This is the repo used for Stellar Build-a-thon.

0 stars 0 forks 0 watching Branches Activity
Tags

Public repository

main 1 Branch 0 Tags Go to file t Go to file Add file Code

Pswaikar1742 fix: Correct Option<Address> handling in smart contract calls 7df28f1 · 8 minutes ago

.devcontainer	feat: Initial project structure with devcontainer se...	yesterday
demo-accounts	fix: Update demo accounts and create backend i...	34 minutes ago
docs	docs: Add comprehensive technical documentati...	4 hours ago
documentation	docs: Clean README, add system stats and 3-mi...	4 hours ago
frontend	fix: Correct Option<Address> handling in smart ...	8 minutes ago
smart-contract	feat: Complete StellarPledge production system ...	14 hours ago
.gitignore	docs: Add comprehensive technical documentati...	4 hours ago
BACKEND_FIX_GUIDE.md	fix: Update demo accounts and create backend i...	34 minutes ago
BLOCKCHAIN_INTEGRATION_COMPLETE.md	docs: Add comprehensive blockchain integration...	11 minutes ago
CHANGELOG.md	docs: Add comprehensive technical documentati...	4 hours ago
DOCUMENTATION_CREATION_SUMMARY....	docs: Add comprehensive technical documentati...	4 hours ago
INTEGRATION_COMPLETE.md	docs: Add integration complete summary with fu...	25 minutes ago
ORGANIZATION_COMPLETE.md	docs: Add comprehensive technical documentati...	4 hours ago
ORGANIZATION_SUMMARY.md	docs: Add comprehensive technical documentati...	4 hours ago
README.md	docs: Clean README, add system stats and 3-mi...	4 hours ago
SYSTEM_STATS.md	docs: Clean README, add system stats and 3-mi...	4 hours ago

🌟 StellarPledge# 🌟 StellarPledge

The Automated Creator Economy Protocol on Stellar Blockchain****The Automated Creator Economy Protocol on Stellar Blockchain

Stellar Testnet Stellar Testnet

Soroban Smart Contracts Soroban Smart Contracts

React 19.0 React 19.0

License MIT License MIT

> Transform crowdfunding with automated reward distribution, trustless blockchain execution, and zero platform fees.> Transform crowdfunding with automated reward distribution, trustless blockchain execution, and zero platform fees.

Built for the **Stellar Build-a-thon** 🚀 Built for the **Stellar Build-a-thon** 🚀

🎯 The Problem## 🎯 Overview

Traditional crowdfunding platforms have:StellarPledge is a decentralized crowdfunding platform that leverages Stellar's blockchain and Soroban smart contracts to enable **automated perk distribution** - our core innovation that eliminates manual reward fulfillment.

- ❌ High platform fees (5-10%)
- ❌ Centralized control of funds### The Problem
- ❌ Manual reward distribution (weeks/months delay)Traditional crowdfunding platforms have:
- ❌ Limited transparency- ❌ High platform fees (5-10%)
- ❌ Trust-based intermediaries- ❌ Centralized control of funds
- ❌ Manual reward distribution (weeks/months delay)

✨ Our Solution- ❌ Limited transparency

StellarPledge is a decentralized crowdfunding platform leveraging Stellar blockchain with:### Our Solution

- ✅ Zero platform fees (only network costs ~$0.01)- ✅ Zero platform fees (only network costs ~$0.01)
- ✅ Decentralized smart contract escrow- ✅ Decentralized smart contract escrow
- ✅ **Automatic reward distribution in real-time** (Core Innovation!)- ✅ **Automatic reward distribution in real-time**
- ✅ Complete on-chain transparency- ✅ Complete on-chain transparency
- ✅ Trustless execution

✨ Key Features

🚀 Key Features

� Standalone Wallet System

🔐 Standalone Wallet SystemNo browser extensions required!

No browser extensions required!- **Create New Wallet** - Generate Stellar keypair in-app

- **Create New Wallet** - Generate Stellar keypair in-app- **Import Wallet** - Use existing secret key (S...)
- **Import Wallet** - Use existing secret key (S...)- **Read-Only Mode** - Browse with public key (G...)
- **Read-Only Mode** - Browse with public key (G...)- **Secure Encryption** - Password-protected local storage
- **Secure Encryption** - Password-protected local storage

🎁 Automated Perk Distribution (Core Innovation ⭐)

When a backer pledges enough to meet the perk threshold, the smart contract **automatically** triggers a cross-contract call to transfer reward tokens - all in ONE atomic transaction!### Why StellarPledge?[Read Standalone Wallet Documentation →]

Example Flow:

```
Alice creates campaign:**Traditional Crowdfunding:**### 👥 Role-Based Access
  Goal: 10,000 XLM
  Perk threshold: 500 XLM → 1 FILMCREDIT token- ❌ High fees (5-10%)

Bob pledges 100 XLM → No perk (below threshold)- ❌ Manual reward fulfillment- **Creators** - Launch campaigns with custom
reward tiers---

Charlie pledges 500 XLM → 🎉 INSTANT REWARD!- ❌ Limited transparency

Result: TWO operations in ONE transaction:- ❌ Centralized control- **Funders** - Back projects and earn tokenized rewards
  1. 500 XLM → Campaign escrow
  2. 1 FILMCREDIT → Charlie's wallet (automatic!)
```

StellarPledge:- **Multi-User Support** - Independent sessions per browser tab## 🚀 The Problem

No manual fulfillment. No delays. All on-chain. Provable.

- ✅ Minimal fees (<$0.01)

👥 Role-Based Access

- **Creators** - Launch campaigns with custom reward tiers- ✅ Automated rewards
- **Funders** - Back projects and earn tokenized rewards
- **Multi-User Support** - Independent sessions per browser tab- ✅ Complete transparency

🎨 Modern UI/UX- ✅ Decentralized trustless### 💰 Smart Campaign ManagementTraditional crowdfunding platforms are centralized, slow, and expensive. They take significant fees and act as a trusted middleman, creating a barrier between creators and their communities. Reward distribution is manual, error-prone, and can take months.

