Cookie Jar - Virtual Fund Management App Project Brief

Project Overview

Cookie Jar is a mobile-first decentralized application that functions as a virtual petty cash drawer. Users can create, manage, and claim funds from "cookie jars" (smart contracts) deployed on various blockchain networks. The app allows for controlled fund distribution with features like whitelisting, claim windows, and transaction history.

Core Pages

1. Landing Page

- Two prominent buttons: "Search for a Jar" and "Create a Jar"

- Cookie-themed background imagery

- Footer with GitHub link and attributions to RG & partners, Allo

2. Jar Search Page

- Connect wallet button for authenticated access

- Search field for entering EVM addresses to find accessible jars

- Tabbed results showing:

  - "Claim from Jars" - jars the user can withdraw from

  - "Manage Jars" - jars the user can administer

- Empty state with message: "Looks like the provided address doesn't have access to any cookie jars 😢" with link to create a jar

- Each jar displayed as a card with key information

3. Claimable Jar Detail Page

- Back navigation to search results

- Jar information header with name, description, balance, network
- Withdrawal interface with:
  - Availability status (available now or countdown timer)
  - Maximum withdrawal amount in jar's token
  - Amount selection slider and direct input
  - Reason field (minimum 20 characters)
  - Submit button that enables when inputs are valid
- Claim history section showing all previous transactions

4. Adminable Jar Detail Page

- Jar information header (same as claimable view)
- Fund addition section with:
  - Amount input with token denomination
  - Fee display (minimum 1%, adjustable upward)
  - Dynamic calculation showing jar amount vs. fee
- Withdrawal settings with:
  - Maximum withdrawal limit configuration
  - Claim window duration setting
  - Individual save buttons for each setting (only enabled when changed)
- Access control with whitelist and blacklist management
- Transaction history showing all claims
- Jar ownership management section
- "Danger zone" with jar emptying functionality

5. Jar Creation Page

- Form with fields for:

- Jar owner (defaults to creator's address)
- Jar title and description
- Claim window configuration
- Network selection dropdown
- Token selection with warning that jars can only accept one token type
- Initial funding amount
- Maximum withdrawal amount
- Whitelist and blacklist configuration
- Create button to deploy the jar

Key Components

SearchField

- Input for EVM addresses with validation
- Queries smart contracts across networks to find accessible jars
- Search triggered by button or Enter key

SearchResults

- Tabbed interface with conditional default tab selection
- Container for JarOverviewCard components
- Clear empty state handling

JarOverviewCard

- Displays jar name, description, network, balance, max withdrawal
- Shows claimable timing information
- Includes truncated contract address with copy function

- Clickable to navigate to detailed views

JarHeaderCard

- Comprehensive jar information display
- Network badge and block explorer link
- Balance with token denomination
- Expandable/collapsible description

ClaimHistoryCard

- Total claims counter
- Scrollable transaction list with ClaimTransactionCard components
- Empty state handling

WithdrawCookies

- Dynamic withdrawal availability status
- Amount selection via slider and direct input
- Reason field with validation
- Submit button with conditional enabling

AddFunds

- Amount input with token denomination
- Fee calculation and display (minimum 1%)
- Submit transaction controls

WithdrawalSettings

- Maximum withdrawal limit configuration

- Claim window duration setting with time unit selection
- Individual save buttons that enable only when fields change

AccessControl
- Manages whitelist and blacklist through UserListCard components
- Explanatory tooltips for each access control mechanism

UserListCard
- Address input with validation
- Bulk import/export functionality
- List management controls
- Dedicated save button for changes

JarOwnerCard
- Current owner display with edit option
- Transfer ownership flow with multiple confirmations
- Address validation

Design Requirements
- Mobile-first responsive design
- Clear navigation with back buttons
- Consistent terminology (use "claim window" not "cooldown period")
- Tooltips explaining complex functionality
- Visual distinction between enabled/disabled states
- Loading indicators and transaction feedback

Technical Considerations

- Multi-chain support (Optimism, Base, Gnosis, Cello, Unichain, Arbitrum)
- Support for ETH and all ERC20 tokens
- 1% mandatory fee representation
- Claim window global mechanism (when anyone claims, everyone must wait)
- Address validation and formatting
- Token decimal handling for user-friendly display
- Warning messages for irreversible actions

Special Notes

- Save buttons only enable when corresponding fields change
- Token selection in jar creation needs clear warning about single-token limitation
- Admin pages only accessible to jar owners
- Claimable jars visible to everyone but only whitelist can claim
- User-friendly time displays (days/hours/minutes instead of seconds when appropriate)

This brief provides guidance for creating the UI MVP with Vercel's v0, focusing on simplicity while capturing all essential functionality.