from host

### GAME PLAY PHASE

8. Game played off-chain (frontend handles all game logic)



### GAME END & DISTRIBUTION PHASE

9. Host calls Game Contract.endGame(winners[], amounts[])
10. Game Contract calculates distributions based on preset rules
11. Game Contract calls Treasury Contract.distributePrizes()
12. Treasury Contract distributes player prizes on Stellar Network
13. Treasury Contract calculates charity portion for bridging
14. Treasury Contract calls Allbridge to bridge USDT to EVM charity wallet
15. Allbridge delivers bridged funds to designated EVM charity wallet
16. Game Contract marks game as complete and can be deactivated



Users/Frontend



Smart Contracts



Blockchain Networks



External Services

### Key Architecture of Smart Contracts:

- **Separate Treasury Contract:** Handles all DeFi operations, DEX swaps, and bridging
- **Individual Game Contracts:** Each room gets its own contract (Stellar/Soroban approach)
- **Native DEX Integration:** Uses Stellar's built-in DEX for USDGLO → USDT swaps
- **Cross-chain Bridging:** Allbridge integration for charity donations to EVM
- **SimpleSigner Integration:** Wallet-agnostic transaction signing
- **Multi-asset Support:** Handles stablecoins, tokens, and NFTs natively



# Technology Stack

## Backend

- Node.js + Express
- Socket.IO for real-time multiplayer
- In-memory game state (no persistent DB required yet)
- Stellar Blockchain

## Frontend

- React + TypeScript
- Zustand for state management
- SimpleSigner for wallet auth

## Infrastructure

- Railway (PaaS for backend)
- GitHub Actions (CI/CD)

## Automated Testing

- Soroban CLI & local net for contract testing
- E2E: Cypress (planned for MVP)
- Jest for frontend unit tests.

## Integrations

- Allbridge: For Glo → USDT cross-chain transfer
- Stellar DEX: For USDGLO stablecoin swaps
- The Giving Block: EVM-compatible USDT wallet target
- Coala Pay: Donations
- Glo Dollar (USDGLO): Regenerative stablecoin used for donations
- SimpleSigner: Wallet onboarding without browser extensions