- Responsive design (mobile, tablet, desktop)
- Dark/Light themes
- Smooth animations (Framer Motion)
- Real-time toast notifications---- **Goal Setting** - Define funding targets in XLM
- Intuitive campaign creation wizard

✨ Key Features- Deadline Management - Time-bound campaigns---

🚀 Quick Start

Prerequisites

- Node.js 16+ and npm### For Creators- **Reward Tiers** - Threshold-based token distribution
- Git
- 📝 **Campaign Creation Wizard**: Easy 3-step process

Installation

- 🎁 **Configurable Rewards**: Set threshold-based token rewards- **Real-Time Progress** - Live updates across all users## ✨ Our Solution: StellarPledge

~~~
# Clone the repository- 📊 **Real-Time Analytics**: Track pledges, backers, and progress

git clone https://github.com/Pswaikar1742/StellarPledge.git

cd StellarPledge- 💰 **Instant Fund Access**: Claim funds when goal is met


# Install frontend dependencies- 🔐 **Secure Wallet Integration**: Create or import Stellar wallets

cd frontend

npm install### 🎁 Automated RewardsStellarPledge is a **trustless, open-source protocol** that leverages Soroban smart contract


# Start development server### For Funders

npm start

```- 🔍 **Campaign Discovery**: Browse active campaigns- **Token Distribution** - Automatic reward tokens for qualifying pledge


The app will open at `http://localhost:3000`- 💎 **Reward Preview**: See eligible rewards before pledging


### First Time Setup- ⚡ **Instant Pledges**: Blockchain-fast transactions- **Threshold Logic** - Smart contract validates pled


1. **Clear browser storage** (F12 console):- 🏆 **Automatic Rewards**: Receive tokens at threshold

```javascript

localStorage.clear();- 📈 **Portfolio Tracking**: View all backed campaigns- **Instant Delivery** - Rewards issued immediately

location.reload();
~~~

2. **Create demo users**:### Technical Innovation- **On-Chain Proof** - All transactions visible on StellarWhen a backer pledges enough to meet the perk threshold, the smart contract **automatically** triggers a cross-contract call to transfer reward tokens (any Stellar Classic Asset) to the backer - all in one atomic transaction!

```
window.setupDemoUsers()- 🤖 **Smart Contract Automation**: Cross-contract token transfers
```

- 🔄 **Event-Driven Architecture**: Real-time multi-user updates

3. **Login with demo accounts**:
   - **Alice** (Creator): `alice@example.com` / `alice123` - 🎨 **Modern UI/UX**: Tailwind CSS with Framer Motion animations
   - **Bob** (Funder): `bob@example.com` / `bob123`
   - **Charlie** (Funder): `charlie@example.com` / `charlie123` - 📱 **Responsive Design**: Works on all devices### 🎨 Modern UI/UX**Example:**

---- 🌓 **Dark/Light Mode**: User preference support

📁 Project Structure- Responsive Design - Works on all devices```

~~~
StellarPledge/

├── frontend/                    # React application- **Dark/Light Themes** - Toggle between modesAlice creates campaign: 10,000 XLM goal, 500 XLM perk threshold → 1 FILMCREDIT token reward

│   ├── src/

│   │   ├── components/        # UI components## 🚀 Quick Start

│   │   ├── pages/             # 8 main pages

│   │   ├── contexts/          # State management- **Smooth Animations** - Framer Motion poweredBob pledges 100 XLM → No perk (below threshold)

│   │   ├── services/          # Wallet & Blockchain services

│   │   └── utils/             # Helper functions### Prerequisites

│   └── package.json

|- **Toast Notifications** - Real-time feedbackCharlie pledges 500 XLM → 🎉 INSTANT: Gets 1 FILMCREDIT automatically!

├── smart-contract/            # Soroban smart contract

│   ├── src/- Node.js 16+ and npm

│   │   └── lib.rs            # 5 core functions (213 LOC)

│   └── Cargo.toml- Git```

│

├── documentation/             # 420+ pages of docs

│   ├── demo/                 # Demo scripts

│   ├── technical/            # Architecture guides### Installation---

│   └── guides/               # How-to guides

│

└── SYSTEM_STATS.md           # Project statistics

``````bash**Result:** Two operations in ONE transaction:



---# Clone the repository



## 🛠️ Technology Stackgit clone https://github.com/Pswaikar1742/StellarPledge.git## 🎯 Why StellarPledge?1. 500 XLM → Campaign escrow



### Frontendcd StellarPledge

- **Framework:** React 19.2.0

- **Styling:** Tailwind CSS + Framer Motion2. 1 FILMCREDIT → Charlie's wallet (automatic!)

- **Blockchain:** Stellar SDK 11.3.0

- **Wallet:** Standalone (AES encryption)# Install frontend dependencies

- **UI Components:** Radix UI + Lucide Icons
~~~

~~~~
cd frontendTraditional crowdfunding platforms have significant limitations:

### Smart Contract

- **Language:** Rustnpm install

- **SDK:** Soroban SDK 20.3.1

- **Network:** Stellar Testnet- ❌ High platform fees (5-10%)---

- **Functions:** 5 core contract functions

# Start development server

### Security

- AES-256-GCM encryption for private keysnpm start- ❌ Centralized control of funds

- Password-protected keystores

- Client-side session management```

- Input validation and sanitization

- ❌ Slow payouts (weeks/months)## 🏆 Key Features

---

The app will open at `http://localhost:3000`

## 🔗 Smart Contract

- ❌ Manual reward distribution

**Deployed on Stellar Testnet:**

- **Contract ID:** `CCL2RYVGR4RWLB6Q6JSMRGKIUUYLDKSJGUDCGNPMXGVDMHFLLHMTNVUH`### First Time Setup

- **Explorer:** [View on Stellar Expert]
(https://stellar.expert/explorer/testnet/contract/CCL2RYVGR4RWLB6Q6JSMRGKIUUYLDKSJGUDCGNPMXGVDMHFLLHMTNVUH)

- ❌ Limited transparency### Decentralized Escrow

### Core Functions

```rust1. **Open browser console** (F12)

create_campaign()  // Create new campaign with optional perks

pledge()           // Pledge with automatic reward distribution2. **Create demo users**:All funds are held securely on-chain
in the Soroban smart contract. No intermediaries, no platform control.

claim_funds()      // Creator claims funds after success

withdraw_refund()  // Backer gets refund if campaign fails```javascript

get_campaign()     // Retrieve campaign details

```window.setupDemoUsers()**StellarPledge solves these problems:**



### Automated Perk Logic```

```rust

if let Some(perk) = &campaign.perk {- ✅ Zero platform fees### Instantaneous Settlements

    if total_backer_pledge >= perk.threshold {

        // Cross-contract call to Stellar Asset Contract3. **Login with demo accounts**:
~~~~

~~~
        let perk_token_client = token::Client::new(&env, &perk.asset_address);

        perk_token_client.transfer(&campaign.creator, &backer, &perk.amount);   - Creator: `alice@example.com` / `alice123`-
✅ Decentralized smart contract escrowPayouts and refunds are processed in ~5 seconds via Stellar's fast consensus.

        log!("✅ Perk transferred automatically!");

    }   - Funder: `bob@example.com` / `bob123`

}

```   - Funder: `charlie@example.com` / `charlie123`- ✅ Instant settlements (~5 seconds)



---



## 🎬 Live Demo---- ✅ Automated reward distribution### Micro-Pledge Capable



### Quick Demo (2 minutes)

1. Alice creates campaign: Goal 6,000 XLM, reward at 3,000 XLM

2. Bob pledges 2,000 XLM → No reward## 📁 Project Structure- ✅ Complete transparency on-chainNear-zero fees on the Stellar
network enable community micro-funding. Support creators with any amount.

3. Charlie pledges 4,000 XLM → Earns reward! 🎉



### Full Demo (8-9 minutes)

Follow our **[Live Demo Script](documentation/demo/LIVE_DEMO_SCRIPT.md)** for complete presentation.```



### Demo Accounts (Pre-funded Testnet)StellarPledge/

| Name | Role | Public Key | Funding |

|------|------|------------|---------|├── frontend/                # React application---### Automated On-Chain Perks

| **Alice** | Creator | GA4N...P5TM | 10,000 XLM |

| **Bob** | Backer | GBQH...PNDE | 10,000 XLM ||   ├── src/

| **Charlie** | Backer | GDBJ...H33Y | 10,000 XLM |

│   │   ├── components/     # Reusable UI components**Our core innovation!** Smart contract automatically rewards backers
with custom asset tokens (collectibles, access tokens, NFTs) when they meet pledge thresholds.

**Verify on stellar.expert:** All accounts are live and funded on testnet!

│   │   ├── pages/          # Main application pages

---

│   │   ├── context/        # React Context providers## 🚀 Quick Start

## 📚 Documentation

│   │   ├── services/       # Stellar/Wallet services

### 🎬 Demo & Presentation

- **[Live Demo Script](documentation/demo/LIVE_DEMO_SCRIPT.md)** ⭐ - 8-9 minute presentation script│   │   ├── utils/
# Helper functions### Three Perk Modes

- **[3-Minute Pitch](documentation/demo/3_MINUTE_PITCH.md)** - Quick pitch for judges
~~~

- **[Complete Demo Guide](documentation/demo/COMPLETE_DEMO_GUIDE.md)** - Full walkthrough│   │   └── constants/   # Configuration constants

- **[Pre-Demo Checklist](documentation/demo/PRE_DEMO_CHECKLIST.md)** - Setup before presentation

│   └── public/   # Static assets### Prerequisites- **Automatic:** Smart contract handles distribution (recommended)

### 🔧 Technical Documentation

- **[System Architecture](documentation/technical/SYSTEM_ARCHITECTURE.md)** - Complete system design│

- **[Smart Contract Architecture](documentation/technical/SMART_CONTRACT_ARCHITECTURE.md)** - Rust contract deep dive

- **[Wallet System](documentation/technical/WALLET_SYSTEM_ARCHITECTURE.md)** - Wallet implementation├── smart-contract/ # Soroban smart contract- Node.js v16+ and npm- **Manual:** Creator distributes rewards later

- **[Frontend Architecture](documentation/technical/FRONTEND_ARCHITECTURE.md)** - React app structure

- **[Stellar Expert Guide](documentation/technical/STELLAR_EXPERT_GUIDE.md)** - Blockchain verification│   ├── src/

### 📖 Guides│   │   └── lib.rs   # Main contract code- Modern web browser (Chrome, Firefox, Edge, Safari)- **No Perks:** Simple crowdfunding only

- **[Demonstration Q&A](documentation/guides/DEMONSTRATION_QA.md)** - 27 technical questions

- **[Testing Guide](documentation/guides/TESTING-GUIDE.md)** - Manual testing instructions│   └── Cargo.toml   # Rust dependencies

- **[Real Accounts Guide](documentation/guides/REAL_FUNDED_ACCOUNTS_GUIDE.md)** - Using testnet accounts

│

### 📊 Project Metrics

- **[System Stats](SYSTEM_STATS.md)** - Complete project statistics├── documentation/   # All documentation

---│   ├── demo/   # Demo scripts and guides### Installation### Transparent & Trustless

## 🌐 Use Cases│   ├── technical/   # Technical documentation

### 🎬 Film Production│   ├── guides/   # User guides- Every transaction viewable on Stellar Expert

Campaign: Fund indie film (5,000 XLM goal)

Perk: Early screening ticket NFT at 1,000 XLM pledge│   └── FINAL_DELIVERY_SUMMARY.md

### 🎵 Music Album│```bash- Open-source smart contract code

Campaign: Record album (2,000 XLM goal)

Perk: Exclusive track token at 500 XLM pledge├── demo-accounts/   # Funded testnet accounts

### 🎮 Game Development│   ├── Alice.txt# Clone the repository- No platform fees

Campaign: Build indie game (10,000 XLM goal)

Perk: In-game currency at 2,000 XLM pledge│   ├── Bob.txt

~~~
### 📚 Book Publishing│   └── Charlie.txtgit clone https://github.com/Pswaikar1742/StellarPledge.git- Provable on-chain
ownership

**Campaign:** Publish novel (1,000 XLM goal)

**Perk:** Signed digital copy at 200 XLM pledge│



---└── README.md             # This filecd StellarPledge



## 💡 Key Innovations```



### 1. Automated On-Chain Perks ⭐---

First crowdfunding platform with **instant, automatic** reward distribution via cross-contract calls.

---

### 2. Standalone Wallet System

No browser extensions needed - complete wallet management built into the app.# Install frontend dependencies



### 3. Threshold-Based Incentives## 📚 Documentation

Smart contract validates pledge amounts and triggers rewards automatically.

cd frontend## 🛠️ Technology Stack

### 4. Any Stellar Asset as Reward

Support for any token: NFTs, access tokens, collectibles, memberships.### 🎬 Demo & Presentation



---- **[Live Demo Script](documentation/demo/LIVE_DEMO_SCRIPT.md)** ⭐ - Copy/paste ready presentation (8-9 min)npm install



## 📊 Project Status- **[Complete Demo Guide](documentation/demo/COMPLETE_DEMO_GUIDE.md)** - Full walkthrough with
verification



```- **[Pre-Demo Checklist](documentation/demo/PRE_DEMO_CHECKLIST.md)** - Setup before presentation- **Blockchain:** Stellar
Testnet (Soroban)

✅ Smart Contract:     100% Complete (Deployed on testnet)

✅ Standalone Wallet:  100% Complete (3 connection modes)- **[Demo Changes](documentation/demo/DEMO_CHANGES.md)** - Recent
updates

✅ Frontend UI:        100% Complete (8 pages, 16+ components)

✅ Documentation:      100% Complete (420+ pages)# Start development server- **Smart Contract:** Rust + Soroban SDK 20.5.0

✅ Demo Accounts:      100% Complete (3 funded accounts)

✅ System Testing:     100% Complete (Manual testing)### 🔧 Technical Documentation
~~~

- Current State Explanation - Mock vs Real blockchainnpm start- **Frontend:** React 19.2.0

Status: 🟢 **PRODUCTION READY** for demonstration

- Wallet Balance Guide - Balance system architecture

- Real Accounts Integration - Testnet integration```- **Wallet:** Standalone wallet system (no extensions required)

🔗 Quick Links

- Repository: github.com/Pswaikar1742/StellarPledge
- Smart Contract: View on Stellar Expert### 📖 User Guides- **SDK:** Stellar SDK 11.3.0
- Demo Accounts:
  - Alice (Creator)- **Real Funded Accounts Guide** - Using testnet accounts
  - Bob (Backer)
  - Charlie (Backer)- **Testing Guide** - Manual testing instructionsThe application will open at: **http://localhost:3000**- **Security:** AES encryption, password-protected keystores

👥 Team### 📊 Project Summary

Team of 3 passionate blockchain developers building the future of creator economy on Stellar!- **Final Delivery Summary** - Complete project overview

Built for the **Stellar Build-a-thon** with ❤️### First-Time Setup### Smart Contract Architecture

📄 License

MIT License - See LICENSE file for details## 🎬 Demo

---1. **Clear Browser Storage** (Press F12 to open console):
```rust

**🌟 StellarPledge: Where Creators Meet Their Community On-Chain 🌟**### Quick Demo (2 minutes)

*Last Updated: October 26, 2025 | Version: 1.0.0 | Status: Demo Ready*```javascript// Automated perk distribution logic

1. Start server: `npm start` in `/frontend`
2. Setup: `window.setupDemoUsers()` in consolelocalStorage.clear();if let Some(perk) = &campaign.perk {
3. Alice creates campaign: Goal 6,000 XLM, reward at 3,000 XLM
4. Bob pledges 2,000 XLM: No rewardlocation.reload(); if total_backer_pledge >= perk.threshold {
5. Charlie pledges 4,000 XLM: Earns reward! 🎉

```
### Full Demo (8-9 minutes)

        let perk_token_client = token::Client::new(&env, &perk.asset_address);

Follow **[Live Demo Script](documentation/demo/LIVE_DEMO_SCRIPT.md)** for complete presentation.

2. **Load Demo Accounts**:   perk_token_client.transfer(&campaign.creator, &backer, &perk.amount);

```javascript        log!("✅ Perk transferred automatically!");

## 🛠 Technology Stack

window.setupDemoUsers()     }
```

### Frontend

- React 18, Tailwind CSS, Framer Motion```}

- React Router, Lucide Icons

- @stellar/stellar-sdk```

### Smart ContractThis creates three test accounts:

- Rust, Soroban SDK

- Stellar Testnet- **Alice** (Creator): `alice@example.com` / `alice123`**Key Innovation:** Cross-contract calls enable automatic token transfers without manual intervention.

### Tools- **Bob** (Funder): `bob@example.com` / `bob123`

- Node.js, npm, Webpack, ESLint

- **Charlie** (Funder): `charlie@example.com` / `charlie123`---

## 🔗 Smart Contract

---## 🎬 Live Demo

### Features

- ✅ Campaign creation with parameters

- ✅ Pledge with automatic reward distribution

- ✅ Threshold-based incentives## 📖 Complete Demo Guide**Deployed Smart Contract:**

- ✅ Refund mechanism

- ✅ Fund claiming- Network: Stellar Testnet

### Key Functions### **See COMPLETE_DEMO_GUIDE.md for full demonstration instructions**- Address: `CD4L4MPVSJ3RLAUYQ3ID2M75VWVVMDFBTESJIY4UULFFN33X2KNRTJXY`

~~~rust

create_campaign()  // Create new campaign- Explorer: [View on Stellar Expert](https://stellar.expert/explorer/testnet/contract/CD4L4MPVSJ3RLAUYQ3ID2M75VWVVMDFBTESJIY4UULFFN33X2KNRTJXY)

pledge()           // Pledge with auto rewards

claim_funds()      // Claim after successQuick demo flow (8-9 minutes):

withdraw_refund() // Refund if failed

```1. **Alice** creates campaign (6000 XLM goal, 3000 XLM reward tier)**Frontend Application:**

### Status2. **Bob** pledges 2000 XLM (below threshold, no reward)- URL: `https://localhost:3000` (local dev)

- 📝 Code: Complete

- 🔄 Deployment: Ready for testnet3. **Charlie** pledges 4000 XLM (above threshold, earns FOUNDER token)- Status: Standalone wallet 100% complete, campaign UI in development

- 📍 Location: `/smart-contract/src/lib.rs`
~~~

```
4. Campaign reaches goal → Alice sees "Pending Approval" status

---

### Demo Flow (5 Minutes)

## 🎉 Demo Accounts

---

| Name | Role | Email | Password |

|------|------|-------|----------|1. **Alice (Creator)** creates campaign: 10,000 XLM goal, 500 XLM perk threshold

| Alice | Creator | alice@example.com | alice123 |

| Bob | Funder | bob@example.com | bob123 |## 🏗️ Architecture2. **Bob (Student)** pledges 100 XLM → No perk (below
threshold)

| Charlie | Funder | charlie@example.com | charlie123 |

3. **Charlie (Investor)** pledges 500 XLM → 🎉 **Automatic FILMCREDIT transfer!**

**Setup**: `window.setupDemoUsers()` in console

### Tech Stack4. View on Stellar Expert: Two operations in one transaction

---

- **Frontend:** React 18, Tailwind CSS, Framer Motion5. Alice claims 600 XLM when goal met

## 📞 Quick Links

- **Blockchain:** Stellar Network (Testnet)

- **Repository**: [github.com/Pswaikar1742/StellarPledge](https://github.com/Pswaikar1742/StellarPledge)

- **Demo Script**: [LIVE_DEMO_SCRIPT.md](documentation/demo/LIVE_DEMO_SCRIPT.md)- **Smart Contract:** Rust + Soroban SDK**Wow
Factor:** Live cross-contract call with on-chain proof!

- **Documentation**: [/documentation](documentation/)

- **Wallet:** Custom wallet system with AES encryption

---

- **State Management:** React Context + Custom Hooks---

## 🌟 Highlights

- **Routing:** React Router v6

- ✨ Automated threshold-based rewards

- 🚀 Complete full-stack implementation## 🚀 Quick Start

- 💎 Production-quality code and UI

- 📚 Comprehensive documentation### Project Structure

- 🎬 Tested demo flow

- 🔗 Stellar blockchain native```### Prerequisites



---StellarPledge/



**Built with ❤️ for the Stellar ecosystem**├── frontend/                    # React application- **Node.js** (v16+) and npm
```

~~~
*Last Updated: October 26, 2025 | Version: 1.0.0 | Status: Demo Ready*│   ├── src/- **Stellar testnet account** with XLM ([Get from Friendbot](https://laboratory.stellar.org/#account-creator?network=test))

│   │   ├── components/          # UI components- **No wallet extension required** - Built-in standalone wallet system

│   │   │   ├── ui/             # Reusable UI primitives

│   │   │   └── Wallet/         # Wallet management### Installation

│   │   ├── context/            # Global state

│   │   ├── pages/              # Route pages```bash

│   │   ├── services/           # API services# Clone the repository

│   │   ├── utils/              # Helper functionsgit clone https://github.com/Pswaikar1742/StellarPledge.git

│   │   └── App.js              # Main app componentcd StellarPledge

│   └── package.json

├── smart-contract/              # Soroban smart contract# Install frontend dependencies

│   ├── src/cd frontend

│   │   └── lib.rs              # Contract logicnpm install

│   └── Cargo.toml

├── COMPLETE_DEMO_GUIDE.md       # Full demonstration guide# Start the development server (HTTPS required for Freighter)

├── PRE_DEMO_CHECKLIST.md        # Pre-presentation checklistnpm start

├── WALLET_BALANCE_GUIDE.md      # Balance system docs```

└── README.md                    # This file

```The application will open at: **https://localhost:3000**


---### Connect Your Wallet


## 💡 Key Features in DetailChoose one of three connection modes:


### Wallet System1. **Create New Wallet** - Generate a new Stellar keypair

- **3 Connection Modes:** Create, Import, Read-Only2. **Import Existing Wallet** - Import using your secret key (S...)

- **Password Protection:** AES encryption for secret keys3. **Connect Read-Only** - View campaigns using public key (G...)

- **Mock Balances:** 10,000 XLM initial balance for testing

- **Real-Time Updates:** Balance changes reflect immediately### Build Smart Contract


### Campaign Management```bash

- **3-Step Creation Wizard:**cd smart-contract

  1. Basic Info (Title, Description)cargo build --target wasm32-unknown-unknown --release

  2. Funding Goal (Amount, Duration)

  3. Reward Tier (Threshold, Token details)# Optimize (optional)
~~~

~~~~
  soroban contract optimize --wasm target/wasm32-unknown-unknown/release/stellar_pledge_contract.wasm

- **Status Tracking:**```

  - Active: Currently accepting pledges

  - Pending Approval: Goal reached, awaiting creator action---

  - Successful: Funds claimed by creator

  - Failed: Goal not met, refunds available## 📋 Project Structure


### Pledge System```

- **Real-Time Validation:** Check balance before pledgeStellarPledge/

- **Threshold Preview:** See if you'll earn rewards├── README.md                    # This file

- **Instant Confirmation:** Toast notifications with details├── docs/                        # Complete documentation

- **Balance Updates:** Immediate deduction from wallet│   ├── COMPLETE-WORKFLOW.md     # Step-by-step user journeys

│   ├── IMPLEMENTATION-SUMMARY.md # Technical implementation details

### Multi-User Support│   ├── PROJECT-STATUS.md        # Current status & roadmap

- **Independent Sessions:** Each tab maintains separate user│   └── QUICK-REFERENCE.md       # Quick start guide

- **Event-Based Sync:** Real-time updates across components├── demo-accounts/               # Demo account details

- **localStorage Persistence:** Data survives page refreshes│   ├── Alice.txt                # Creator account

│   ├── Bob.txt                  # Small backer

---│   └── Charlie.txt              # High backer

├── smart-contract/              # Soroban smart contract

## 🎮 User Journeys│   ├── Cargo.toml

│   └── src/

### Creator Journey (Alice)│       └── lib.rs               # Automated perk distribution logic

1. Register/Login → Select "Create Campaign" role└── frontend/                    # React application

2. Connect wallet → Initial 10,000 XLM    ├── src/

3. Create campaign with reward tier    │   ├── components/          # UI components

4. Monitor pledge progress    │   ├── context/             # State management

5. Claim funds when goal reached    │   ├── hooks/               # Custom React hooks

    │   ├── utils/               # Error handling, validation

### Funder Journey (Bob/Charlie)    │   └── constants/           # Configuration

1. Register/Login → Select "Fund Projects" role    ├── INTEGRATION-GUIDE.md     # Frontend API documentation

2. Connect wallet → Initial 10,000 XLM    └── package.json

3. Browse active campaigns```

4. View campaign details

5. Make pledge → Earn rewards if qualifying---

6. View backed campaigns in dashboard
~~~~

~~~~
## 🎮 How It Works

---

### For Creators

## 🛠️ Development

### For Creators

### Available Scripts

**Setup (One-Time)**

```bash1. Install Freighter wallet → Switch to Testnet

# Start development server2. Create your custom perk token (e.g., FILMCREDIT)

npm start3. Deploy asset contract to Stellar



# Build for production**Launch Campaign**

npm run build1. Visit StellarPledge → Connect wallet

2. Click "Create Campaign"

# Run tests3. Set goal, deadline, and optional perk threshold

npm test4. Choose perk mode: Automatic, Manual, or None

5. Approve transaction → Campaign goes live!

# Eject configuration (one-way operation)

npm run eject**Outcome**

```- ✅ **Success:** Claim all funds when goal met

- ❌ **Failed:** Backers get automatic refunds

### Debug Commands

### For Backers

Open browser console (F12) and use:

**Browse & Pledge**

```javascript1. Install Freighter → Switch to Testnet

// Check current user2. Browse active campaigns

window.whoAmI()3. Select campaign → Enter pledge amount

4. See perk preview if you qualify

// View all campaigns5. Approve transaction → Done!

window.viewCampaigns()

**Automated Perks**

// View all users- Pledge ≥ threshold → Get perk token instantly

window.viewUsers()- All in one atomic transaction

- On-chain proof of ownership

// Check wallet balances

window.walletBalance.getAll()**Refunds**
~~~~

```
 - Campaign fails → Withdraw refund button appears

 // Initialize balance- One click → Instant refund

 window.walletBalance.init("GXXX...", 10000)- No manual process required


 // Reset everything---

 localStorage.clear()

 location.reload()## 🎯 Use Cases
```

🎬 Film Production

---Campaign: Fund indie film

Perk: Early screening ticket NFT at 1000 XLM

📚 Documentation

🎵 Music Albums

- COMPLETE_DEMO_GUIDE.md - Full demonstration guide with step-by-step instructionsCampaign: Record new album
- PRE_DEMO_CHECKLIST.md - Pre-presentation checklist and troubleshootingPerk: Exclusive track token at 500 XLM
- WALLET_BALANCE_GUIDE.md - Mock wallet balance system documentation
- DEMO_GUIDE.md - Quick reference guide### 🎮 Game Development
- DEMO_CHANGES.md - Technical implementation detailsCampaign: Build indie game

Perk: In-game currency at 2000 XLM

📚 Book Publishing

🎯 Use CasesCampaign: Publish novel

Perk: Signed digital copy at 200 XLM

🎬 Film Production

Launch campaigns to fund indie films with premiere ticket NFTs as rewards---

🎵 Music Albums## 🔐 Security Features

Raise funds for album recording with exclusive track tokens

- ✅ **Smart Contract Escrow** - Funds locked until goal met

🎮 Game Development- ✅ Freighter Integration - No private keys in browser

Build indie games with in-game currency as backer rewards- ✅ **Transaction Simulation** - Preview before signing

- ✅ **Automatic Refunds** - Failed campaigns return funds

📚 Book Publishing- ✅ TESTNET Enforcement - Demo safety guaranteed

Fund book publication with signed digital copy tokens- ✅ **Comprehensive Error Handling** - 15+ scenarios covered

🎨 Art Projects ⚠️ Note: This is a testnet demonstration. Production deployment requires security audit.

Support artists with limited edition digital collectibles

📊 Project Status

🔐 Security

~~~
### Current Implementation (Demo/Testnet)✅ Smart Contract:        100% Complete (Deployed)

- ✅ Password-protected wallet encryption✅ Standalone Wallet:     100% Complete (3 connection modes)

- ✅ Client-side session management✅ Backend Logic:         100% Complete

- ✅ localStorage data persistence✅ Error Handling:        100% Complete

- ✅ Input validation and sanitization✅ Documentation:         100% Complete (Professional)

- ✅ Balance validation before transactions⏳ Campaign UI:           In Development

⏳ Demo Testing:          Pending asset creation

### Production Considerations```

- 🔒 Backend authentication server required

- 🔒 JWT token-based sessions**Current Phase:** Standalone wallet complete, campaign UI implementation in progress

- 🔒 Smart contract security audit

- 🔒 Rate limiting and DDoS protection---

- 🔒 HTTPS enforcement

- 🔒 Password hashing (bcrypt/argon2)## 📚 Documentation



**⚠️ Note:** This is a testnet demonstration. Do NOT use real funds or deploy to mainnet without proper security audit.All
comprehensive documentation is in the `/docs` folder:



---- **[COMPLETE-WORKFLOW.md](docs/COMPLETE-WORKFLOW.md)** - Step-by-step user journeys for Alice, Bob, and Charlie

- **[IMPLEMENTATION-SUMMARY.md](docs/IMPLEMENTATION-SUMMARY.md)** - Technical implementation details and testing

## 🐛 Troubleshooting- **[PROJECT-STATUS.md](docs/PROJECT-STATUS.md)** - Current status, metrics, and roadmap

- **[QUICK-REFERENCE.md](docs/QUICK-REFERENCE.md)** - Quick commands and links

### Common Issues

**Frontend Integration:**

**Issue:** Balance shows 0 XLM- **[frontend/INTEGRATION-GUIDE.md](frontend/INTEGRATION-GUIDE.md)** - Complete API
documentation for developers

```javascript

// Solution: Initialize balance manually---

window.walletBalance.init("YOUR_PUBLIC_KEY", 10000)
~~~

```
location.reload()## 🤝 Contributing
```

Built for the **Stellar Build-a-thon**!

Issue: User not showing in header

~~~
// Solution: Check current user1. Fork the repository

window.whoAmI()2. Create a feature branch

// If null, login again3. Make your changes

```4. Submit a pull request


**Issue:** Campaign not updating---

```javascript

// Solution: Refresh or check campaigns## 📄 License

window.viewCampaigns()

location.reload()MIT License - See LICENSE file for details
~~~

Issue: Compilation errors

~~~
# Clear cache and restart

rm -rf node_modules/.cache- **Repository:** [GitHub - StellarPledge](https://github.com/Pswaikar1742/StellarPledge)

npm start- **Smart Contract:** [View on Stellar Expert]
(https://stellar.expert/explorer/testnet/contract/CD4L4MPVSJ3RLAUYQ3ID2M75VWVVMDFBTESJIY4UULFFN33X2KNRTJXY)

```- **Stellar Documentation:** [Soroban Docs](https://soroban.stellar.org/)

- **Freighter Wallet:** [freighter.app](https://www.freighter.app/)

---

---

## 📊 Project Status

## 👥 Team

✅ **Complete:**

- Multi-user authentication systemPassionate blockchain developers building the future of creator economy on Stellar!

- Role-based access control

- Wallet integration (Create/Import/Read-Only)---

- Campaign creation wizard

- Pledge functionality with rewards**🌟 StellarPledge: Where Creators Meet Their Community On-Chain 🌟**

- Mock balance system

- Real-time updates across tabs*Last Updated: October 25, 2025*

- Complete documentation
~~~

~~~~
1. **Install Freighter Wallet:**

⏳ **Future Enhancements:**   - Add the [Freighter extension](https://www.freighter.app/) to your browser

- Smart contract integration   - Create or import a wallet

- Backend API for production   - Switch to **Testnet** network in Freighter settings

- Campaign approval/decline flow

- Refund mechanism2. **Fund Your Account:**

- Transaction history   - Go to [Stellar Laboratory](https://laboratory.stellar.org/#account-creator?network=test)

- Email notifications   - Click "Generate keypair" → "Fund account with Friendbot"

- Social sharing features   - You'll receive 10,000 XLM testnet tokens

   - Repeat for 2-3 accounts (Alice, Bob, Charlie)

---

### Demo: Automated Perk Distribution

## 🤝 Contributing

#### Step 1: Create FILMCREDIT Asset (Alice)

Built for the **Stellar Build-a-thon**!

```bash

Contributions welcome:# Create a custom Stellar asset for perk rewards

1. Fork the repository# Note: This will be integrated into the UI in production

2. Create a feature branch (`git checkout -b feature/amazing-feature`)```

3. Commit your changes (`git commit -m 'Add amazing feature'`)

4. Push to the branch (`git push origin feature/amazing-feature`)#### Step 2: Alice Creates Campaign with Perk

5. Open a Pull Request

1. Open **https://localhost:3001**

---2. Click **"Connect Wallet"** → Approve in Freighter

3. Navigate to **"Create Campaign"**

## 📄 License4. Fill in:

   - **Goal:** 1000 XLM

This project is licensed under the MIT License - see the [LICENSE](LICENSE) file for details.   - **Deadline:** 24 hours

   - **Enable Perk:** ✅

---   - **Perk Threshold:** 500 XLM

   - **Reward Token:** FILMCREDIT contract address

## 🔗 Links   - **Reward Amount:** 1 token

5. Click **"Create"** → Sign transaction in Freighter

- **GitHub Repository:** [StellarPledge](https://github.com/Pswaikar1742/StellarPledge)

- **Stellar Network:** [stellar.org](https://stellar.org)#### Step 3: Bob Pledges (No Perk)

- **Soroban Documentation:** [soroban.stellar.org](https://soroban.stellar.org)
~~~~

```
1. Disconnect Alice's wallet

---2. Connect **Bob's wallet** via Freighter

3. Click on Alice's campaign

## 👥 Team4. Pledge **100 XLM**

5. Sign transaction

Passionate blockchain developers building the future of decentralized crowdfunding!6. **Result:** Campaign shows 100/1000 XLM
(10% funded) - No perk transferred



---#### Step 4: Charlie Pledges (Perk Triggered! 🎉)



## 🎉 Acknowledgments1. Disconnect Bob's wallet

2. Connect **Charlie's wallet** via Freighter

- Built on [Stellar](https://stellar.org) blockchain3. Click on Alice's campaign

- UI components inspired by [shadcn/ui](https://ui.shadcn.com)4. Pledge **500 XLM**

- Animations powered by [Framer Motion](https://www.framer.com/motion)5. Sign transaction

- Icons from [Lucide React](https://lucide.dev)6. **Magic Happens:** Two transfers in ONE transaction:

   - ✅ 500 XLM → Campaign escrow

---   - ✅ 1 FILMCREDIT → Charlie (automatic!)



**🌟 StellarPledge: Empowering Creators, Connecting Communities 🌟**#### Step 5: Verify on Block Explorer



*Last Updated: October 26, 2025*```

Visit: https://stellar.expert/explorer/testnet/tx/[TRANSACTION_HASH]

You'll see TWO operations:
1. XLM payment: Charlie → Contract
2. FILMCREDIT transfer: Alice → Charlie (automated by contract!)
```

Step 6: Alice Claims Funds

1. Campaign now shows 600/1000 XLM (60% funded, SUCCESSFUL)
2. Connect **Alice's wallet**
3. Click **"Claim Funds"**
4. Alice receives **600 XLM** (Bob's 100 + Charlie's 500)

🔗 Deployed Smart Contract

- **Network:** Stellar Testnet
- **Contract Address:** `CD4L4MPVSJ3RLAUYQ3ID2M75VWVVMDFBTESJIY4UULFFN33X2KNRTJXY`
- **Explorer:** [View on Stellar Expert]

📋 Smart Contract API

Core Functions

```
// Create a new campaign with optional perk
create_campaign(
    creator: Address,
    goal: u128,
    deadline: u64,
    perk: Option<PerkConfig>  // NEW: Automated perk system
) -> u32

// Pledge to a campaign (automatic perk check)
pledge(
    campaign_id: u32,
    backer: Address,
    amount: u128
)

// Creator claims funds from successful campaign
claim_funds(campaign_id: u32, creator: Address)

// Backer withdraws refund from failed campaign
withdraw_refund(campaign_id: u32, backer: Address)

// Get campaign details
get_campaign(campaign_id: u32) -> Campaign
```

PerkConfig Structure

```
pub struct PerkConfig {
    pub threshold: u128,        // Minimum pledge to qualify
    pub asset_address: Address, // Stellar Asset Contract address
    pub amount: i128,          // Reward amount (in stroops)
}
```

Automated Perk Logic

```
// Inside pledge() function
if let Some(perk) = &campaign.perk {
    if total_backer_pledge >= perk.threshold {
        // Cross-contract call to Stellar Asset Contract
        let perk_token_client = token::Client::new(&env, &perk.asset_address);
        perk_token_client.transfer(&campaign.creator, &backer, &perk.amount);
        log!(&env, "✅ Perk transferred automatically!");
    }
}
```

🎨 Frontend Integration

Custom React Hooks (Complete Backend API)

The frontend provides 10+ custom hooks for easy integration:

```
// Wallet Management
const { publicKey, connectWallet, disconnectWallet } = useWallet();

// Campaign Operations
const { create, isCreating } = useCreateCampaign();
const { makePledge, isPledging } = usePledge();
const { handleClaimFunds } = useClaimFunds();

// Data Queries
```

```
const { campaigns, loading } = useCampaignList();
const { campaign } = useCampaignDetail(campaignId);
const { percentage, remaining } = useCampaignProgress(campaign);

// Perk Features
const { qualifies, perkInfo } = usePerkCheck(campaign, pledgeAmount);
const isCreator = useIsCreator(campaign);

// User Data
const { myCampaigns } = useMyCampaigns();
const { myPledges } = useMyPledges();
```

Complete Documentation

See `/frontend/INTEGRATION-GUIDE.md` for:

- ✅ Complete hook documentation
- ✅ Data type definitions
- ✅ Code examples
- ✅ Error handling patterns
- ✅ Demo implementation guide

🛠️ Development

Build Smart Contract

```
cd smart-contract
cargo build --target wasm32-unknown-unknown --release
soroban contract optimize --wasm target/wasm32-unknown-unknown/release/stellar_pledge_contract.wasm
```

Deploy to Testnet

```
# Deploy contract
soroban contract deploy \
  --wasm target/wasm32-unknown-unknown/release/stellar_pledge_contract.wasm \
  --source-account <YOUR_SECRET_KEY> \
  --rpc-url https://soroban-testnet.stellar.org \
  --network-passphrase "Test SDF Network ; September 2015"
```

Run Frontend Dev Server

```
cd frontend
npm start  # Starts HTTPS server on port 3001
```

🎯 Key Innovations

1. Automated Perk Distribution ⭐

Traditional crowdfunding: Creator manually sends rewards weeks/months later
StellarPledge: Instant, automatic token transfer when threshold met

2. Cross-Contract Calls

Smart contract interacts with Stellar Asset Contract to transfer tokens:

```
let perk_token_client = token::Client::new(&env, &perk.asset_address);
```

```
  perk_token_client.transfer(&creator, &backer, &amount);
```

3. Any Stellar Asset as Reward

- Movie premiere tickets (FILMCREDIT)
- Music album NFTs (ALBUM001)
- Game currency (GAMECOIN)
- Exclusive memberships (VIPPASS)

4. Production-Ready Architecture

- Clean separation: Smart contract ↔ Soroban.js ↔ React Contexts ↔ Custom Hooks
- Modular, testable, maintainable
- TypeScript-ready data structures
- Comprehensive error handling

🔐 Security Features

- ✅ **Freighter Wallet Integration** - No private keys in browser
- ✅ **Transaction Simulation** - Preview before signing
- ✅ **Smart Contract Escrow** - Funds locked until goal met
- ✅ **Automatic Refunds** - Failed campaigns return funds
- ✅ **Time-Based Logic** - Deadline enforcement on-chain
- ✅ **State Validation** - Prevent double-claims/withdrawals

⚠️ **Testnet Note:** This is a demonstration on Stellar Testnet. For production deployment, conduct thorough security audits.

📊 Project Status

```
✅ Smart Contract:     100% Complete (Deployed)
✅ Backend Logic:      100% Complete
✅ Wallet Integration: 100% Complete (Freighter)
✅ State Management:   100% Complete (React Context)
✅ Custom Hooks:       100% Complete (10+ hooks)
✅ Documentation:      100% Complete
⏳ UI Components:      In Development
⏳ Asset Creation:     Manual (CLI) - UI integration planned
```

🎥 Hackathon Presentation

5-Minute Demo Script

Minute 1: Introduction

- "StellarPledge automates creator rewards using Stellar blockchain"
- Show the problem: Manual reward distribution is slow and error-prone

Minute 2: The Innovation

- Explain cross-contract calls to Stellar Asset Contract
- Show the code: Automatic perk transfer logic

Minute 3: Live Demo - Campaign Creation

- Alice creates campaign with 500 XLM perk threshold

- Show perk configuration UI

Minute 4: Live Demo - Automatic Perk

- Bob pledges 100 XLM (no perk)
- Charlie pledges 500 XLM (perk triggers!)
- Show block explorer: Two transfers in one transaction

Minute 5: Impact & Future

- Creator economy enabler: Musicians, filmmakers, game developers
- Any Stellar asset as reward
- Zero trust required - all automated on-chain

🌐 Use Cases

🎬 Film Production

- **Campaign:** Fund indie film with 5000 XLM goal
- **Perk:** Early screening ticket NFT at 1000 XLM pledge

🎵 Music Album

- **Campaign:** Record album with 2000 XLM goal
- **Perk:** Exclusive track token at 500 XLM pledge

🎮 Game Development

- **Campaign:** Build game with 10000 XLM goal
- **Perk:** In-game currency at 2000 XLM pledge

📚 Book Publishing

- **Campaign:** Publish book with 1000 XLM goal
- **Perk:** Signed digital copy at 200 XLM pledge

👥 Contributing

Built for the **Stellar Build-a-thon** by passionate blockchain developers!

Want to contribute?

1. Fork the repository
2. Create a feature branch
3. Make your changes
4. Submit a pull request

📄 License

MIT License - See LICENSE file for details

🔗 Links

- **Repository:** GitHub - StellarPledge
- **Smart Contract Explorer:** View on Stellar Expert

- Stellar Documentation: [Soroban Docs]
- Freighter Wallet: [freighter.app]

🌟 Building the Future of Creator Economy on Stellar 🌟

Last Updated: October 25, 2025

🌐 Use Cases

🎬 Film Production

- **Campaign:** Fund indie film with 5000 XLM goal
- **Perk:** Early screening ticket NFT at 1000 XLM pledge

🎵 Music Album

- **Campaign:** Record album with 2000 XLM goal
- **Perk:** Exclusive track token at 500 XLM pledge

🎮 Game Development

- **Campaign:** Build game with 10000 XLM goal
- **Perk:** In-game currency at 2000 XLM pledge

📚 Book Publishing

- **Campaign:** Publish book with 1000 XLM goal
- **Perk:** Signed digital copy at 200 XLM pledge

👥 Contributing

Built for the **Stellar Build-a-thon** by passionate blockchain developers!

Want to contribute?

1. Fork the repository
2. Create a feature branch
3. Make your changes
4. Submit a pull request

📄 License

MIT License - See LICENSE file for details

🌐 Use Cases

🎬 Film Production

- **Campaign:** Fund indie film with 5000 XLM goal
- **Perk:** Early screening ticket NFT at 1000 XLM pledge

🎵 Music Album

- **Campaign:** Record album with 2000 XLM goal
- **Perk:** Exclusive track token at 500 XLM pledge

🎮 Game Development

- **Campaign:** Build game with 10000 XLM goal
- **Perk:** In-game currency at 2000 XLM pledge

📚 Book Publishing

- **Campaign:** Publish book with 1000 XLM goal
- **Perk:** Signed digital copy at 200 XLM pledge

👥 Contributing

Built for the **Stellar Build-a-thon** by passionate blockchain developers!

Want to contribute?

1. Fork the repository
2. Create a feature branch
3. Make your changes
4. Submit a pull request

📄 License

MIT License - See LICENSE file for details

🌐 Use Cases

🎬 Film Production

- **Campaign:** Fund indie film with 5000 XLM goal
- **Perk:** Early screening ticket NFT at 1000 XLM pledge

🎵 Music Album

- **Campaign:** Record album with 2000 XLM goal
- **Perk:** Exclusive track token at 500 XLM pledge

🎮 Game Development

- **Campaign:** Build game with 10000 XLM goal
- **Perk:** In-game currency at 2000 XLM pledge

📚 Book Publishing

- **Campaign:** Publish book with 1000 XLM goal
- **Perk:** Signed digital copy at 200 XLM pledge

👥 Contributing

Built for the **Stellar Build-a-thon** by passionate blockchain developers!

Want to contribute?

1. Fork the repository
2. Create a feature branch
3. Make your changes

4. Submit a pull request

📄 License

MIT License - See LICENSE file for details

🌐 Use Cases

🎬 Film Production

- **Campaign:** Fund indie film with 5000 XLM goal
- **Perk:** Early screening ticket NFT at 1000 XLM pledge

🎵 Music Album

- **Campaign:** Record album with 2000 XLM goal
- **Perk:** Exclusive track token at 500 XLM pledge

🎮 Game Development

- **Campaign:** Build game with 10000 XLM goal
- **Perk:** In-game currency at 2000 XLM pledge

📚 Book Publishing

- **Campaign:** Publish book with 1000 XLM goal
- **Perk:** Signed digital copy at 200 XLM pledge

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

Languages

● JavaScript 93.6% ● Rust 3.7% ● CSS 2.0% ● HTML 0.7%

Suggested workflows

Based on your tech stack

Grunt Configure

Build a NodeJS project with npm and grunt.

Publish Node.js Package Configure

Publishes a Node.js package to npm.

Node.js Configure

Build and test a Node.js project with npm.

More workflows

Dismiss suggestions